1 Create X.509 certificates with XCA

Document-ID: 108396_en_00
Document-Description: AH EN X.509 CERT XCA

Make sure you always use the latest documentation.
It can be downloaded using the following link phoenixcontact.net/products.

Contents of this document

This section explains briefly how to create X.509 certificates using the tool *XCA*.

XCA provides much more functionality than explained in this document. Please refer to the XCA documentation for further information (http://xca.sourceforge.net/xca.html – 15.09.2017). You can download XCA from http://xca.sourceforge.net. The screenshots and descriptions in this chapter are related to XCA v1.3.2.

1.1 Introduction

The enrollment of certificates requires a certification authority (CA) which issues public key certificates for a specific period of time. A CA can be a private (in-house) CA, run by your own organization, or a public CA. A public CA is operated by a third party that you trust to validate the identity of each client or server to which it issues a certificate.

There are several tools available for creating and managing certificates, as for example *Microsoft Certification Authority (CA) Server*, *OpenSSL* and *XCA*.

This application note explains how to create X.509 certificates with the tools **OpenSSL** and **XCA** for setting up a VPN connection using X.509 certificates as authentication method.

The scope of this document is not to be a complete user's guide for the described tools. It shall help you getting familiar with them and to create the required certificates in a short term.

1.1.1 XCA - X Certificate and key management

XCA is intended for the creation and management of X. 509 certificates, certificate requests, RSA, DSA and EC private keys, smart cards and CRLs. Everything that is required for a CA is implemented. All CAs can sign sub-CAs recursively.

For enterprise-wide use, templates are available that can be used and adapted to generate certificates or certificate request. All crypto data is stored in an endian-agnostic file format portable across operating systems.

1.2 Create an XCA database

To create X.509 certificates and keys using XCA you need to create a database first. Proceed as follows:

1. Click **File >> New DataBase**.
2. Specify the filename and the storage location of the database.
3. Click **Save**.
4. Enter a password which protects the database against unauthorized usage. The password will be requested every time you open the XCA database.

1.2.1 Open an XCA database

When restarting XCA, you need to reconnect to a database first. To open an already created database, proceed as follows:

1. Click **File >> Open DataBase**.
2. Select the desired database (file **.xdb*).
3. Click **Open**.

1.2.2 Set default hash algorithm

NOTE: Phoenix Contact recommends using secure and up to date encryption and hash algorithms, as stated in the mGuard Software Reference Manual, available at phoenixcontact.net/products (search for "UM EN MGUARD", choose a product and select the manual in the download area).

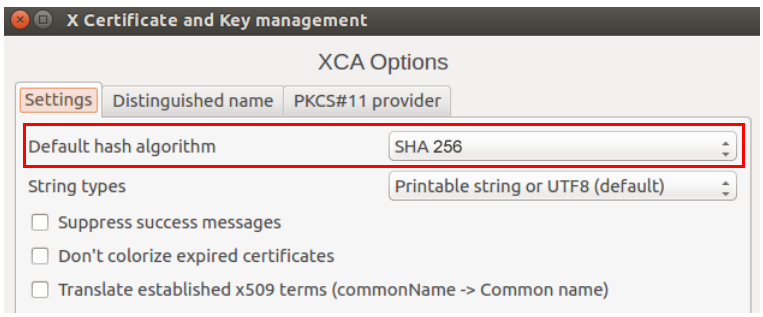
Before you start creating certificates, you should set the default hash algorithm to **SHA 256**. If you don't set the default hash algorithm to SHA 256 you will need to do it every time creating a new certificate.

NOTE: Not all appliances support the functionality of the SHA 2 family

If you are unsure, if all of your appliances support the functionality of the SHA 2 family, the less secure SHA 1 algorithm might be used instead (not recommended by PHOENIX CONTACT and not in accordance with ANSSI-CSPN-2016-09).

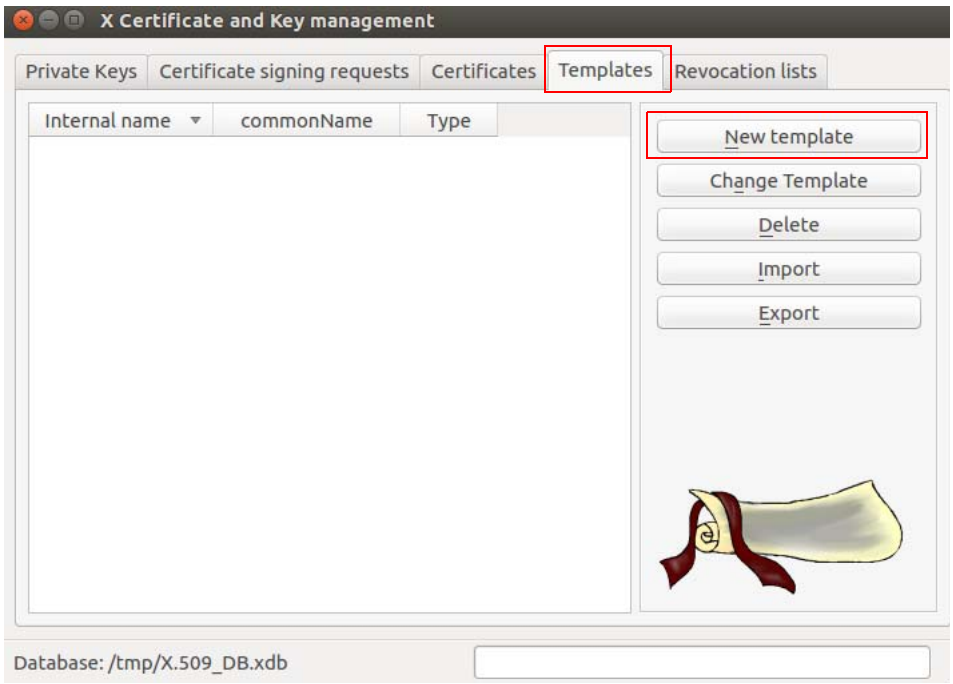
Proceed as follows:

- Click **File >> Options** and set the default hash algorithm to SHA 256 (or the algorithm you will use in your setup).



1.3 Create a certificate template

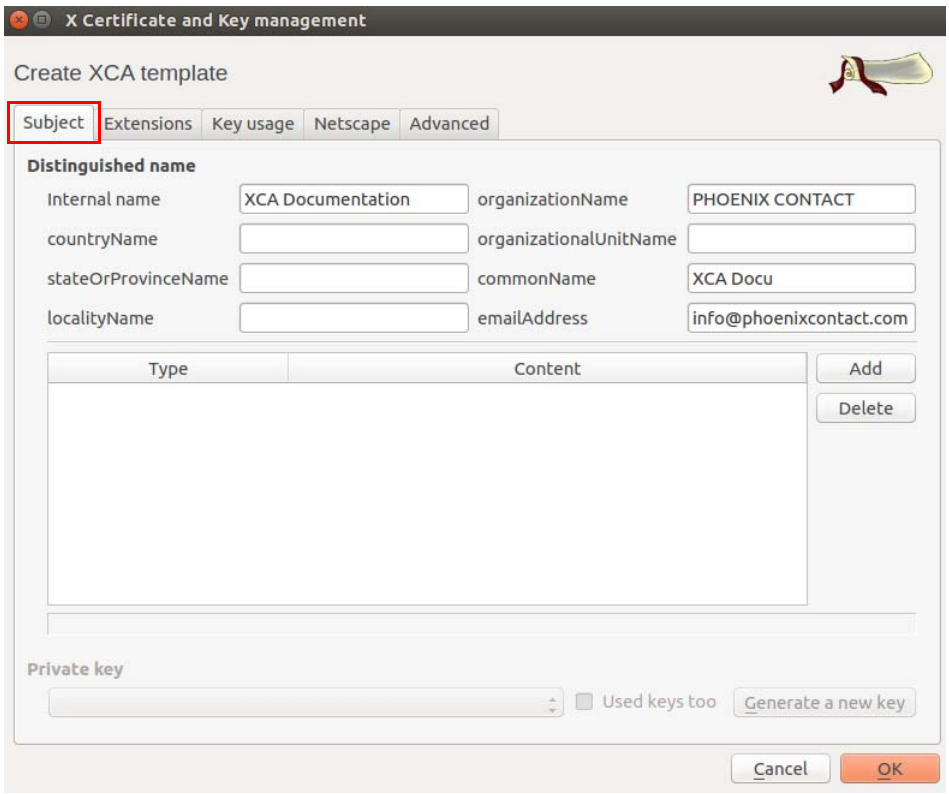
If you need to create more than one certificate it is useful to define a template for consistency reasons and less typing. This template can be used when creating the certificates.



Proceed as follows:

1. Move to the tab **Templates**.
2. Click **New template**.
3. Select the **Preset Template Values** and click **OK**.

1.3.1 Create XCA template >> Tab: Subject

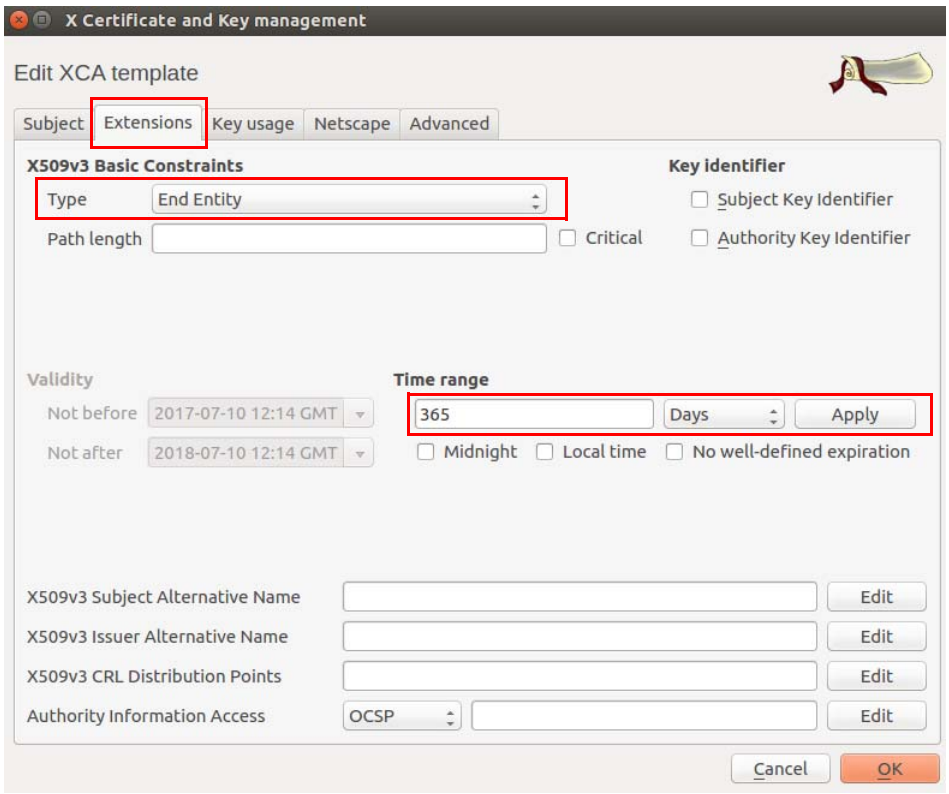


Proceed as follows:

1. Move to the tab **Subject**
2. Use the entry fields from **Internal name** to **emailAddress** for entering the identifying parameters that shall be common for all certificates.

   The template will be stored in *XCA* under the **Internal name**.
3. Move to the tab **Extensions**.

1.3.2 Create XCA template >> Tab: Extensions

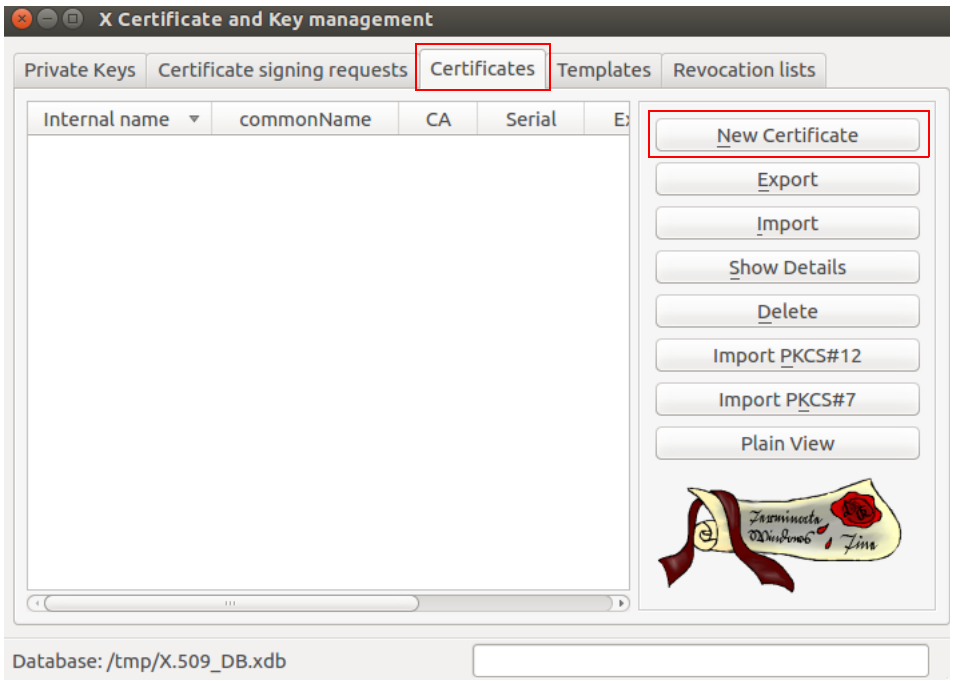


Proceed as follows:

1. In Section **X509v3 Basic Constraints**:
   - Set the **Type** to *End Entity* if you want to use the template for creating client certificates.
   - Set the **Type** to *Certification Authority* if the template should be used for creating CA certificates.
2. In Section **Time Range**:
   - Set the default lifetime of the certificates and click **Apply**.
3. Click **OK** to create the template.

1.4 Create a CA Certificate

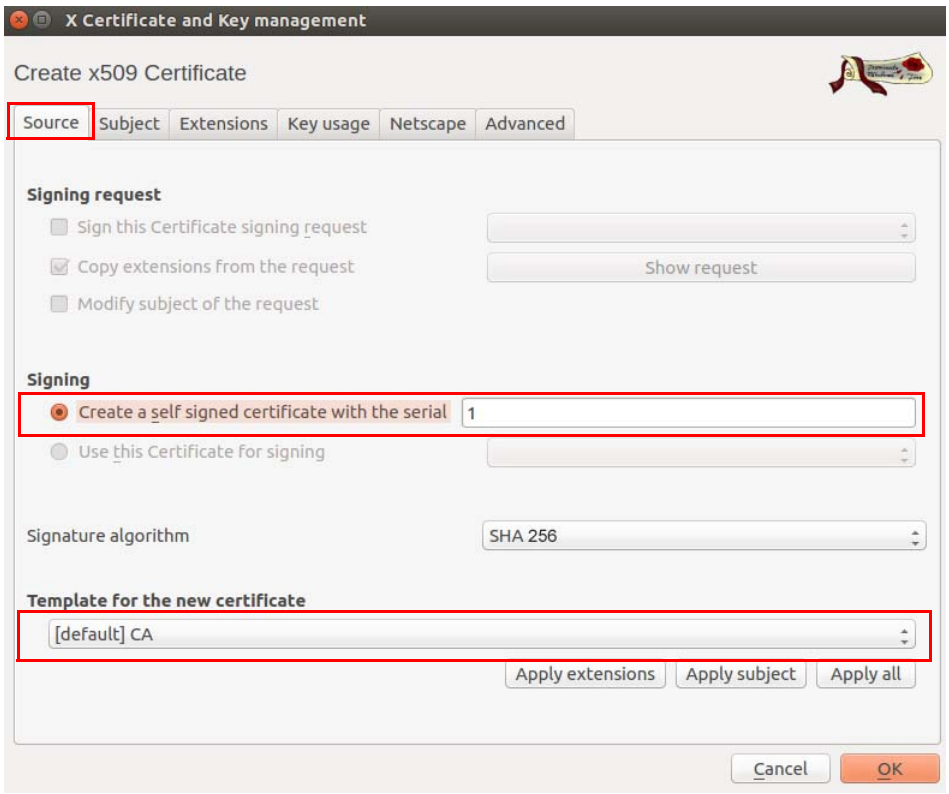
If you don't use self signed client certificates, a client certificate must be signed by the CA certificate to become a valid certificate. Therefore you need to create the CA certificate first before creating the client certificates. The CA certificate is a self signed certificate.



Proceed as follows:

1. Move to the tab **Certificates**.
2. Click **New Certificate**.

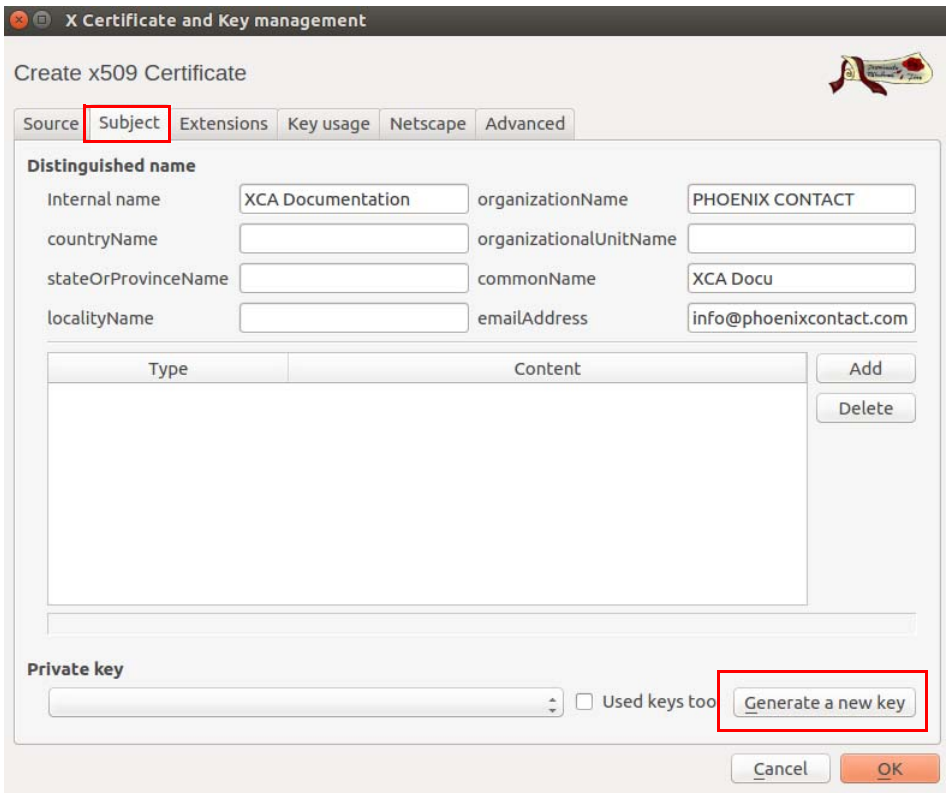
1.4.1 Create x509 (CA) Certificate >> Tab: Source



Proceed as follows:

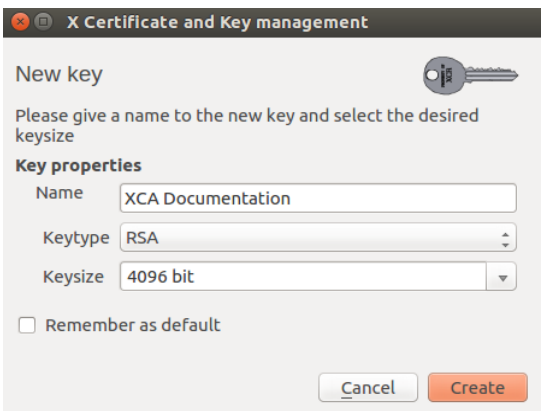
1. Move to the tab **Source**.
2. In Section **Signing**: Ensure that **Create a self signed certificate with the serial** is selected.
3. You may enter a serial number for the certificate or leave the default value.
4. In Section **Template for the new certificate**: If you have created a template for creating CA certificates, you may select it and click **Apply**.
5. Move to the tab **Subject**.

1.4.2 Create x509 (CA) Certificate >> Tab: Subject



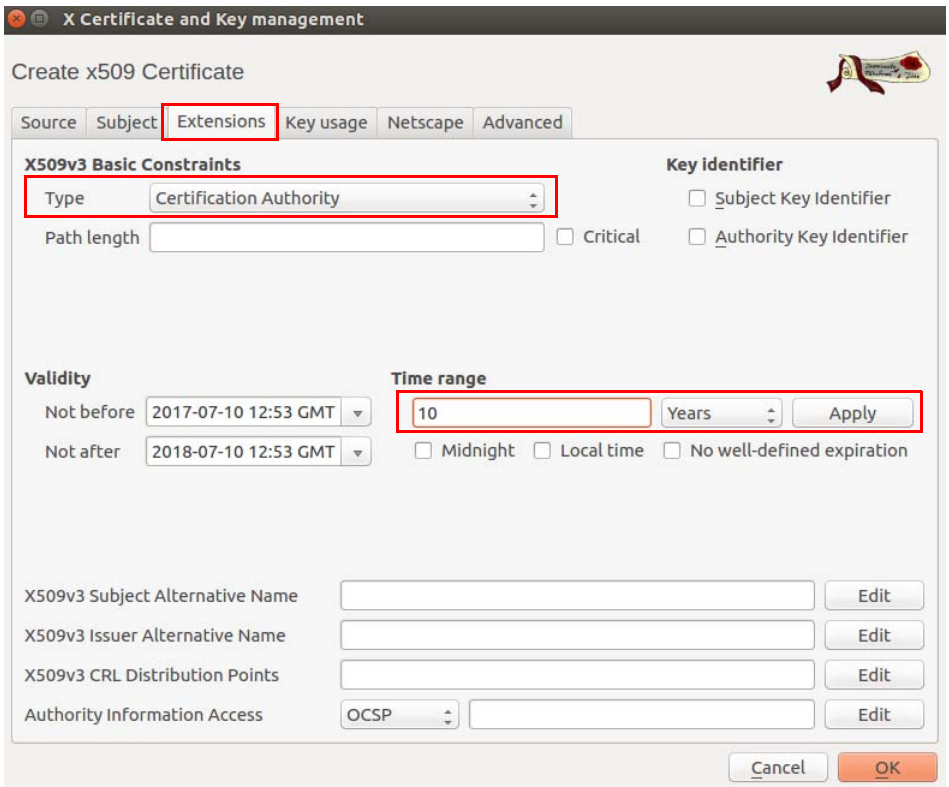
Proceed as follows:

1. In Section **Distinguished name**: Use the entry fields from **Internal name** to **emailAddress** for entering the identifying parameters of the CA.
2. In Section **Private key**: Click **Generate a new key** for creating the private RSA key for the CA.



3. Enter a **Name** for the key, specify the desired **Keytype** and **Keysize** and click **Create**.
4. Move to the tab **Extensions**.

1.4.3 Create x509 (CA) Certificate >> Tab: Extensions



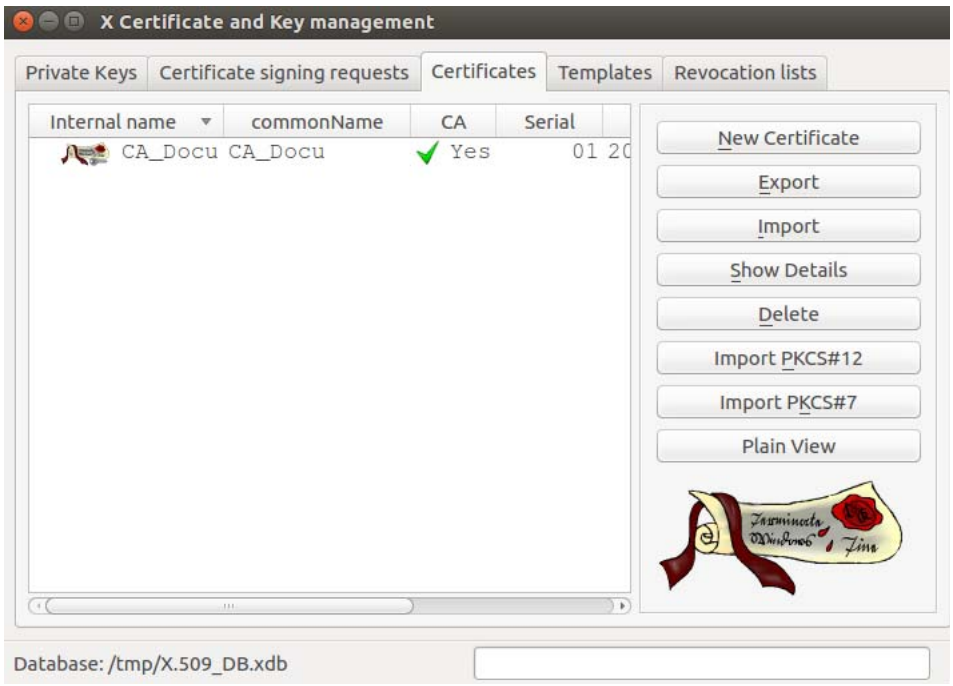
Proceed as follows:

5. In Section **X509v3 Basic Constraints**: Set the **Type** to *Certification Authority*.
6. In Section **Time Range**: Set the default lifetime of the certificates and click **Apply**. For a CA certificate you may want it to last longer than the client certificates so that you do not have to reissue the certificates so often. A lifetime of 10 years might be a good value.
7. Click **Apply**.
8. Click **OK** to create the certificate.
   The CA certificate is displayed in the tab **Certificates**.

1.5 Create a Client Certificate

If you want to create client certificates, you have to create or import a CA certificate first, which will be used to sign the client certificate. By signing the client certificate with the CA certificate, it becomes valid.

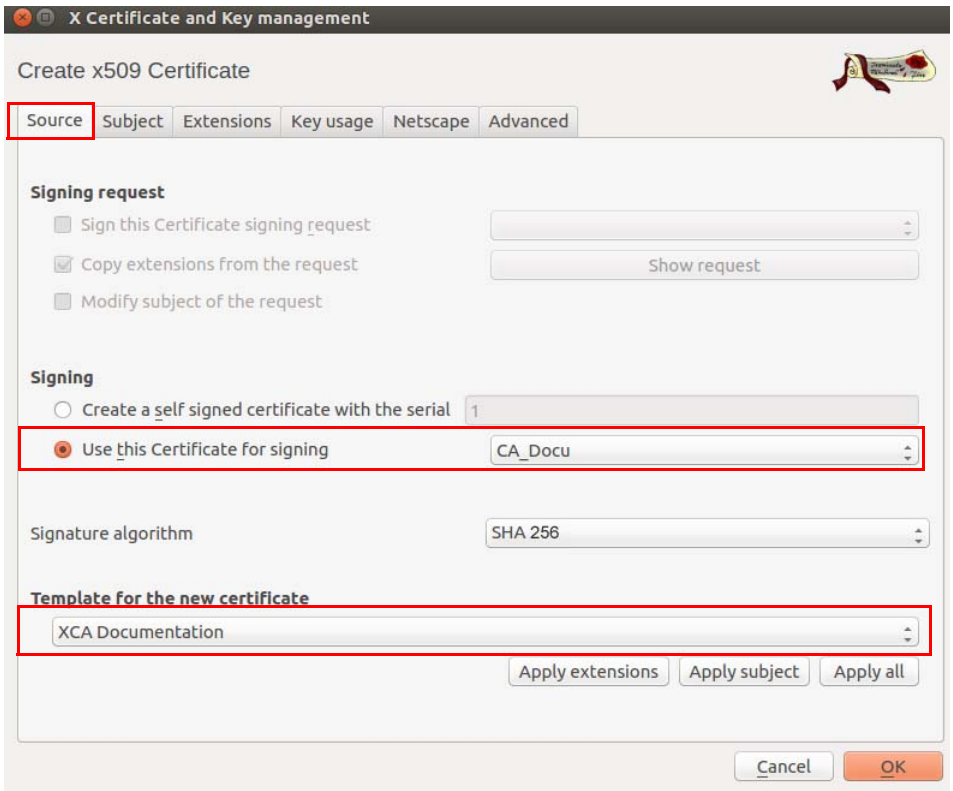
A CA certificate to sign the client certificate must be available in the *XCA* database. If it is not available it has to be created first (see “Create a CA Certificate” on page 7).



Proceed as follows:

1. Move to the tab **Certificates**.
2. Click **New Certificate**.

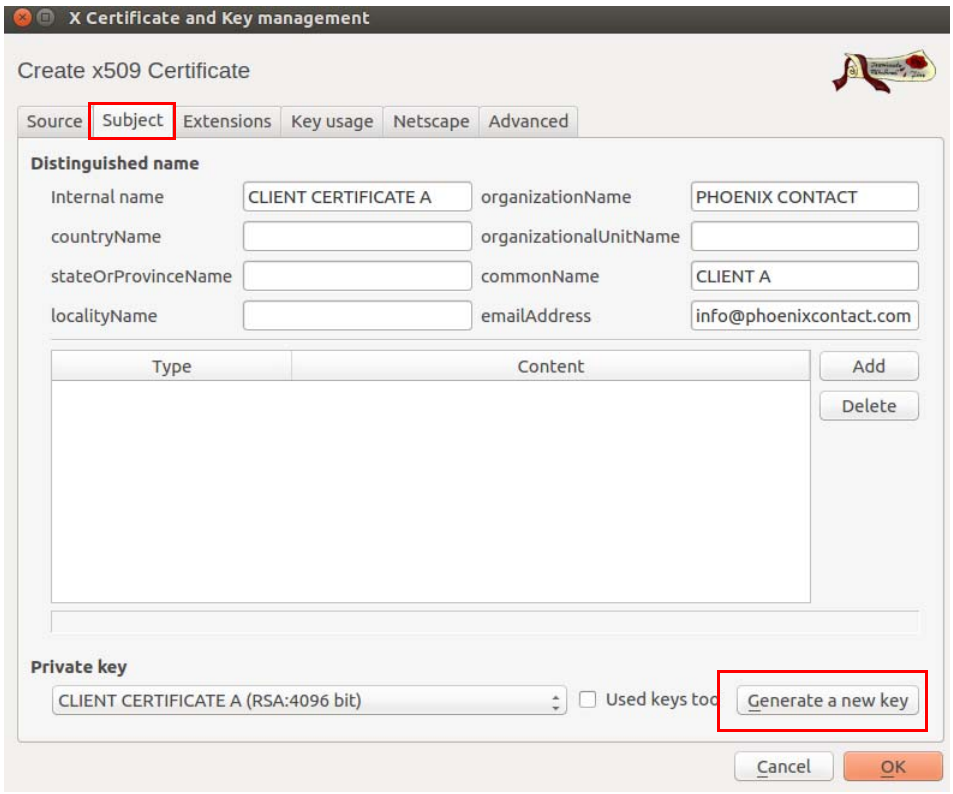
1.5.1 Create x509 (Client) Certificate >> Tab: Source



Proceed as follows:

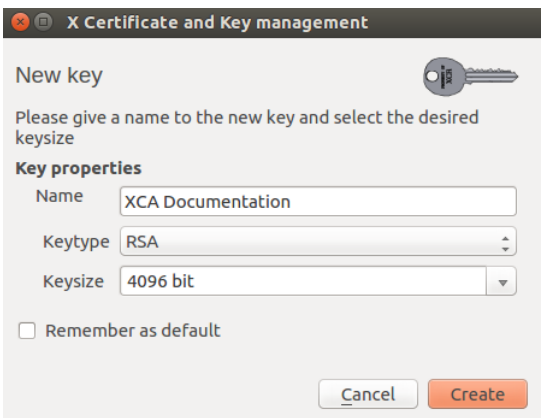
1. Move to the tab **Source**.
2. In Section **Signing**: Ensure that the correct CA is selected in the field **Use this certificate for signing**.
3. In Section **Template for the new certificate**: If you have created a template for creating client certificates, you may select it and click **Apply**.
4. Move to the tab **Subject**.

1.5.2 Create x509 (Client) Certificate >> Tab: Subject



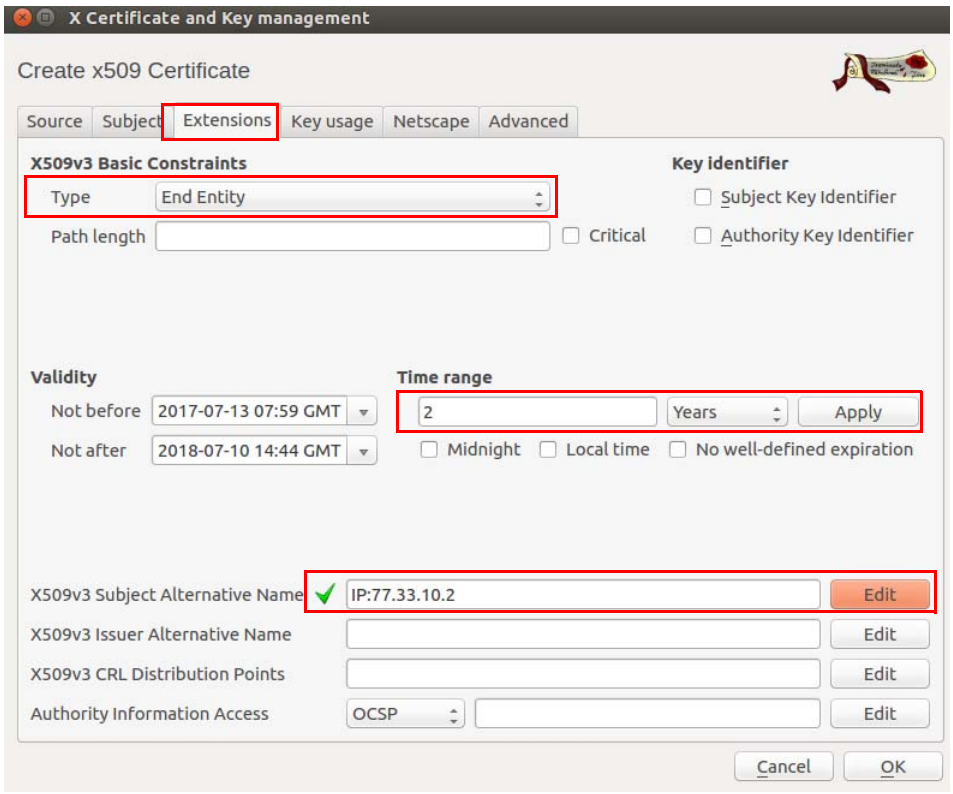
Proceed as follows:

1. In Section **Distinguished name**: Use the entry fields from **Internal name** to **emailAddress** for entering the identifying parameters of the client certificate.
2. In Section **Private key**: Click **Generate a new key** for creating the private RSA key for the certificate.



3. Enter a **Name** for the key, specify the desired **Keytype** and **Keysize** and click **Create**.
4. Move to the tab **Extensions**.

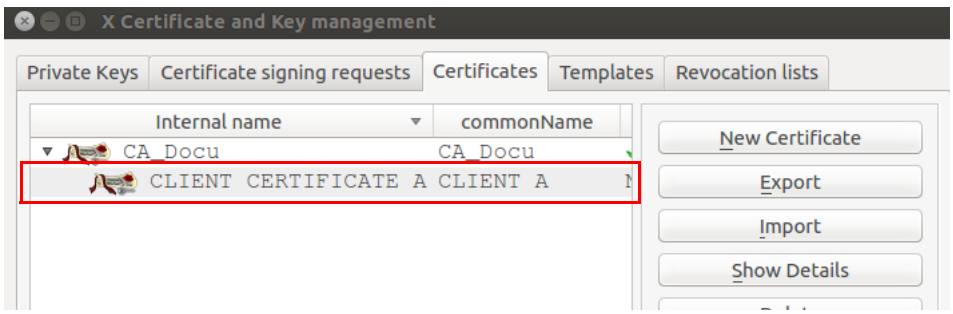
1.5.3 Create x509 (Client) Certificate >> Tab: Extensions



1. In Section **X509v3 Basic Constraints**: Set the **Type** to *End Entity*.
2. In Section **Time Range**: Set the default lifetime of the certificates and click **Apply**.
3. The mGuard uses as default VPN identifier the subject name of the certificate. If you want to use another VPN identifier (e. g. email address, hostname or IP address), this identifier must be present in the certificate as **subject alternative name**.

   To add another identifier, click **Edit** in the line **X509v3 Subject Alternative Name**, select the identifier type (email, DNS or IP), enter its value, click **Add** and then **Apply**.
4. Click **OK** to create the certificate.

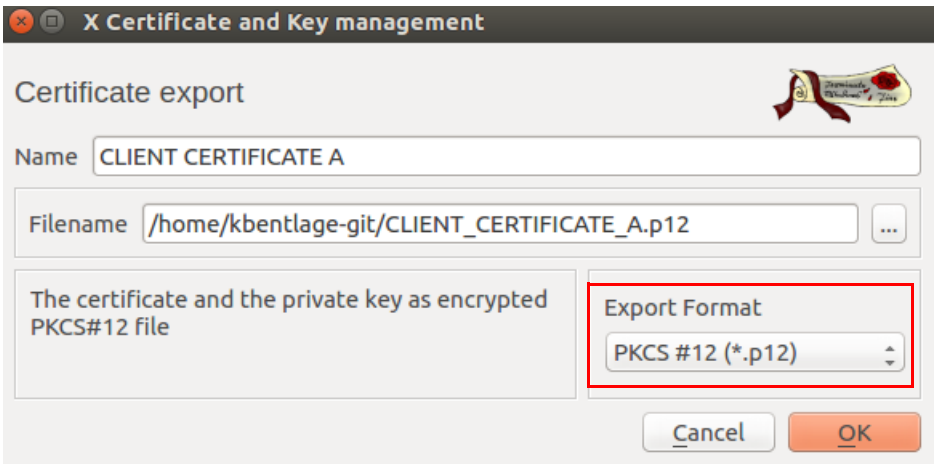
   The client certificate will be displayed in the tab **Certificates** beneath the CA certificate.



1.6 Export a certificate

To export a certificate created with XCA, proceed as follows:

1. Move to the tab **Certificates**.
2. Highlight the certificate that shall be exported.
3. Click **Export**.



4. Select the **Export Format** (PEM or PKCS#12 – see info box below).
5. Specify the desired **Filename** and the location where the export should be stored.
6. Click **OK**.
7. If you export the certificate as PKCS#12 then you'll be prompted to enter a password which protects the export against unauthorized usage. Enter the Password and click **OK**.

PKCS (Public Key Cryptography Standards)

PKCS #12: Personal Information Exchange Syntax v1.1 (defined in **RFC 7292**)

PKCS #12 v1.1 describes a transfer syntax for personal identity information, including private keys, certificates, miscellaneous secrets, and extensions. Machines, applications, browsers, Internet kiosks, and so on, that support this standard will allow a user to import, export, and exercise a single set of personal identity information. This standard supports direct transfer of personal information under several privacy and integrity modes (RFC 7292).

PEM (privacy-enhanced mai) (defined in RFC's 1421 through 1424)

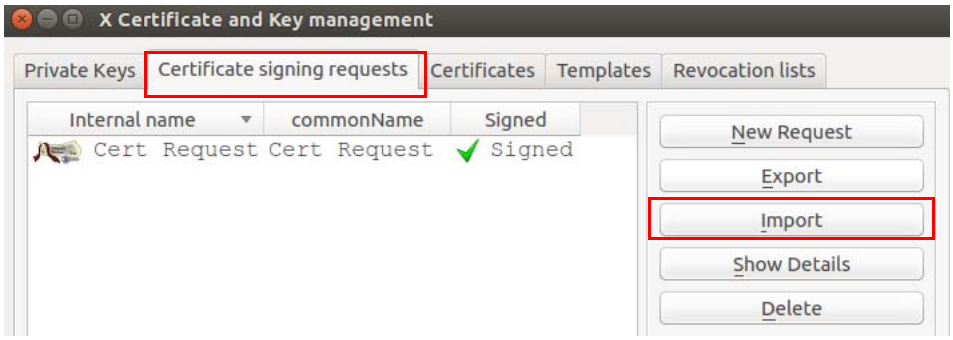
A PEM container may include just the public certificate or an entire certificate chain (including public key, private key, and root certificates).

PEM data is commonly stored in files with a "**.pem**" or "**.cer**" suffix or a "**.crt**" suffix (for certificates), or a "**.key**" suffix (for public or private keys).

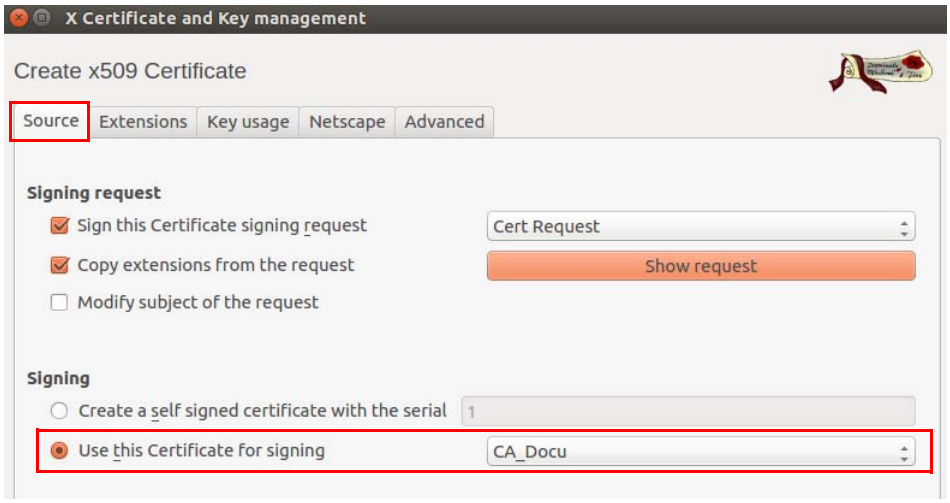
1.7 Sign a Certificate Request with the CA

To sign a certificate request, proceed as follows:

1. Move to the tab **Certificate signing requests**.
2. Click **Import**.
3. Select a certificate request (PKCS#10 file) which should be signed by the CA and click **Open**.
4. The imported certificate request is displayed in the tab **Certificate signing requests**.



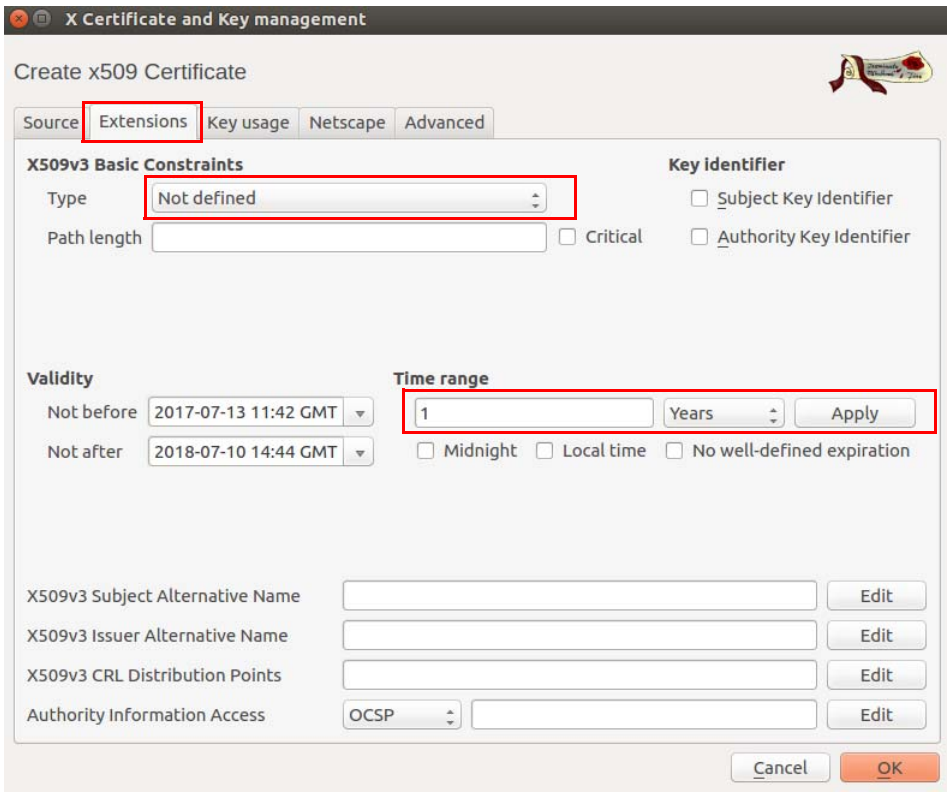
1.7.1 X Certificate and Key Management >> Tab: Source



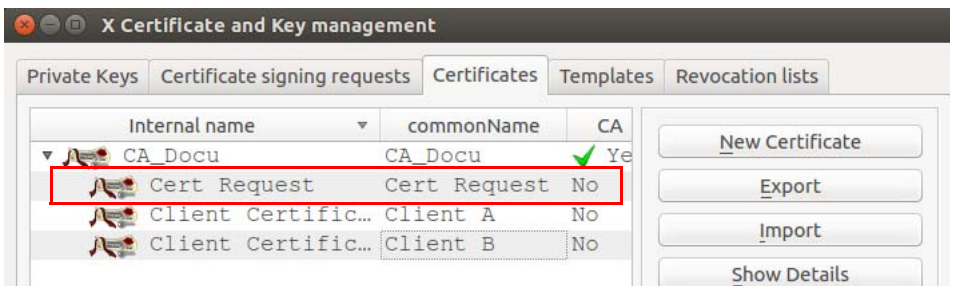
To sign the certificate request, proceed as follows:

1. Move to the tab **Certificate signing requests**.
2. Right click the certificate request and select **Sign** from the context menu.
3. In Section **Signing**: Ensure that the correct CA certificate is selected in the field **Use this certificate for signing**.
4. Move to the tab **Extensions**.

1.7.2 X Certificate and Key Management >> Tab: Extensions



1. In Section **X509v3 Basic Constraints**: Leave **Type** as *Not defined*. Otherwise XCA would copy the certificate extensions twice into the signed certificate.
2. In Section **Time Range**: Set the default lifetime for the new certificate and click **Apply**.
3. Click **OK**.
4. The signed certificate request is displayed in the tab **Certificates** beneath the CA certificate.



1.8 Using a Certificate Revocation List (CRL)

1.8.1 Revoke a certificate

1. Move to the tab **Certificates**.
2. Right click the client certificate that should be revoked and select **Revoke** from the context menu.
3. Edit the parameters and click **OK**.
4. The revoked certificate is marked with a cross icon  and the **Trust state** is *Not trusted*.

1.8.2 Specify the CRL renewal period

1. Move to the tab **Certificates**.
2. Right click the CA and select **CA >> Properties** from the context menu.
3. Enter the desired renewal period into the field **Days until next CRL issuing**.
4. Click **OK**.

1.8.3 Create the CRL

1. Move to the tab **Certificates**.
2. Right click the CA and select **CA >> Generate CRL** from the context menu.
3. Edit the parameters and click **OK**.
4. The CRL is displayed in the tab **Revocation lists**.

1.8.4 Obtain information about a CRL

1. Move to the tab **Revocation lists**.
2. Highlight the CRL and click **Show Details**.

1.8.5 Export of the CRL

1. Move to the tab **Revocation lists**.
2. Highlight the CRL.
3. Click **Export**.
4. Specify the filename and location for storing the CRL.
5. Chose the export format (DER or PEM).
6. Click **OK**.

1.9 Example: VPN connection between two mGuard devices

To create and import the required certificates for a VPN connection between two mGuard devices, proceed as follows:

CA Certificate

- Create a CA certificate as described in chapter “Create a CA Certificate” on page 7.

Client Certificate

- Create a client certificate for **mGuard #1** and a client certificate for **mGuard #2** as described in chapter “Create a Client Certificate” on page 11.

Export certificates

- Export the certificates as described in chapter “Export a certificate” on page 15.

The following exports are required:

– **mGuard #1** as PKCS#12: This export needs to be imported on **mGuard #1** as a *Machine Certificate* (menu: Authentication >> Certificates, tab *Machine Certificates*).

– **mGuard #2** as PKCS#12: This export needs to be imported on **mGuard #2** as a *Machine Certificate* (menu: Authentication >> Certificates, tab *Machine Certificates*).

– **mGuard #1** as PEM: This export needs to be imported on **mGuard #2** as connection certificate (menu: IPsec VPN >> Connections >> (Edit), tab *Authentication*).

– **mGuard #2** as PEM: This export needs to be imported on **mGuard #1** as connection certificate (menu: IPsec VPN >> Connections >> (Edit), tab *Authentication*).