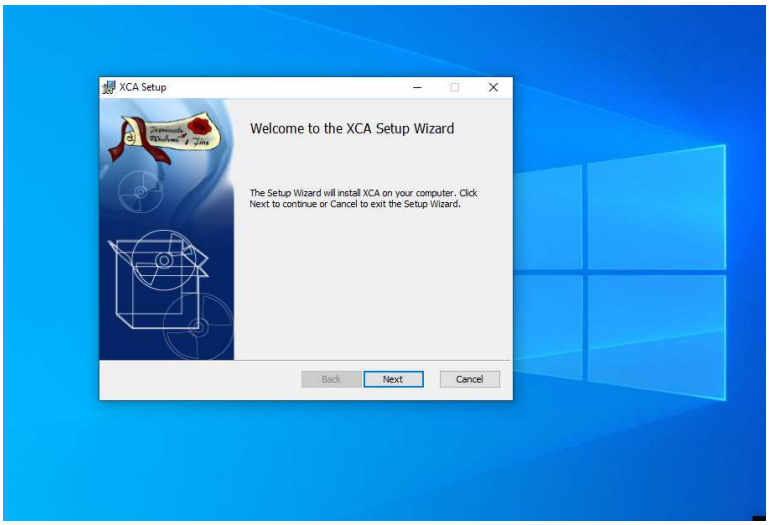
## Creating, Signing and Maintaining Certificates using X-Certificate and Key Management tool:

This application is intended for creating and managing X.509 certificates, certificate requests, RSA, DSA and EC private keys, Smartcards and CRLs. Everything that is needed for a CA is implemented.

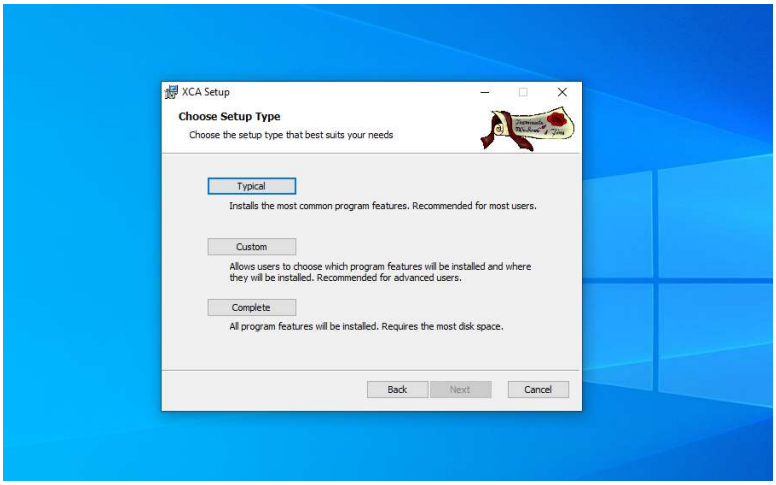
Link to download the tool: https://hohnstaedt.de/xca, http://hohnstaedt.de/xca. Can be installed on any Windows machine and the tool is fairly easy to use for beginners too.

## Steps to Install and use XCA:

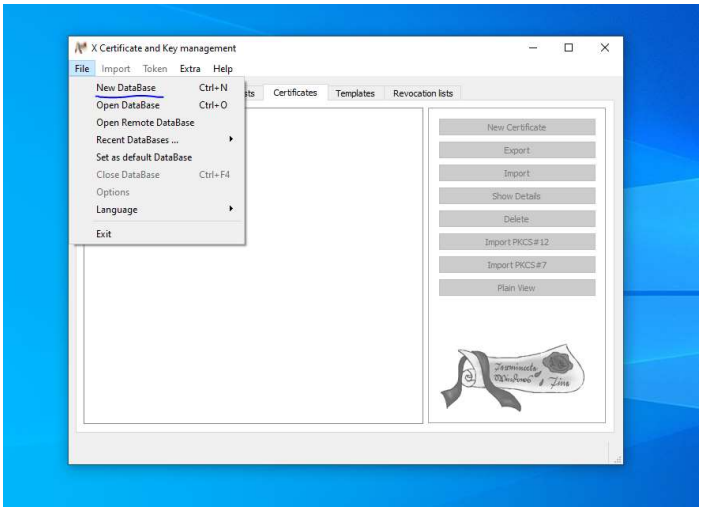
1. Download the software from https://hohnstaedt.de/xca, http://hohnstaedt.de/xca.




2. Click on Complete installation



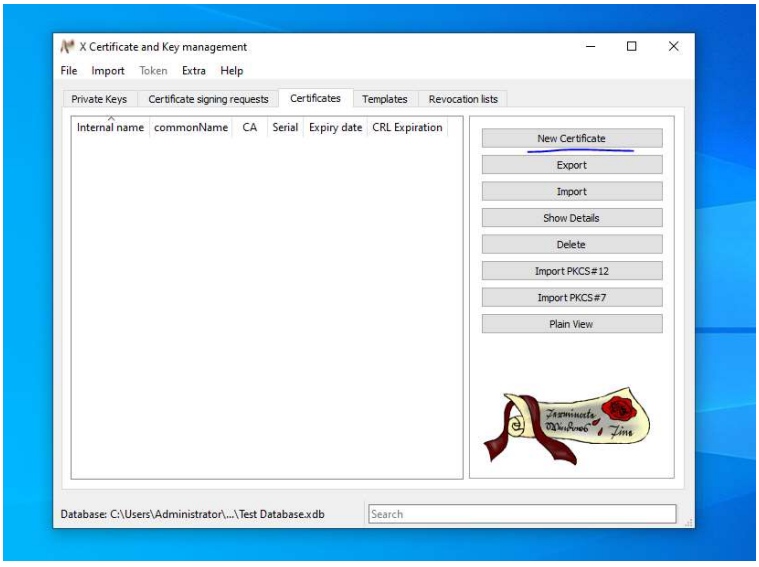
3. Install and Open the XCA application. We can create our own Database by clicking "**New Database**" where the all PKI related key information gets stored permanently and can use it anytime by just "**Open Database**" again. Enter a password (optional) if you want to protect your database from someone accessing your computer.



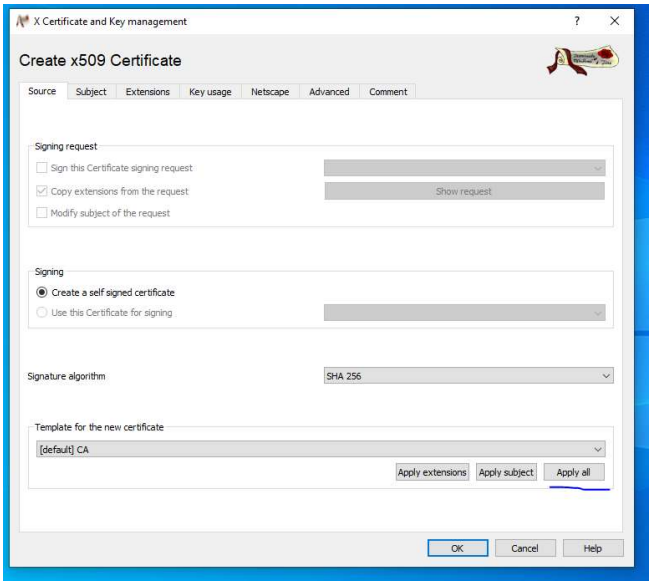

## Creating our own PKI Trust hierarchy by creating RootCA certificate, which can be used to Sign other certificates. Thus we can act as a private Certificate Authority and sign PKI certificate for other devices/users in our organization without having to go to 3rd party CA like GoDaddy etc.

## Steps:

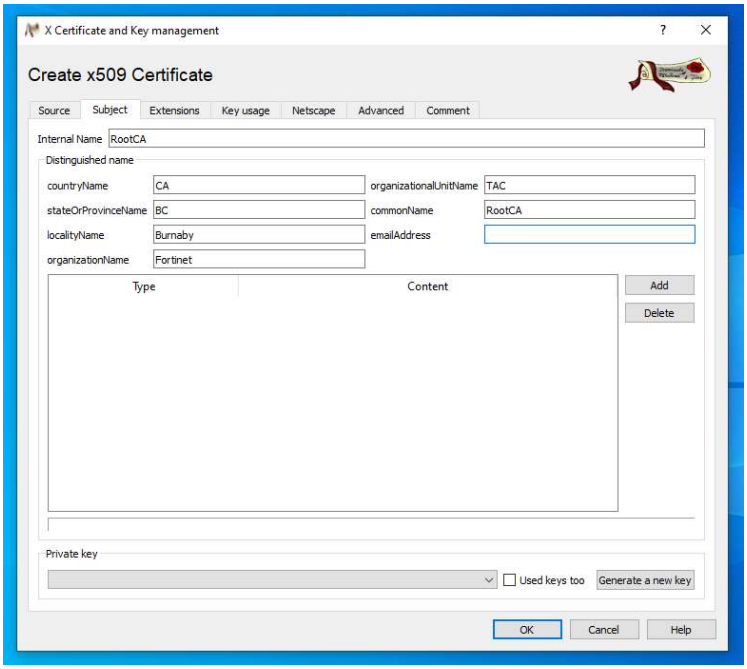
1. Click on create "**New Certificate".**



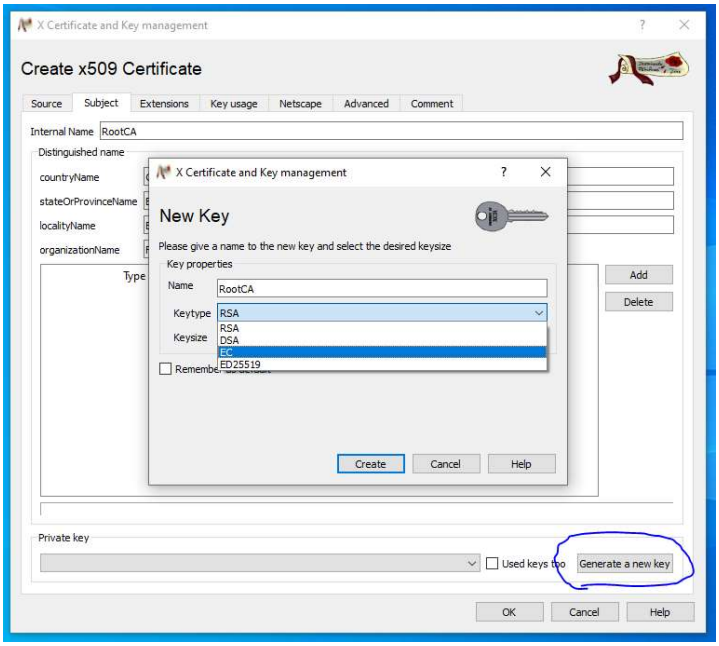
2. A new page pops up. Click on a Template which you want to use. Currently XCA has few default templates already made as seen in the drop down. These templates help us not to worry about certificate attributes which are needed to become a Certificate Authority (example: cA=True and keyUsage=keyCertSign) according to X.509 standard, otherwise we would need to set it manually. For making a CA certificate we select **"[default] CA**" template and select on "**Apply all**" which fills in the required attributes which are needed to create a Root CA Certificate.



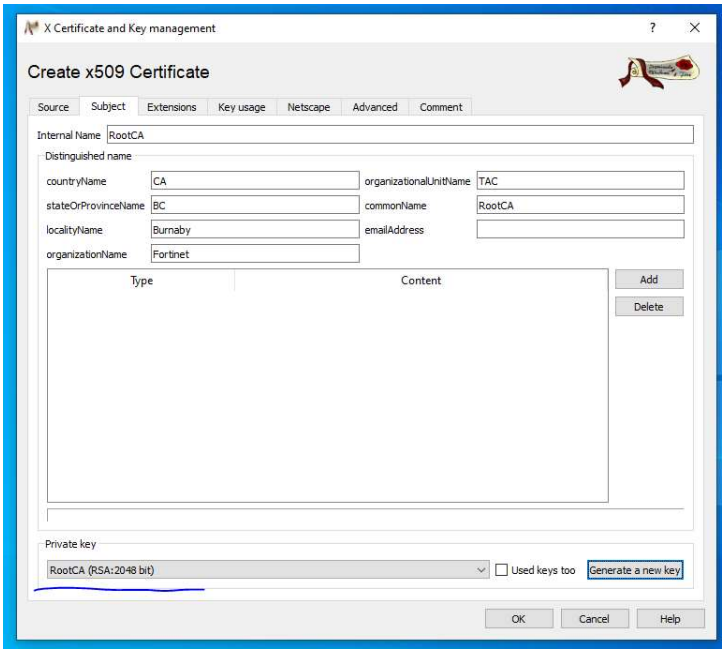

3. Without clicking "**OK",** Go to the "**Subject"** tab and fill the necessary information you need to appear in your CA certificate. "**Internal Name"** is a locally significant name and does not appear in the actual certificate.



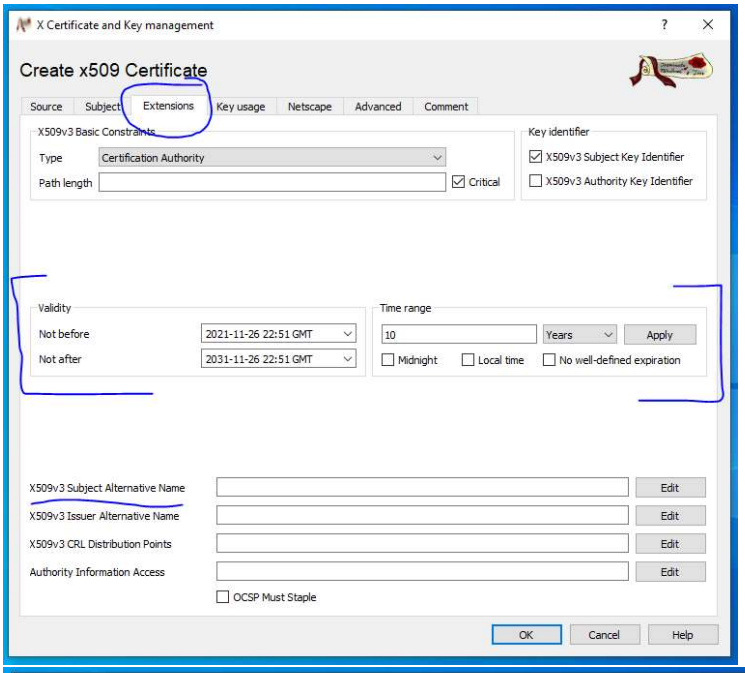
4. Click on "**Generate a new key**", which generates a new private and public keys for your root CA. You can choose the Algorithm (RSA, DSA, EC etc.) to use to generate these new private and public key also called as key pair (private and public key). I have selected RSA algorithm with key size of 2048 bit which are the defaults.

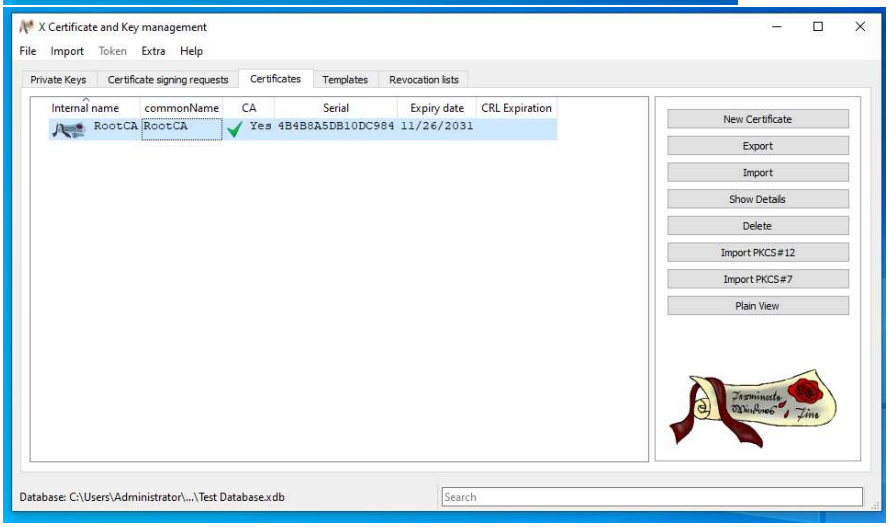



5. Once created the private key from the key pair should appear in the **"Private Key"** as shown below.

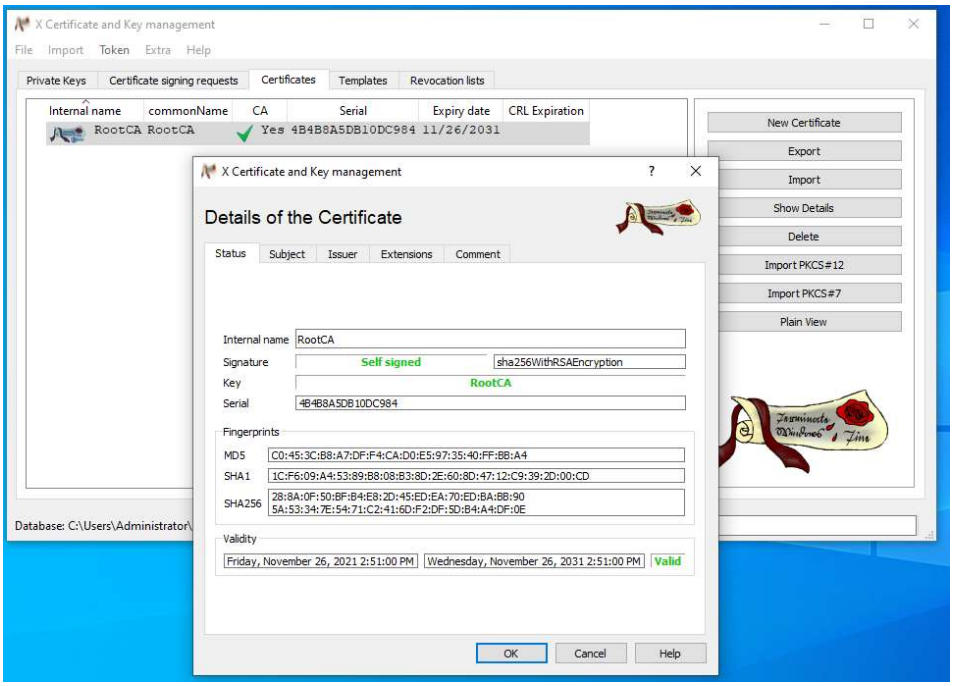


6. Alternatively, you can now Add a "**X509v3 Subject Alternative Name"** (SAN) field in the certificate by clicking on the "**Extensions**" tab. Also you can modify the "**Validity**" dates for this Certificate Authority's Certificate. I will be keeping everything as defaults. After this you can toggle on every tab and see the fields which are automatically filled and highlighted, because of the default template **"[default] CA**" , that we used initially. Now press on **"OK"** at the bottom. Your certificate will now be visible under the "**Certificates"** list.



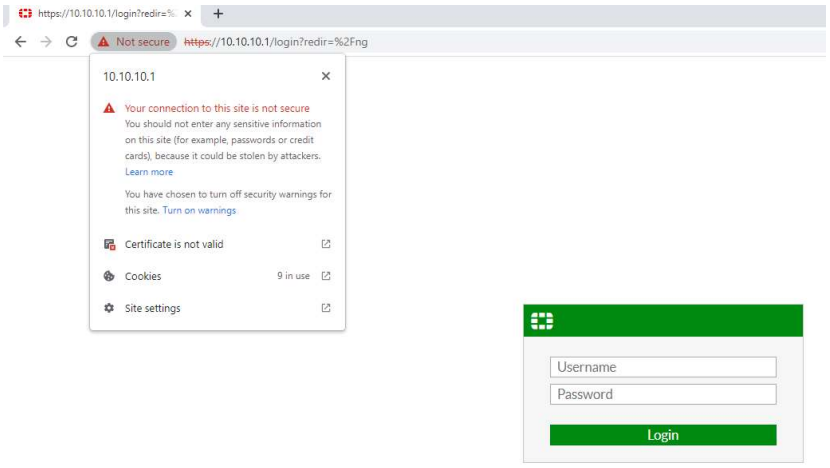

7. If you double click on the certificate it will show up as Self Signed as it's a Root Certificate and all Root Certificates are self-signed certificate. It is signed using the private key that was created when we created the public-private key pair.

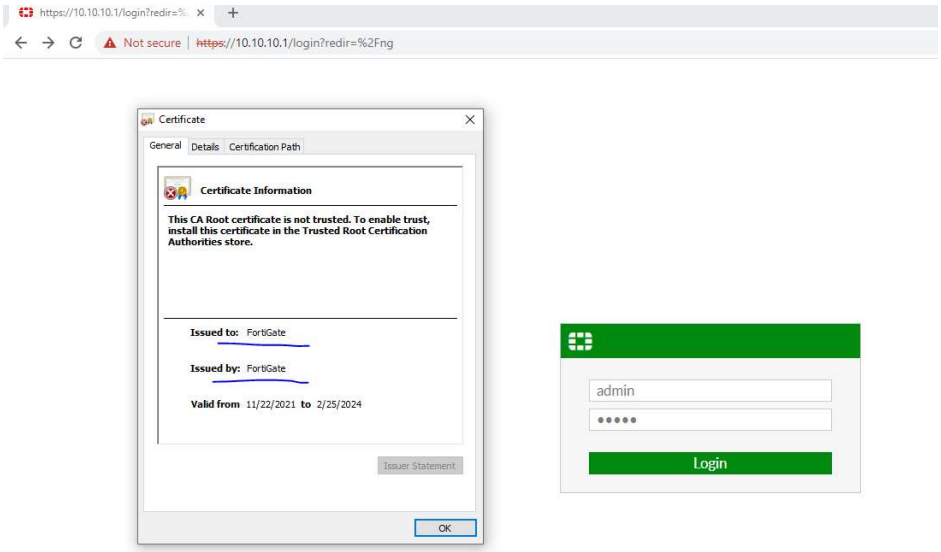



8. Now we can use this Root CA Certificate to Sign the certificate for other devices. Please remember that this is a private Root CA Certificate and by default will not be trusted by any browsers or FortiGate, for it to trust this Root CA Certificate we will need to import this in the Trusted Certificate Authority list of the Browser and FortiGate. Now, let us see how it can be used to Sign Certificate which is used for HTTPS access of the FortiGate in the next section.

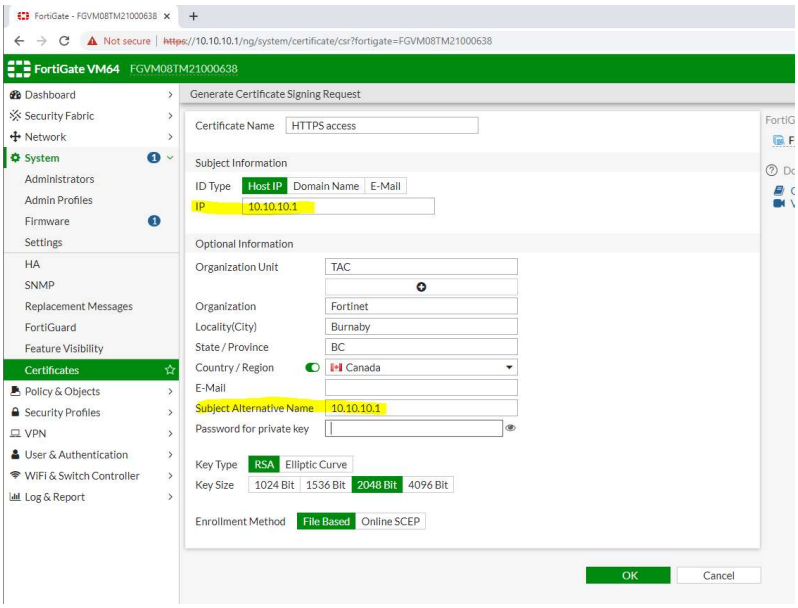
## Creating Certificate for FortiGate which can be used for HTTPS access of the FortiGate.

1. When we HTTPS to the FortiGate's GUI it will show up as Untrusted and will give certificate warnings. This is because it's using FortiGate's default store certificate for HTTPS which is not trusted by the Chrome browser by default and also because the CN name (Issued to) of the certificate is not matching with the name/IP that we have used to access the FortiGate i.e. 10.10.10.1

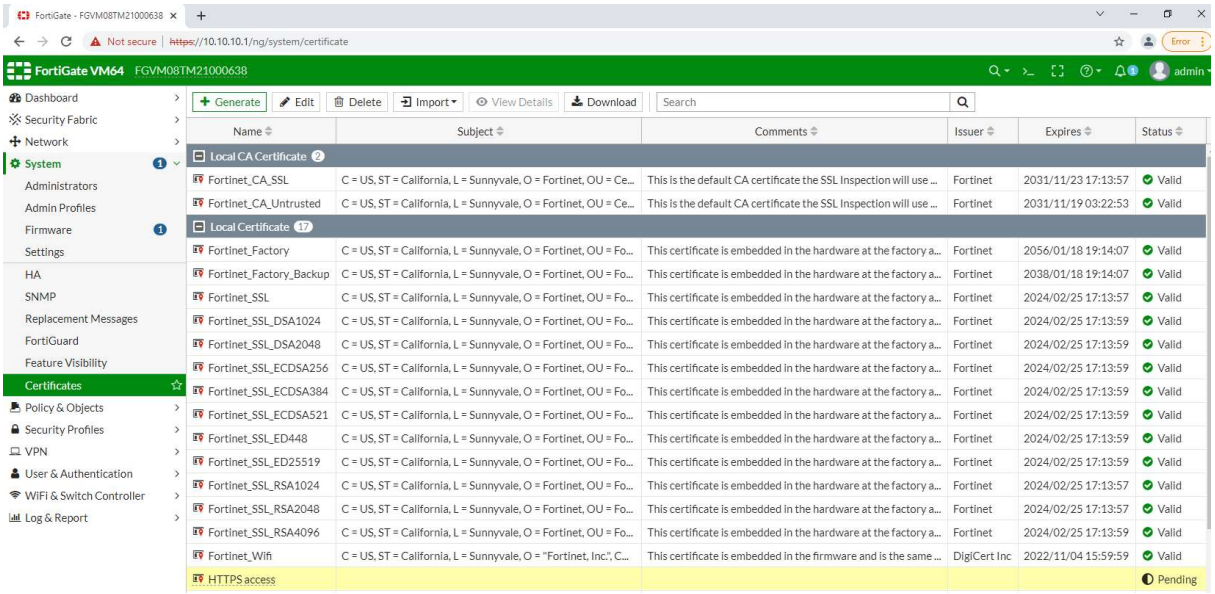



2. To create a certificate for the Fortigate there are 2 methods. 1st method involves creating a Certificate Signing Request (CSR) on the FortiGate itself, which means the public and private key is generated within and by the FortiGate itself. With this method you can download the public key only and give it to the Certificate Authority to sign it, in our case CA is XCA application. Thus your private key is secure and inside the Fortigate, only your public key is public. 2nd method involves letting the Certificate Authority itself generate/create a public and private key for your FortiGate, and then let the same CA sign the public key of the FortiGate. With this method your private key is with Certificate Authority and this maybe a privacy concern. I will demonstrate the 1st method as 2nd method is more or less the same using which we created our Root CA certificate.
3. To generate CSR on the FortiGate, go to **System-→Certificates→Generate→CSR**. Fill in the required details as shown. I have used **"Host IP"** as we are accessing Fortigate using its IP address. If you a have a local DNS record for the hostname of the FortiGate you can use or if you have a public IP to access FortiGate which has a DNS mapping globally then you can use the "**Domain Name"** in the "**Subject Information"** field and access the FortiGate using its Domain Name. Once done Click on **"OK".**

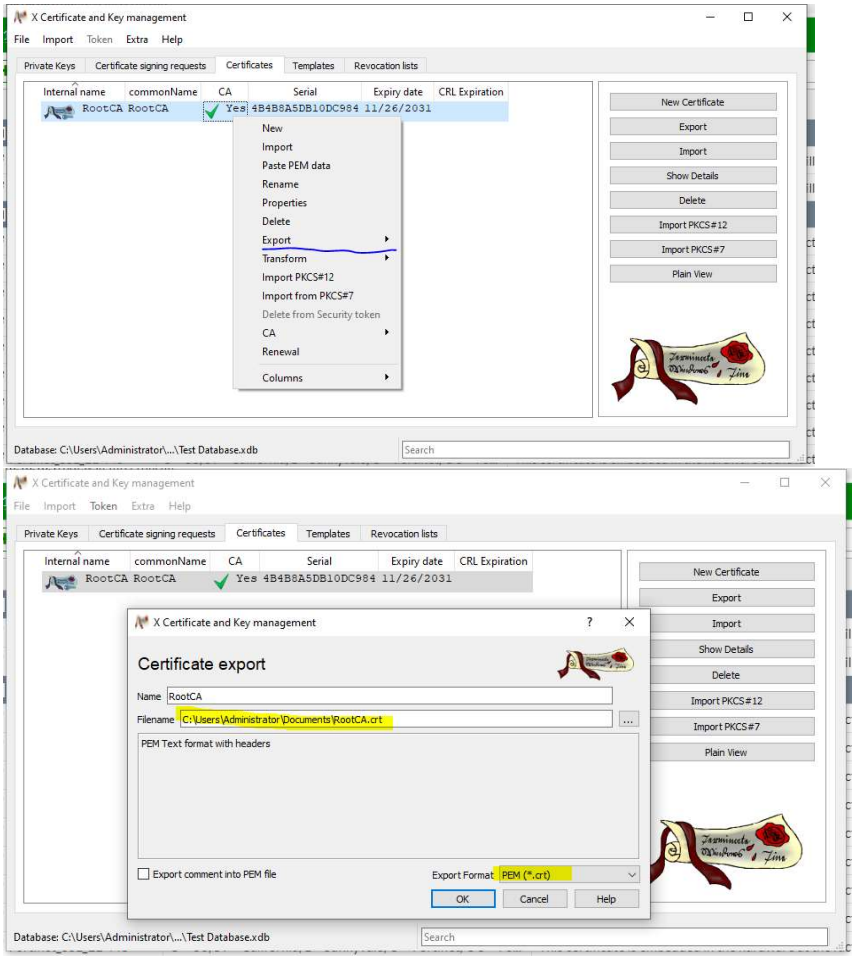



4. The certificate will be shown in the Certificate List and will be shown in **Pending state** as its not yet signed by a Trusted Certificate Authority yet. Right click on the **HTTPS access certificate** and Download it. It will be downloaded in a **.csr** format which means it has just the public key of FortiGate in it along with other details like Subject name, SAN name, OU etc. of the FortiGate which was just filled in the CSR.

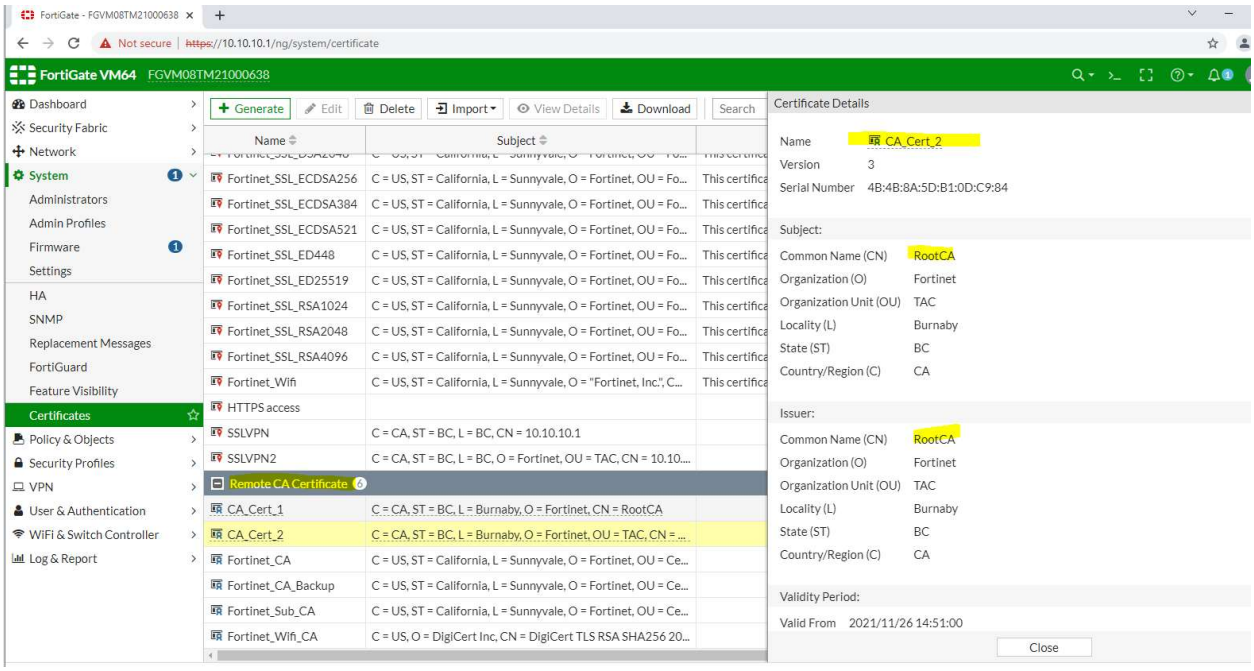



5. As our XCA Application is acting as a private CA, the Root CA certificate made by it is not trusted globally, thus we have to import our newly created Root CA certificate on the Fortigate in its Trusted Authority list, so that the Fortigate starts to trust any certificate signed by this CA, so now let us import the RootCA in the FortiGate by exporting it from the XCA Application.
6. Steps to Export our Root CA from XCA. Right click on the certificate we created and select **Export→ File→LocationOnYourPC.** Here you can also select the Format that you want to export the

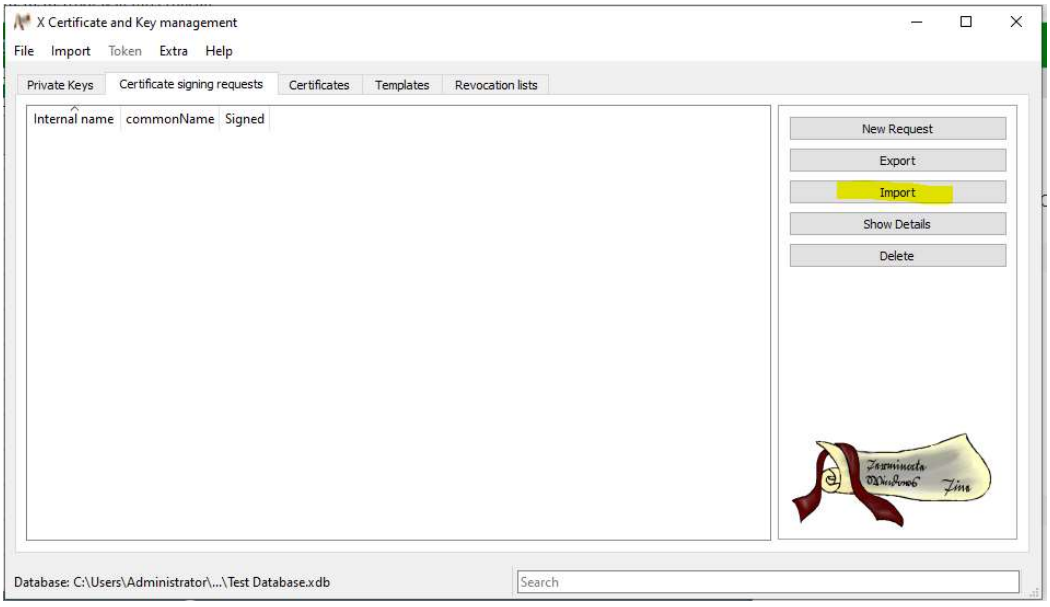
certificate in. Select **.crt** format as .**crt** format just has the public key of the CA, we do not want to export the private key of the CA. Click on "**OK".**

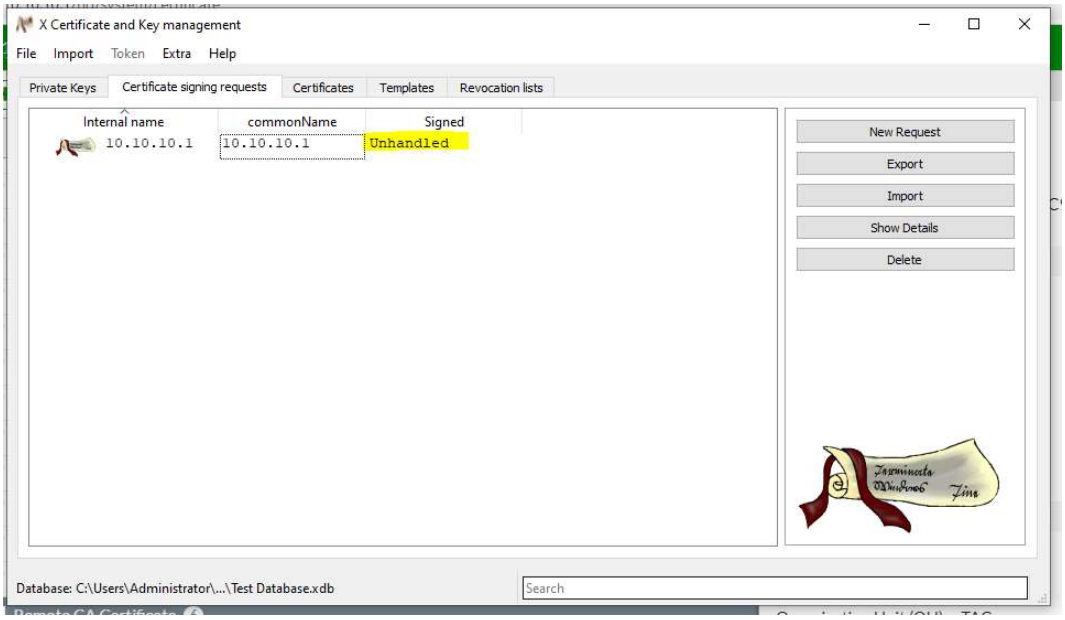



7. Now import this RootCA.crt file on the FortiGate by going to **System→Certificate→Import→CA Certificate→ File→Upload RootCA.crt file**. Once done, this RootCA certificate will now show up in the Remote CA certificate list as **CA_Cert_2** in my case. Now FortiGate will trust any certificate signed by this CA as this certificate is in the Trusted Root Certificate store.

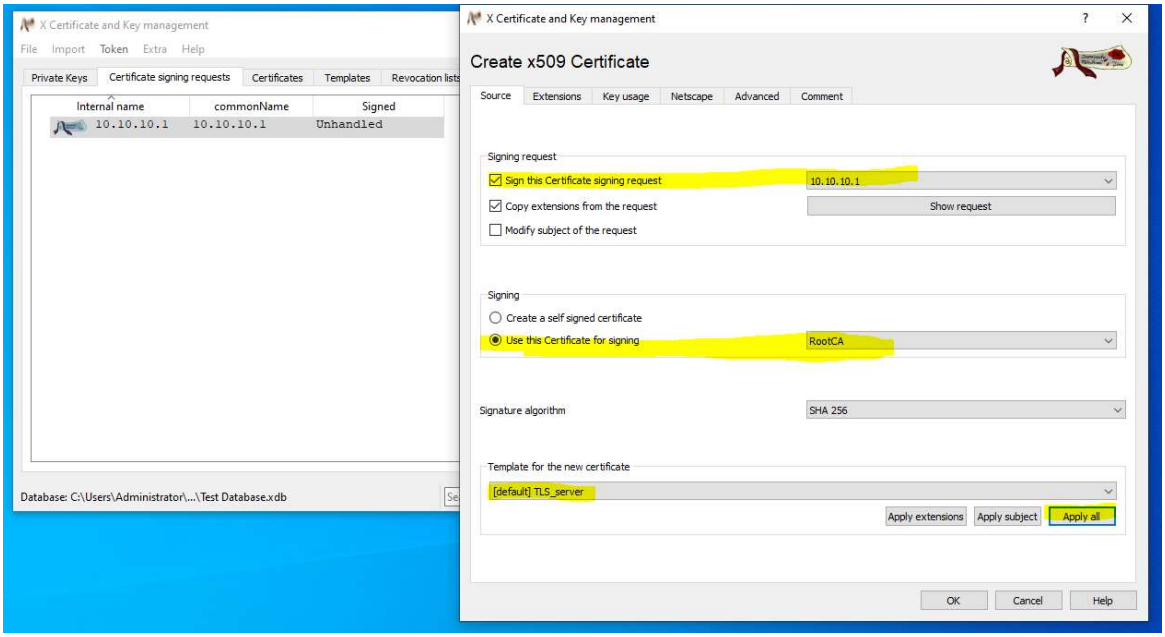


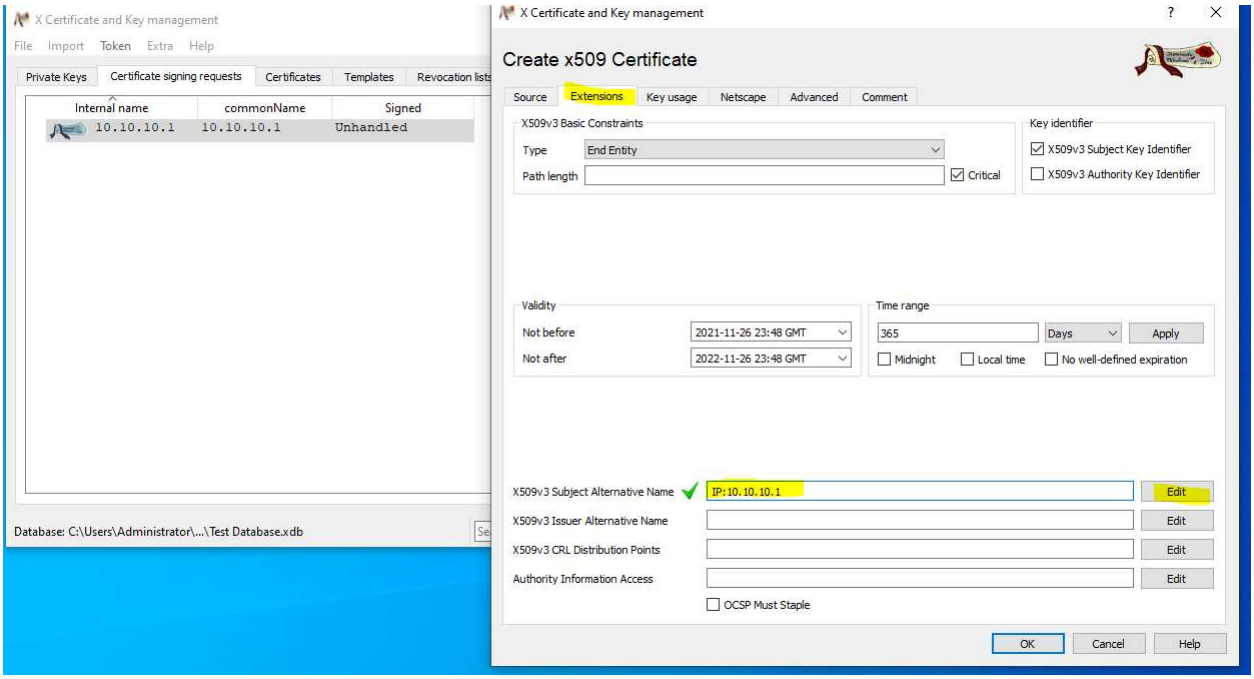

8. Now let's sign the FortiGate's CSR using XCA application (signing will be done by the RootCA's private key by default) and then import the signed certificate on the FortiGate. For that we have to import the FortiGate's CSR in the XCA application by clicking on Import. Once imported it will be shown up as Unsigned under Certificate Signing Request Page.



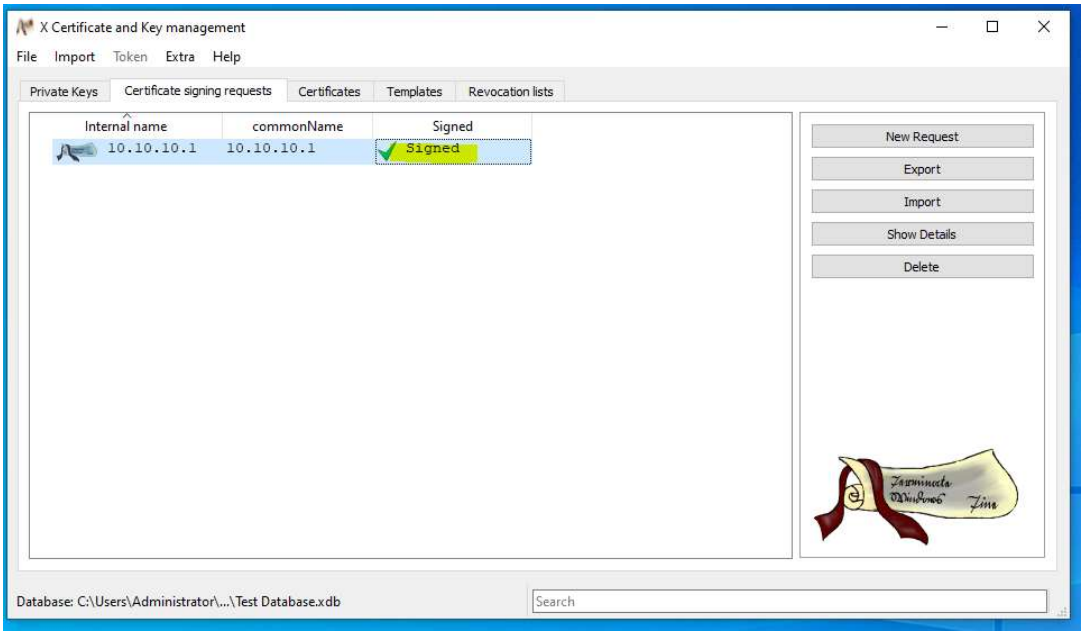


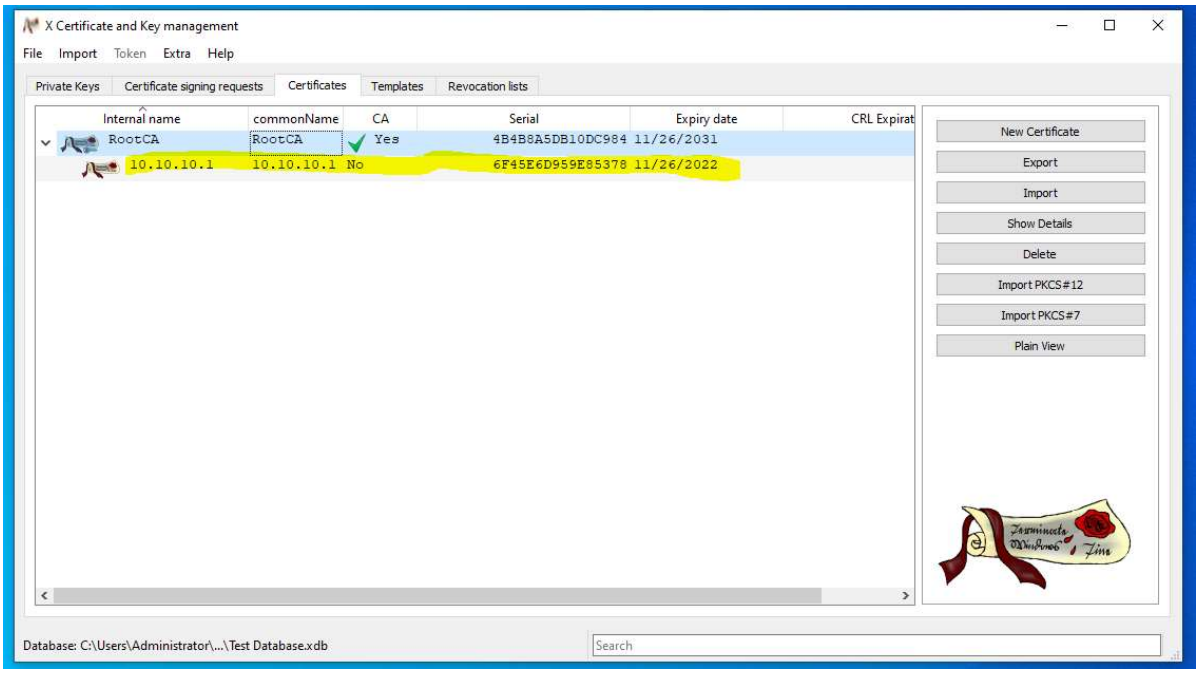

9. Right click on the Certificate and select Sign, a new Pop up will open confirming the details in the CSR once again. Select the Template as **"[default] TLS_Server**" as this is a certificate which will be used on the FortiGate to provide HTTPS service. Select "**Apply all**". Select the Certificate which you want to use to sign this certificate, in our case its our **"RootCA**" certificate. In the **"Extensions"** Tab, fill in the SAN name as 10.10.10.1, this field can be optional as well as our CN name is also 10.10.10.1 which is the IP which we will be using to access the Fortigate. Click on **"OK".**



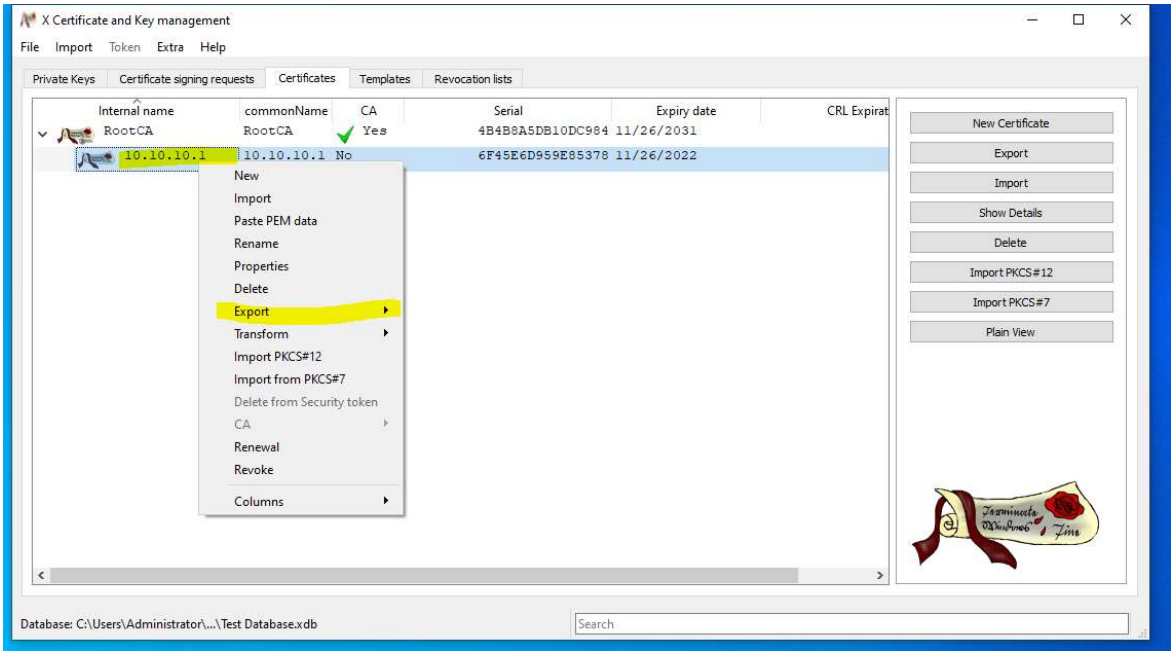



10. Now this certificate will be shown as Signed under the **"Certificate Signing Request**" tab and will also appear under the **"Certificates"** Tab under the RootCA as shown.

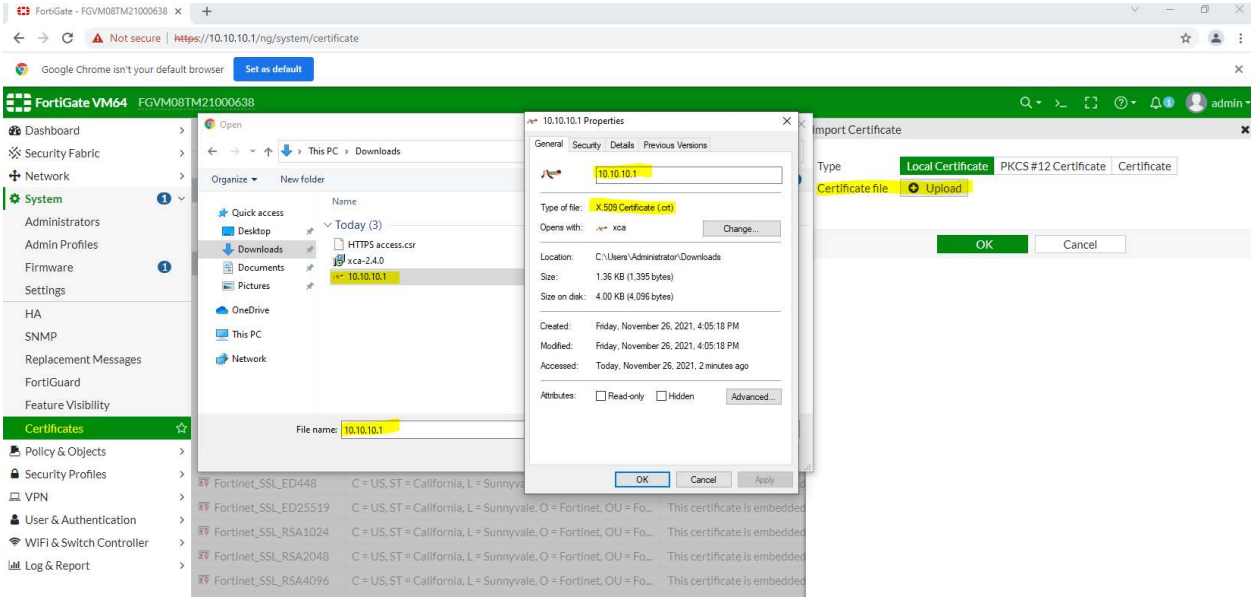




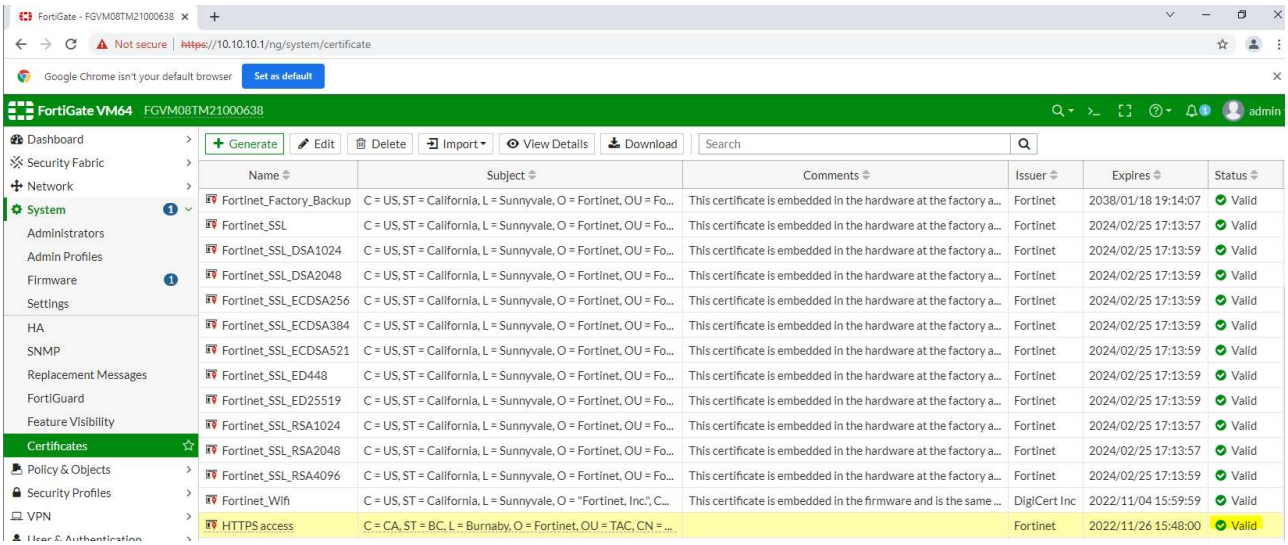

11. Now we will need to **"Export"** this Newly Signed Certificate and import it on the FortiGate. We need to export it in **.crt** format as shown. Save it in a suitable location on your PC.



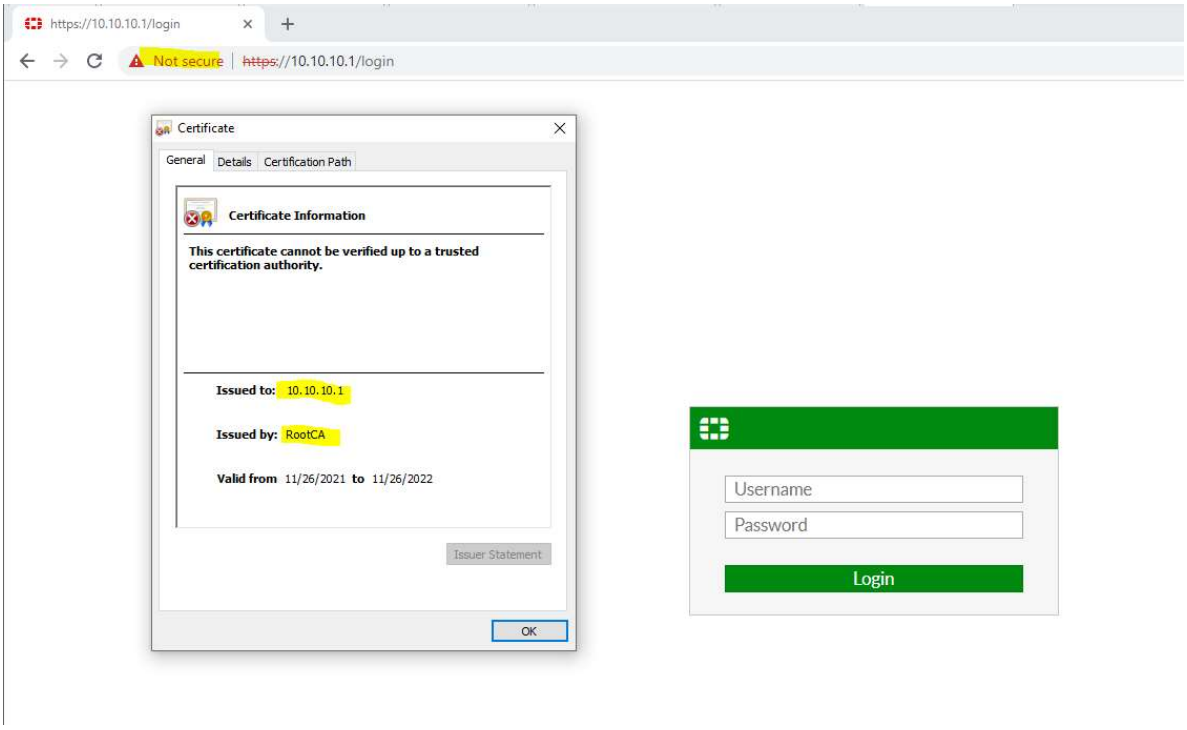
12. Now we need to import this Signed Certificate on the FortiGate by **System→Certificates→Import→Local Certificate→Upload the signed Certificate**



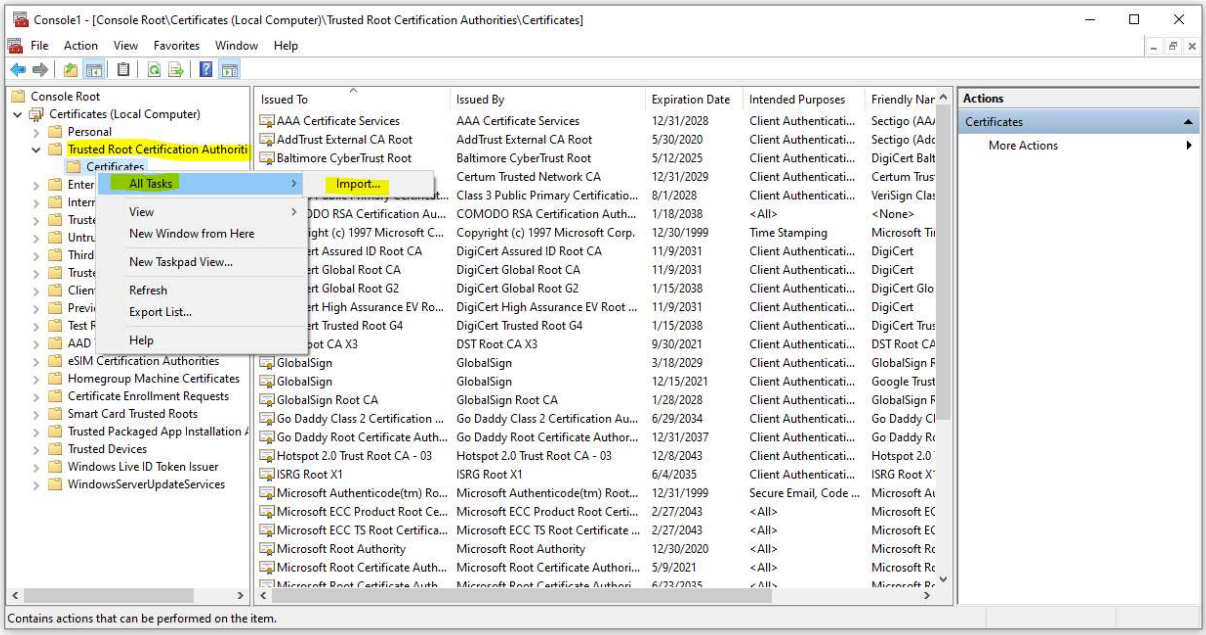

13. Once done, it will be shown up as **"Valid"** under **System→Certificates** as its signed by a Trusted CA now. We can start using this certificate for HTTPS by referencing it under **System→ Settings→HTTPS Server Certificate dropdown.**

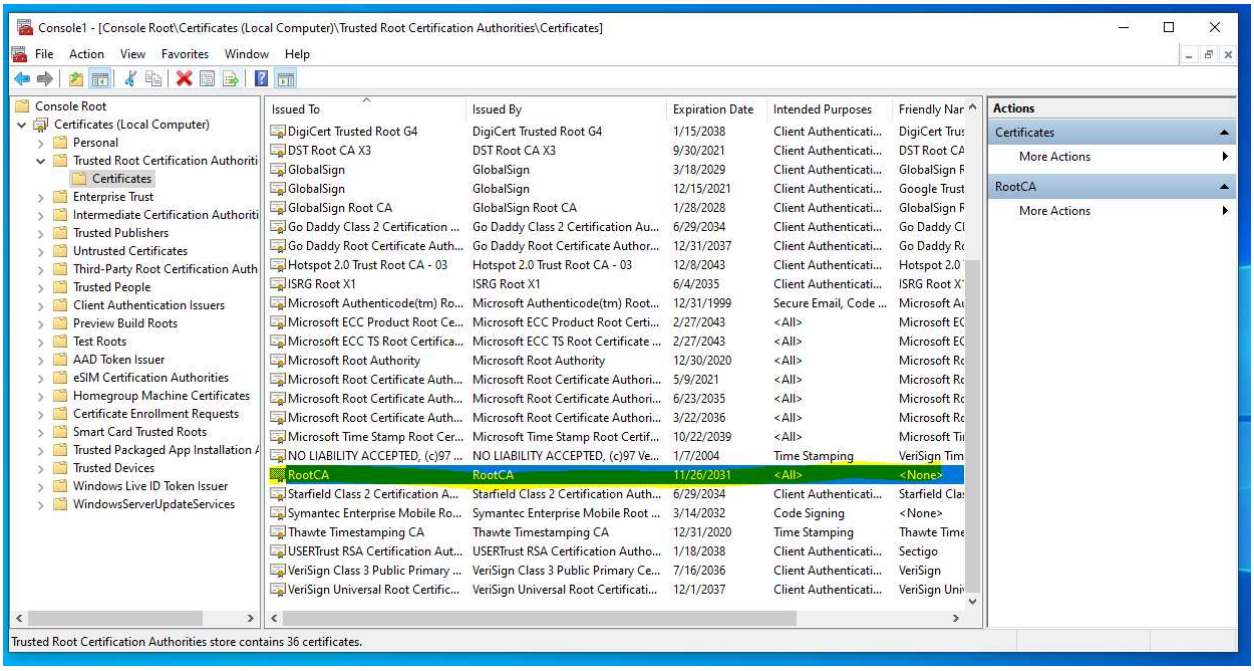



14. Now connect to the FortiGate on https://10.10.10.1, and now you can see the Signed Certificate and the Issued to and Issuer details.

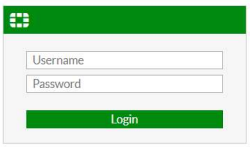


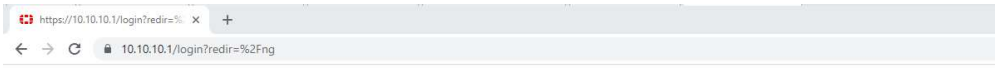

15. But when we access FortiGate on https://10.1.1.1 using Chrome Browser, it is still showing as Not Secure and Untrusted. This is because it's the Chrome Browser which by default does not trust the certificate which is issued by the **RootCA** Certificate Authority, as XCA is not a global and widely trusted Certificate Authority but our own Private CA. To add the Certificate to the Trusted List on Windows machine, open **Search→Run→mmc→File→Add and Remove Snap-ins→Certificates→Add→Computer account→Finish.** Now on the mmc console you can now manage the Windows Certificate Store. Import the **RootCA** certificate in the Trusted Root Certification Authority store. The **RootCA** will now show up in the Trusted Root Certificate Authority store.

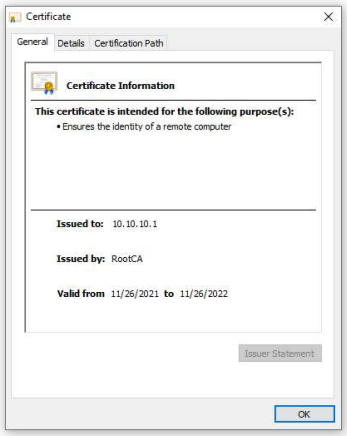


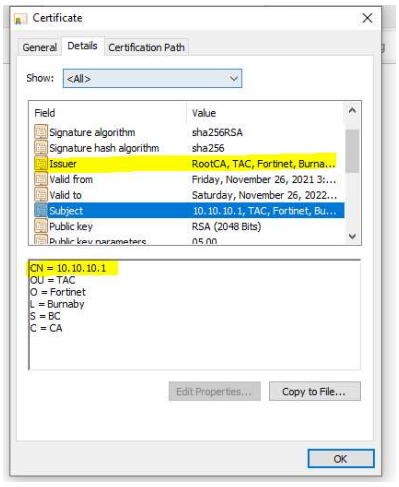

16. Close the mmc and access the FortiGate on https://10.10.10.1 again on a new Chrome browser window. Now there will not be any Certificate Errors as Chrome is trusting the CA who signed the certificate for the FortiGate. Also Chrome is not flagging any error as we are accessing FortiGate on 10.10.10.1 which is same as the CN name (Issued to). Chrome also does a variety of other checks such as checking Validity dates, CRL List, OSCP etc. which is out of scope of this document.






17. In a similar way we can manage to create and sign certificates using XCA for SSLVPN for FortiGate and for other certificate related applications.

18. Few important certificate extensions to remember.

**.csr → CSR file**

**.pkcs12→ File that contains both private and public key.**

**.crt→File that contains the signed certificate, signed by a CA.**

**.pem→ File that contains the private key**