Creating self signed certificates with makecert.exe for development

blog.jayway.com/2014/09/03/creating-self-signed-certificates-with-makecert-exe-for-development

Elizabeth Andrews September 3, 2014

If you've ever had the need of creating self signed certificates you may start out feeling like it's not a straightforward stroll in the park, so here is a blog post that might help you to get started. I will be going through the basics of creating self signed X.509 certificates (Root, server & client) using makecert.exe.
For the complete makecert.exe parameter reference click here.

I'm using a PC with Windows 8.1 Pro and Visual Studio Premium 2013.

Certificate Authority (CA)
Normally most companies would just buy their certificates from a trusted third party certificate authority such as GoDaddy or Verisign, but for development and testing, this might not be the first thing one wants to do. Instead you can create your own self signed certificates, starting with a root CA that can be used to sign other certificates. *(For example ssl certificates for servers and clients)*. When you do this, the certificates are not trusted by default. You must therefore add the root CA to your machine's Trusted Root Certification Authorities Store through the Microsoft Management Console.

NOTE: You can add these two parameters: -sr LocalMachine ^ and -ss Root ^ to the upcoming command batch file, if you want to install the certificate directly into the LocalMachine's Trusted Root Certification Authorities. **BE SURE** to run the Developer Command Prompt as administrator or it will fail. We will however go through how to do this manually so you get a more basic understanding.

The ^ symbol I add to the following cmd batch files means "escape the next line", this makes it more readable instead of one long command string.

Let's do all of this step by step:
Open an empty notepad document and copy and paste the following into notepad:

```
makecert.exe ^
-n "CN=CARoot" ^
-r ^
-pe ^
-a sha512 ^
-len 4096 ^
-cy authority ^
-sv CARoot.pvk ^
CARoot.cer

pvk2pfx.exe ^
-pvk CARoot.pvk ^
-spc CARoot.cer ^
-pfx CARoot.pfx ^
-po Test123
```

This may or may not look a bit frightening or incomprehensive at first, but let me walk you through what is going on here: First we create a certificate with makecert.exe, then we use pvk2pfx.exe to copy the public key and private key information from the .pvk and .cer into a .pfx (personal information exchange) file.

NOTE: Never share your root .pvk or .pfx files if you want to stay secure!
The .pvk file contains your private key for your .cer certificate and the .pfx file contains both the certificate .cer and the private key .pvk, which means that others can sign new certificates with your certificate without your consent. The only file you can share is the .cer file, which only contains the public key.

The makecert.exe parameters:

- -n "CN=CARoot" ➜ Subject's certificate name and must be formatted as the standard: "CN=Your CA Name Here"
  You can also add more than one in the -n parameter for example: "-n "CA=CARoot,O=My Organization,OU=Dev,C=Denmark" and so on. Reference:
  - CN = commonName (for example, "CN=My Root CA")
  - OU = organizationalUnitName (for example, "OU=Dev")
  - O = organizationName (for example, "O=Jayway")
  - L = localityName (for example, "L=San Francisco")
  - S = stateOrProvinceName (for example, "S=CA")
  - C = countryName (for example, "C=US")
- -r ➜ Indicates that this certificate is self signed
- -pe ➜ The generated private key is exportable and can be included in the certificate
- -a sha512 ➜ We declare which signature algorithm we will be using (**DO NOT** use the sha1 algoritm, it is no longer secure)
- -len 4096 ➜ The generated key length in bits
- -cy authority ➜ Specifies that this is a certificate authority
- -sv CARoot.pvk ➜ The subject's .pvk private key file
- CARoot.cer ➜ The certificate file

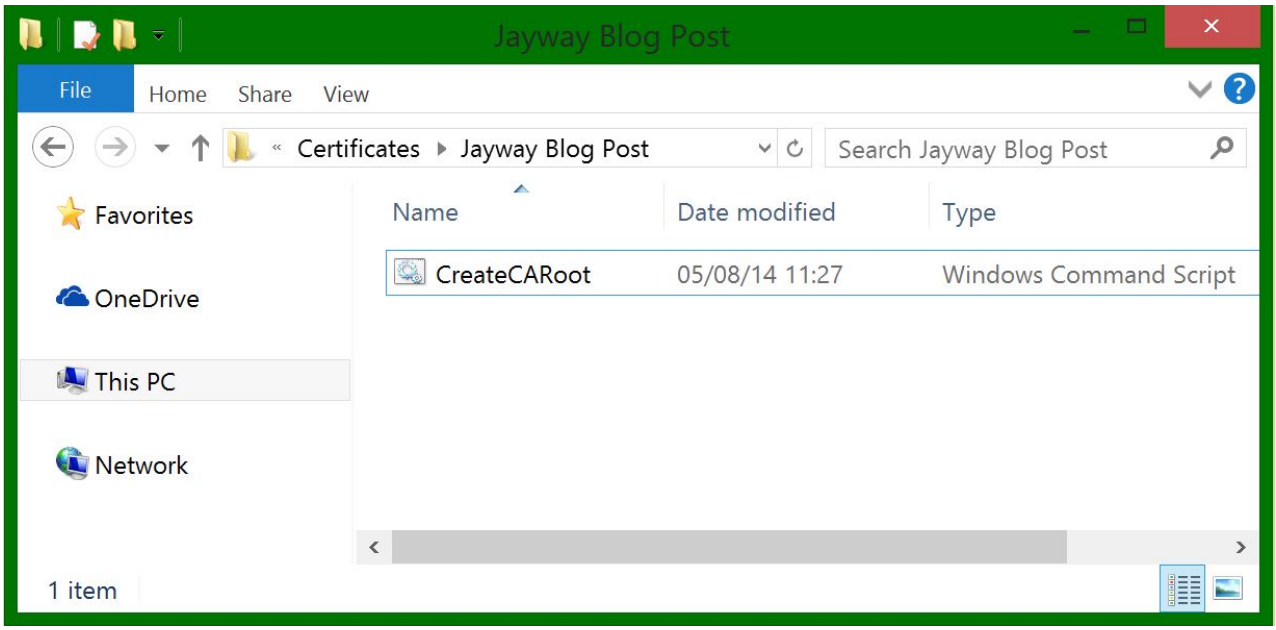
Optional: install certificate directly into the Trusted Root CA store

- -sr LocalMachine ➜ The subject's certificate store location
- -ss Root ➜ The certificate store name

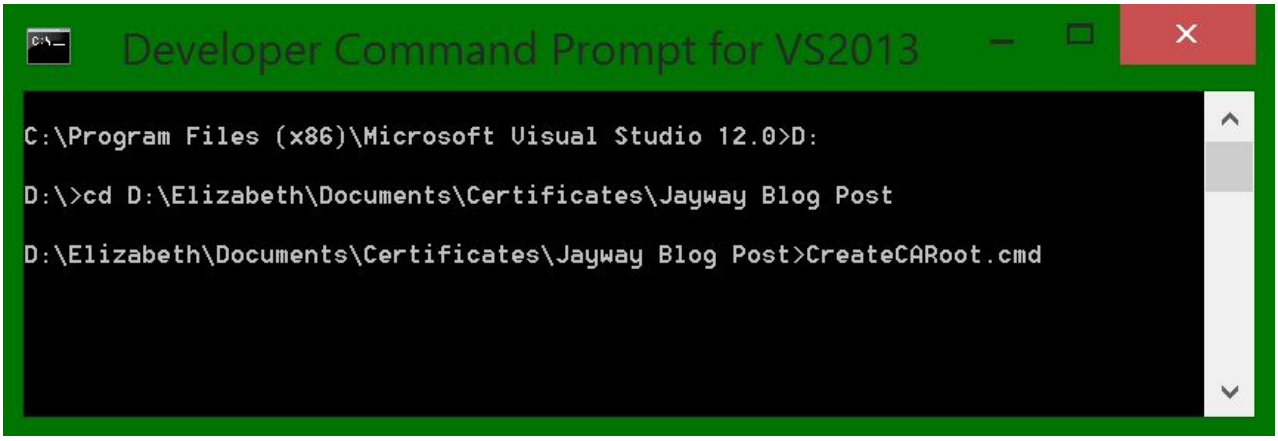
The pvk2pfx.exe parameters:

- -pvk CARoot.pvk ➜ The name of the .pvk file
- -spc CARoot.cer ➜ The name of the .cer file
- -pfx CARoot.pfx ➜ The name of the -pfx file
- -po Test123 ➜ The password for the .pfx file

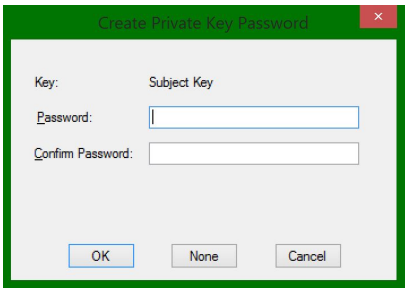
Save the document as "CreateCARoot.cmd" which will create a command batch file. *(You can call it what you want as long as you remember the .cmd ending which will make it a Windows Command Script)*

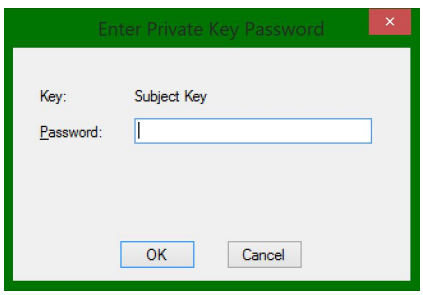


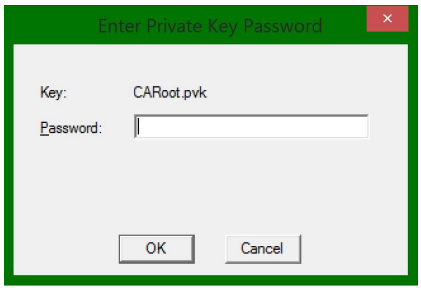
Open a Visual Studio Developer Command Prompt – this is where makecert.exe lives, and navigate to the folder that contains the batch file and run the cmd file



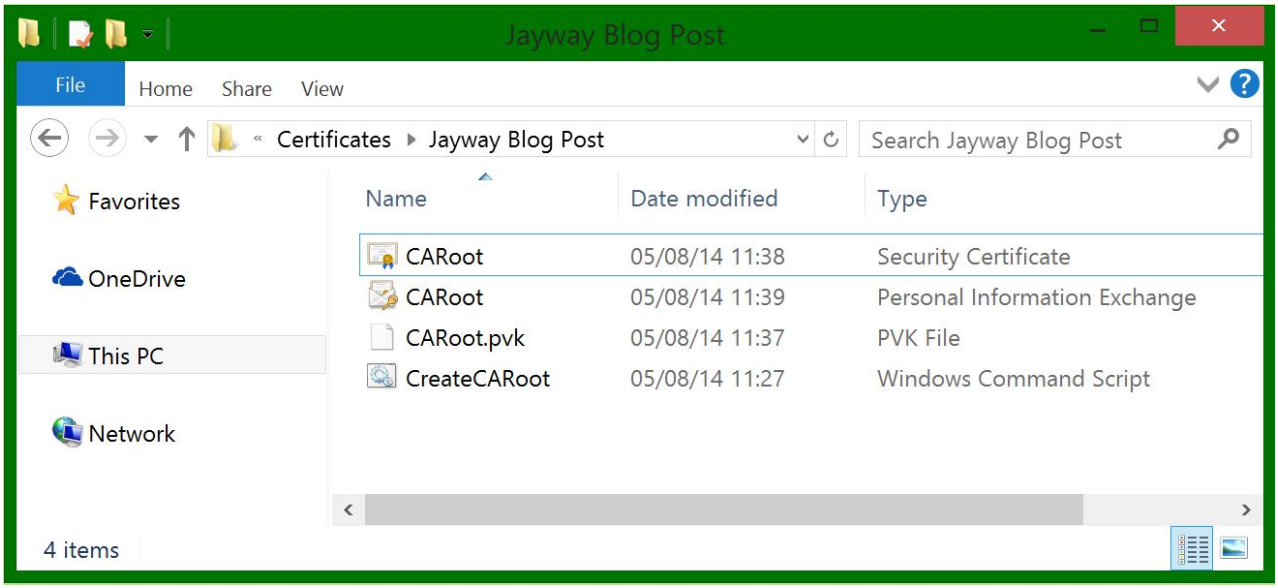
It should now prompt you to enter some passwords. *(This is where we create and use the .pvk private key, so these need to match for success)*







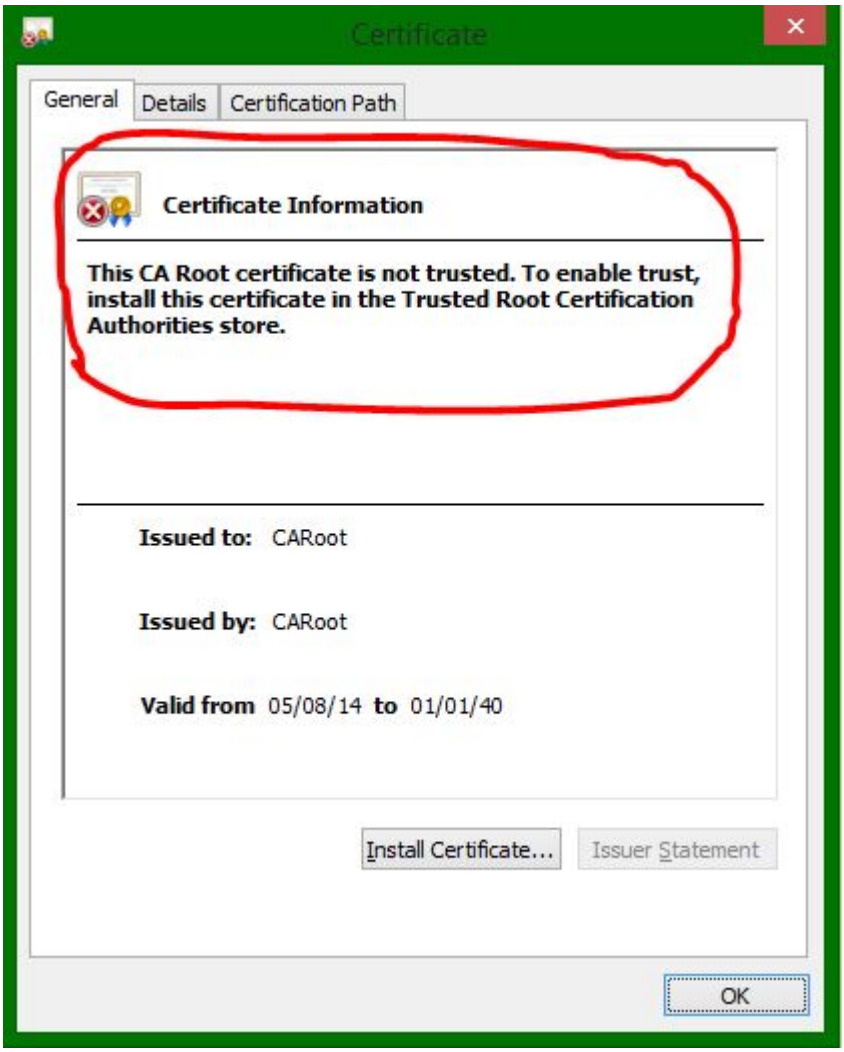
You should now have 3 new files: CARoot.cer, CARoot.pfx and CARoot.pvk in the folder where your batch files are.



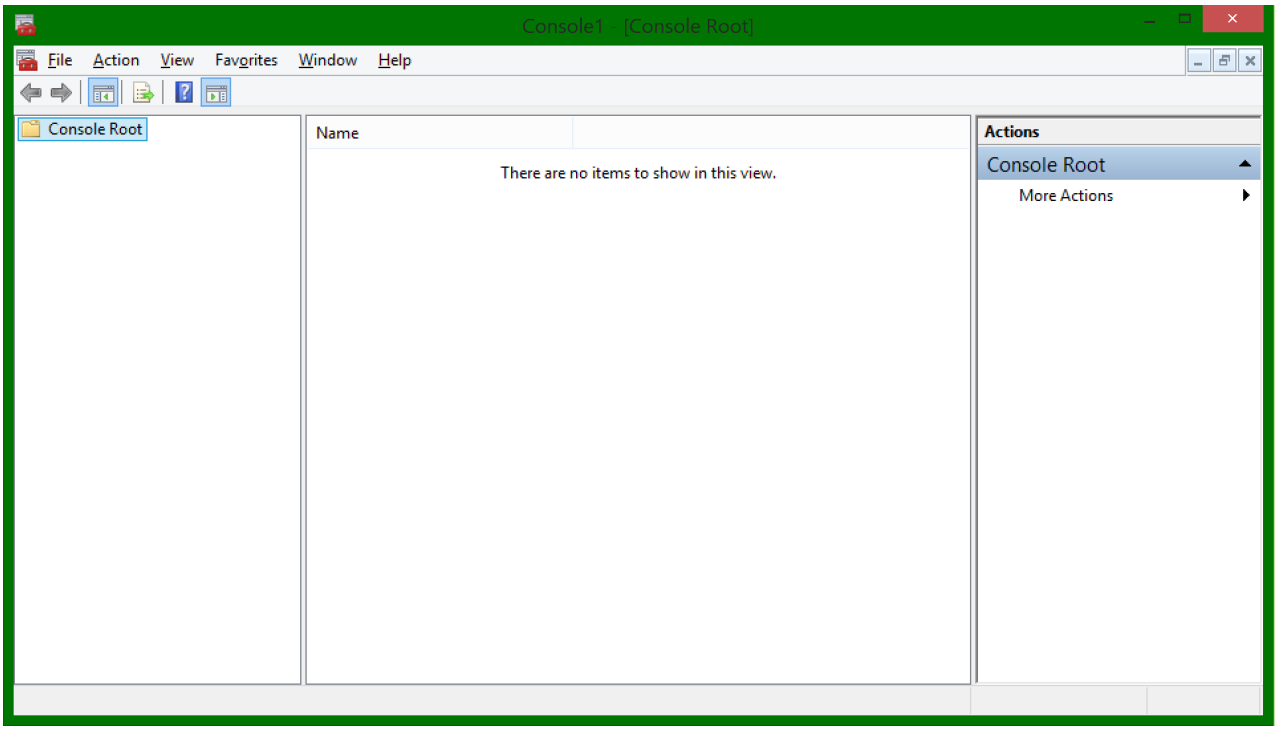
Making It Trusted

(This is a manual walk through if you didn't include the -sr and -ss parameters)

Open your new CARoot.cer file by double clicking it and see that it is not trusted.

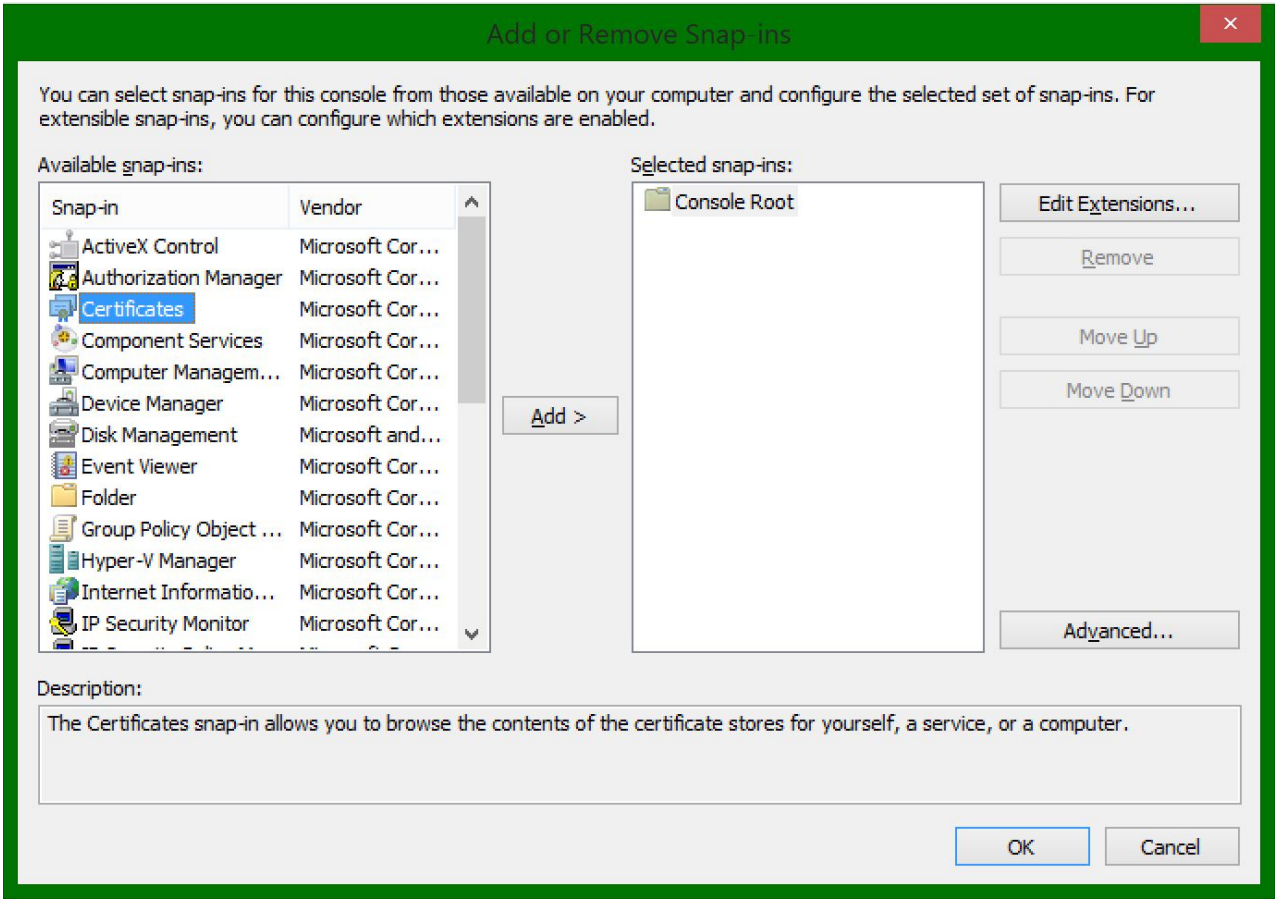


To make it trusted on your machine open up the Microsoft Management Console. *(Find it by searching for mmc in start)*

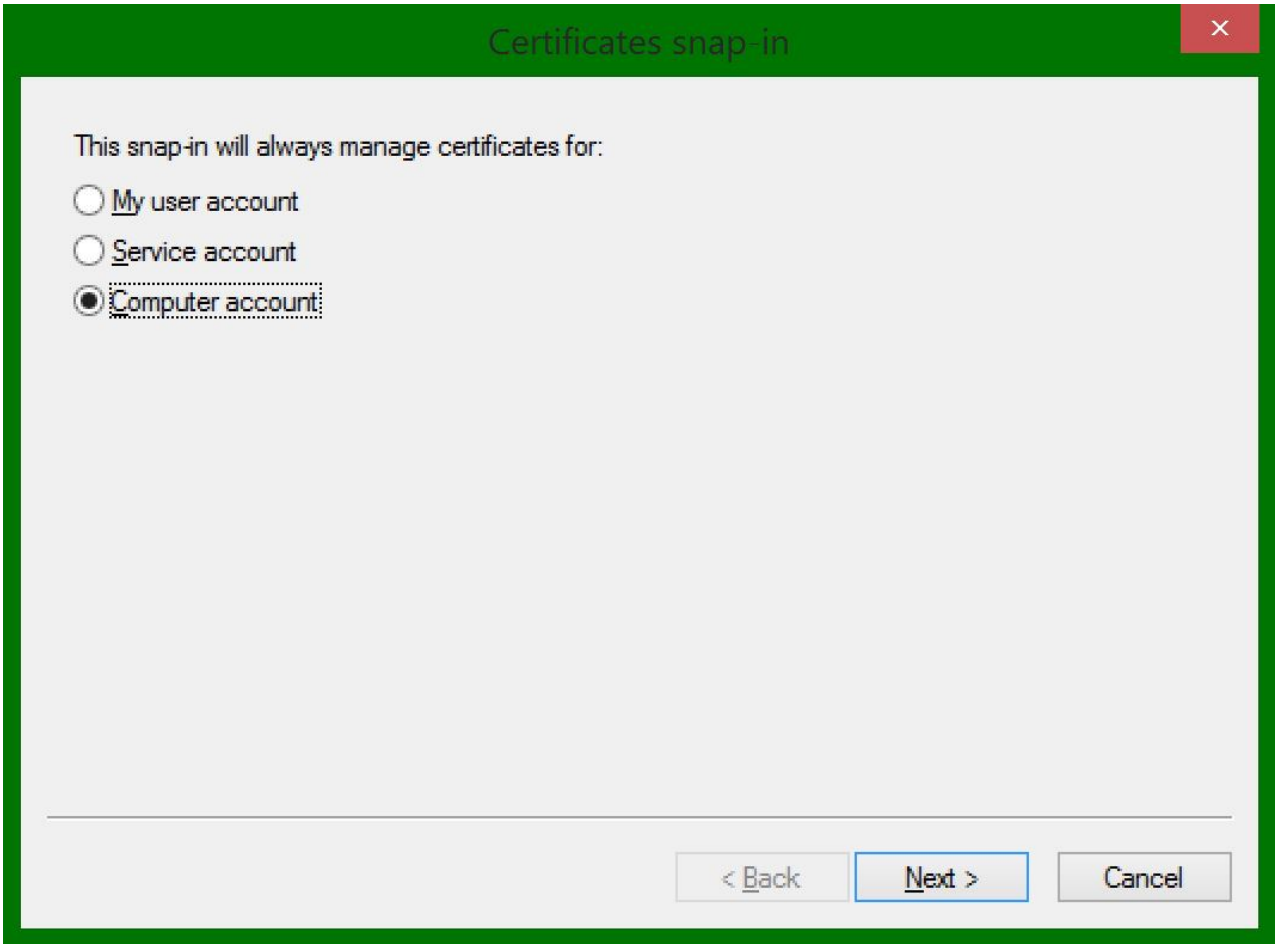


Go to File ➜ Add/Remove Snap-in

Double-click Certificates in the list to the left



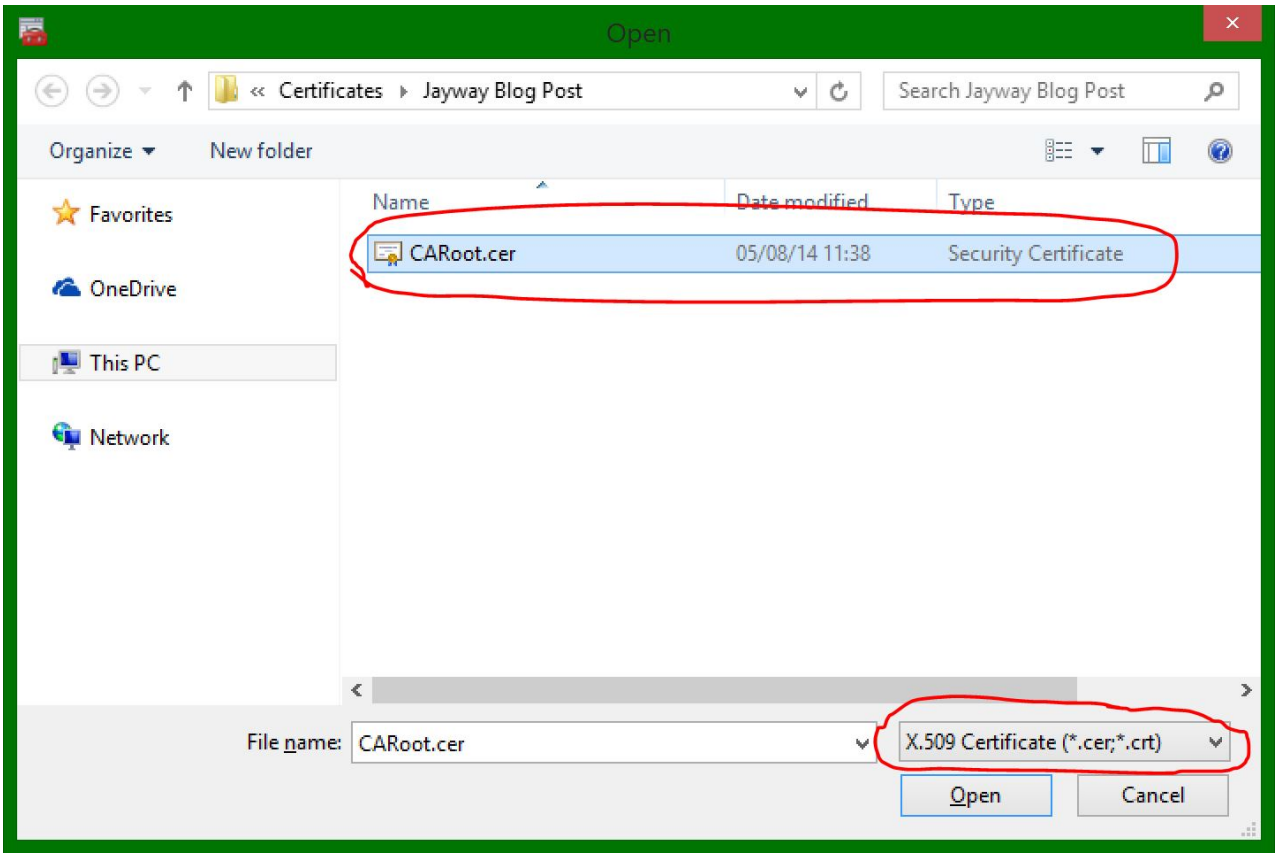
Choose Computer account and just go next, finish and OK



Open the Trusted Root Certification Authorities ➜ Certificates
Here you can see all of the currently trusted certificates that Windows trusts.
(Alot of them ship with Windows out of the box)

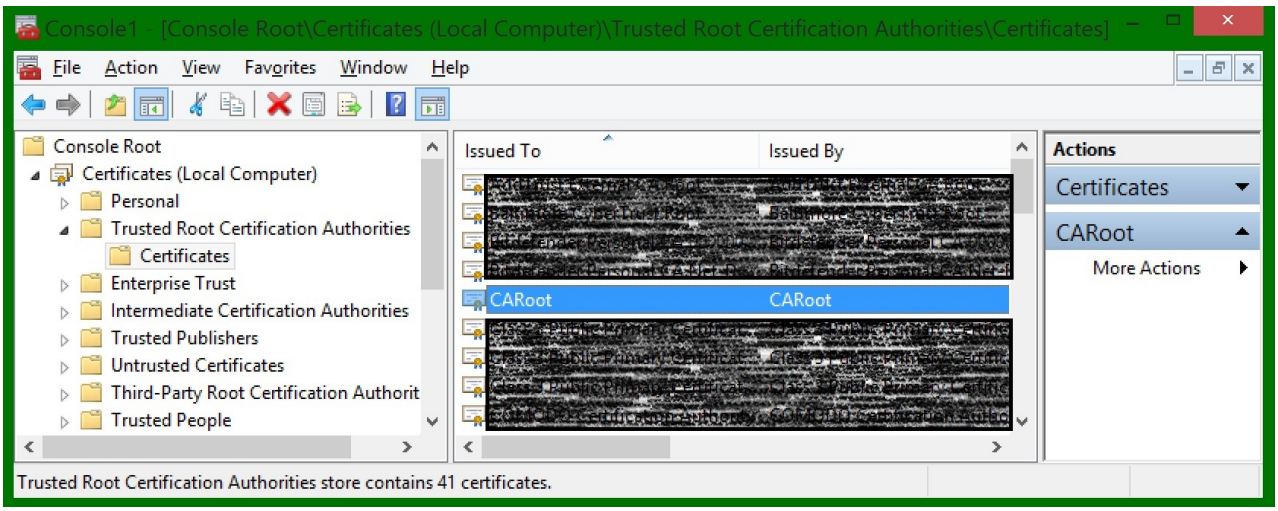
Now right-click the Certificates folder ➜ All tasks ➜ Import…

The certificate Import Wizard will pop up.
Go next ➜ Browse to find the CARoot.cer file we created earlier

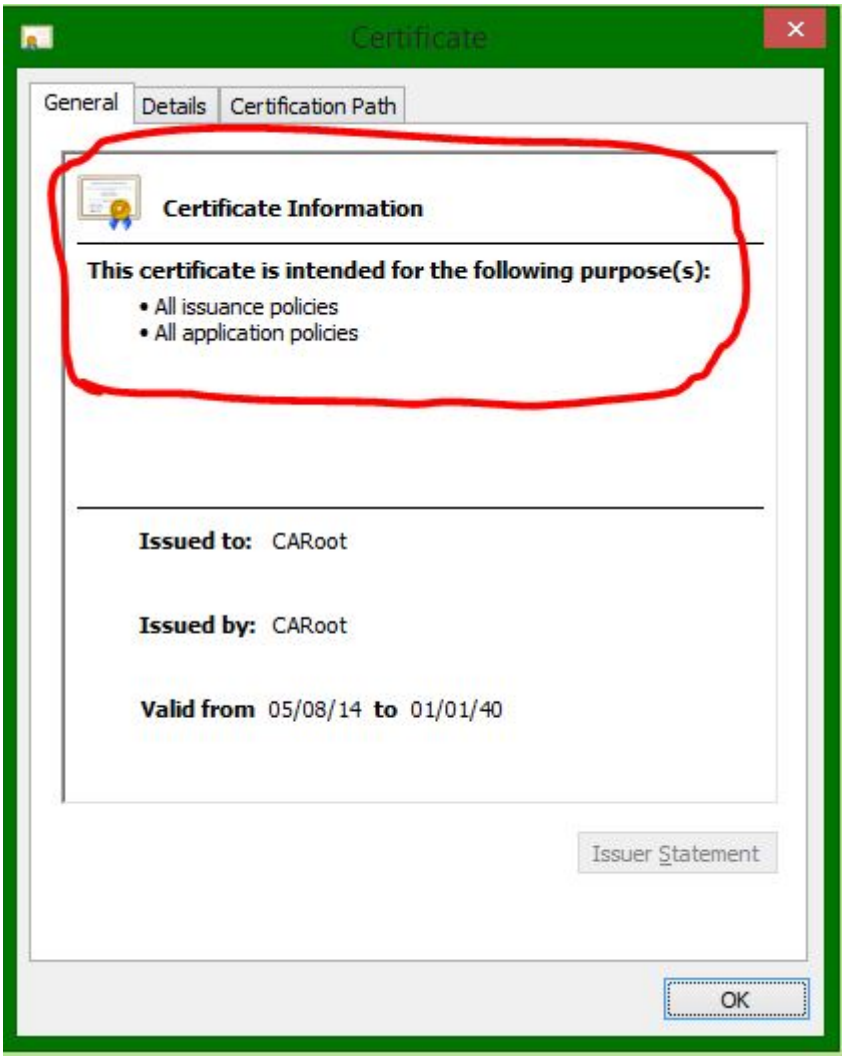


Keep going next until finish where a message box should appear saying “The import was successful”.

Your CARoot certificate should now be in you Trusted Root Certification Authorities store.



Open the CARoot (double-click) and see that it is now trusted by your computer.



Server Certificates

Next up we need a certificate to handle SSL on the server. We will create this with a new command batch file in notepad just like before, this time with these parameters:

```
makecert.exe ^
-n "CN=yourdomain.com" ^
-iv CARoot.pvk ^
-ic CARoot.cer ^
-pe ^
-a sha512 ^
-len 4096 ^
-b 01/01/2014 ^
-e 01/01/2016 ^
-sky exchange ^
-eku 1.3.6.1.5.5.7.3.1 ^
-sv %1.pvk ^
%1.cer

pvk2pfx.exe ^
-pvk %1.pvk ^
-spc %1.cer ^
-pfx %1.pfx ^
-po Test123
```

NOTE: The CN must match your domain otherwise the browsers won't trust your SSL certificate and warn the end user not to proceed to your website

You will recognize most of the parameters, but let me explain the new ones:

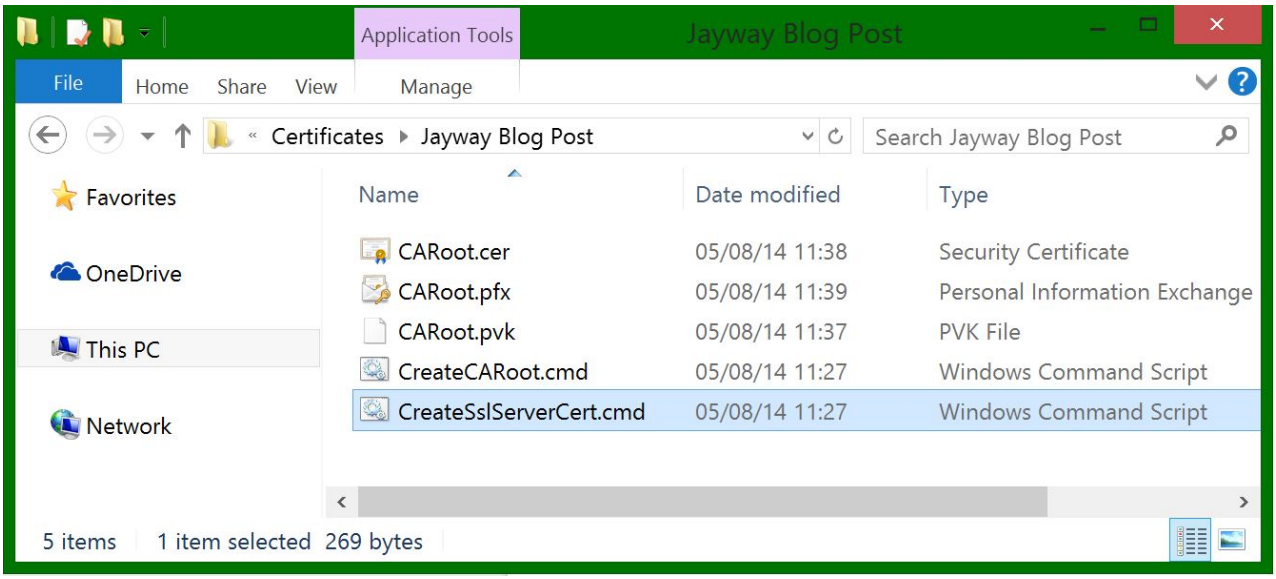
- -n "CN=yourdomain.com" for example ➜ Change this to your domain name in order to connect the SSL server certificate to a specific web server domain. (Examples: "CN=www.yourdomain.com", "CN=yourdomain.com" or the wildcard that will match all urls ending in your domain "CN=*.yourdomain.com".)
  You can also add more than one in the -n parameter for example: "-n "CA=CARoot,O=My Organization,OU=Dev,C=Denmark" and so on. Reference:
  - CN = commonName (for example, "CN=My Root CA")
  - OU = organizationalUnitName (for example, "OU=Dev")
  - O = organizationName (for example, "O=Jayway")
  - L = localityName (for example, "L=San Francisco")
  - S = stateOrProvinceName (for example, "S=CA")
  - C = countryName (for example, "C=US")
- %1 ➜ A command line parameter and will be whatever you type in after .cmd, this will be the file name of your .cer, .pvk and .pfx files
- -iv CARoot.pvk ➜ Issuer's (The CA that signed it) .pvk private key file
- -ic CARoot.cer ➜ The issuer's certificate file
- -b 01/01/2014 ➜ Start of the period where the certificate is valid
- -e 01/01/2016 ➜ End of the valid period
- -sky exchange ➜ Indicates that the key is for key encryption and key exchange
- -eku 1.3.6.1.5.5.7.3.1 ➜ Server authentication OID (Object Identifier). Identifies that this is an SSL Server certificate.

Optional: Install server certificate directly into the LocalMachine Personal certificate store

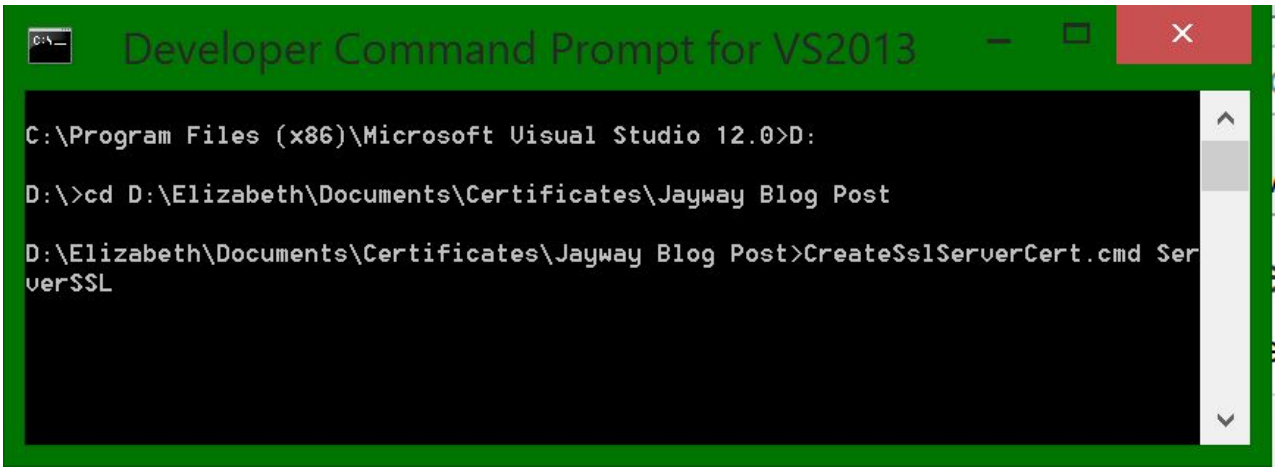
NOTE: This will only install the .cer file into the MMC, in order to import the .pfx file you will have to do it manually.

- -sr LocalMachine ➜ The subject's certificate store location
- -ss My ➜ The certificate store name that will store the output certificate

This will create a SSL certificate to use on your server and will be signed by your CARoot authority.

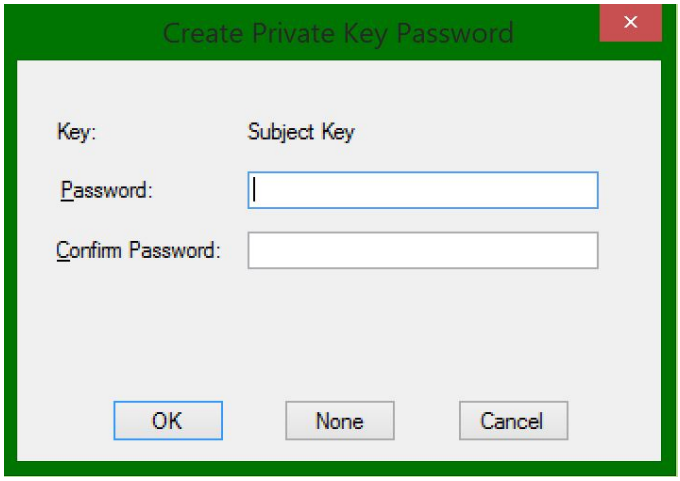


Run it in your Developer Command Prompt the same way as before, only this time type in a name for your certificate after the command. Mine will be: CreateSslServerCert.cmd ServerSSL

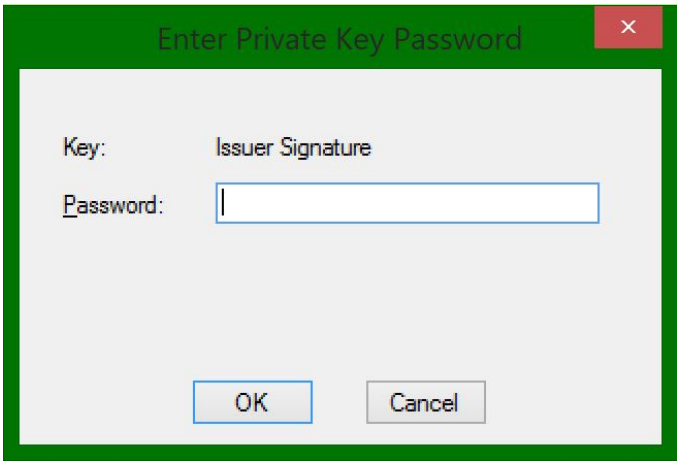


Again it will ask you to create your private key password, use it to verify, also give the issuers password *(which is the one you chose when creating your root CA)* and lastly the private key password you choose in the first window.

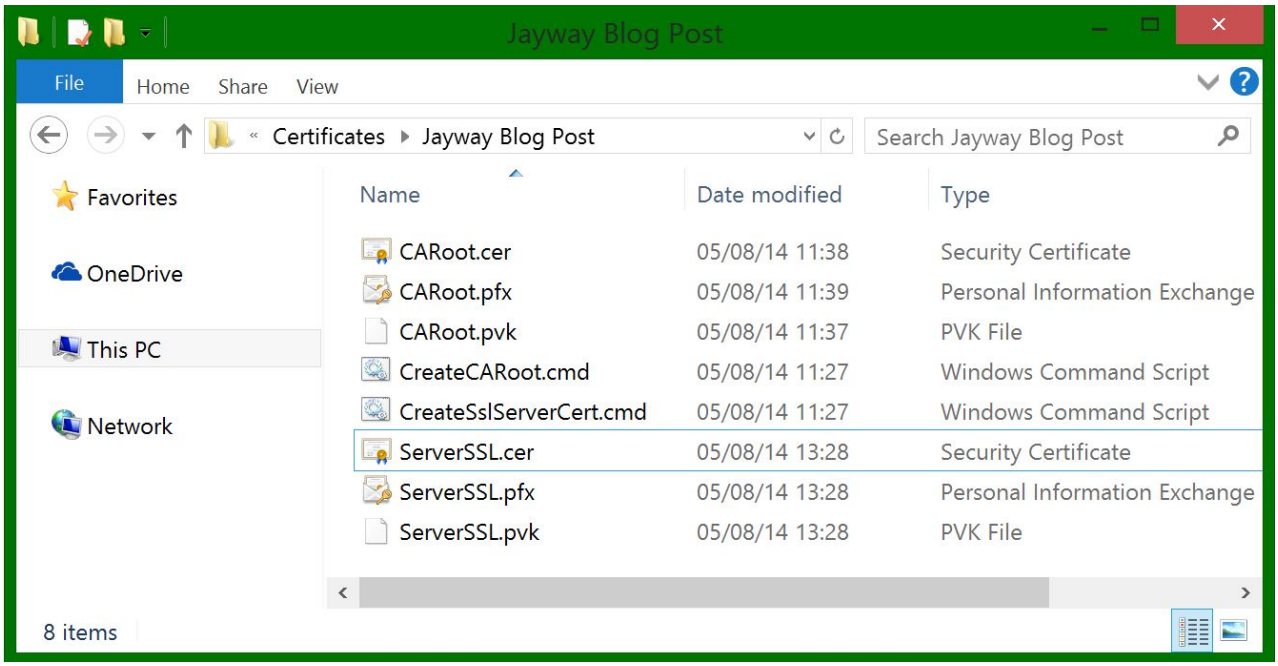
...aaand voila you now have the ServerSSL certificate files.




Enter Private Key Password
Key:
Subject Key
Password:
OK
Cancel


Enter Private Key Password
Key:
Issuer Signature
Password:
OK
Cancel

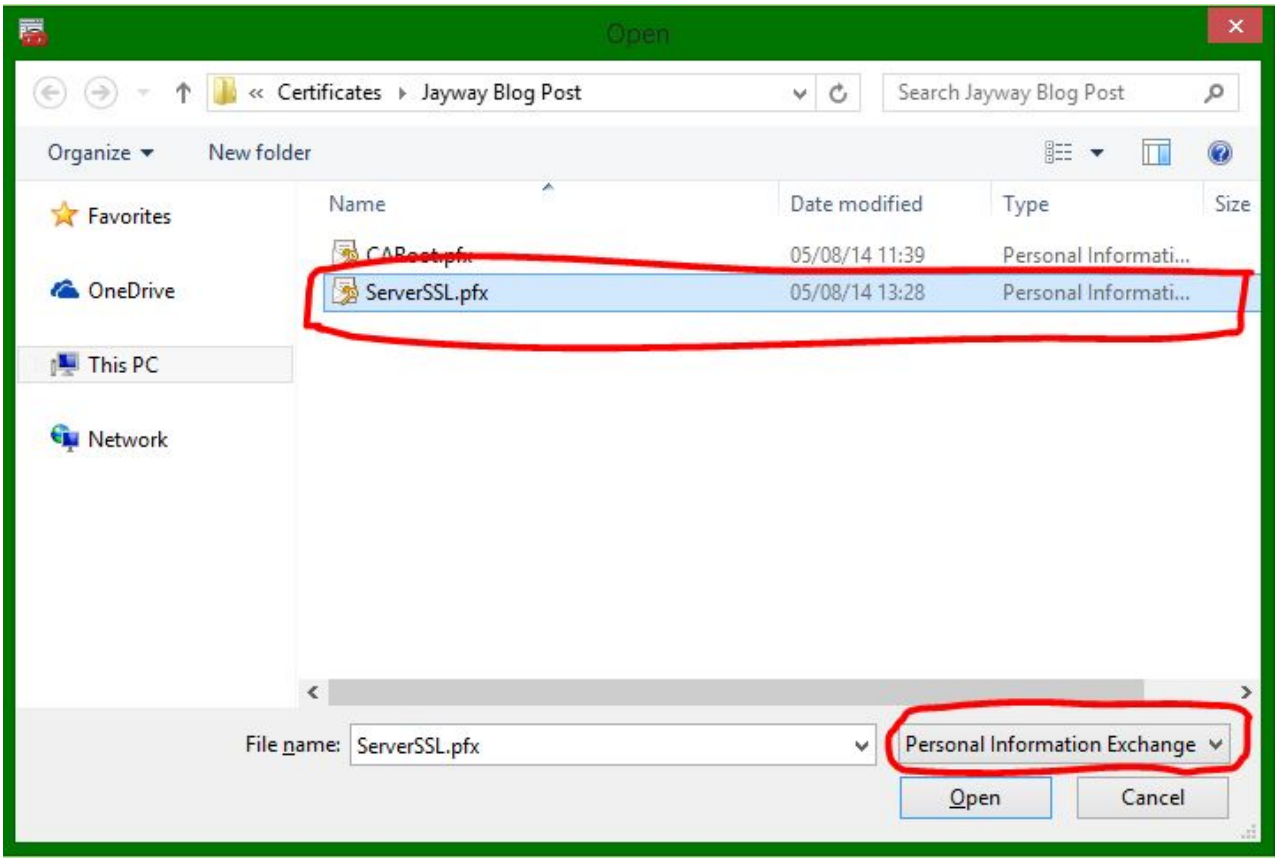

Enter Private Key Password
Key:
ServerSSL.pvk
Password:
OK
Cancel



If you didn't include the -sr and -ss parameters, import the Personal Information Exchange (pfx) certificate into your Personal Certificates in the Microsoft Management Console:
Open the Personal folder ➜ right-click Certificates ➜ Import…

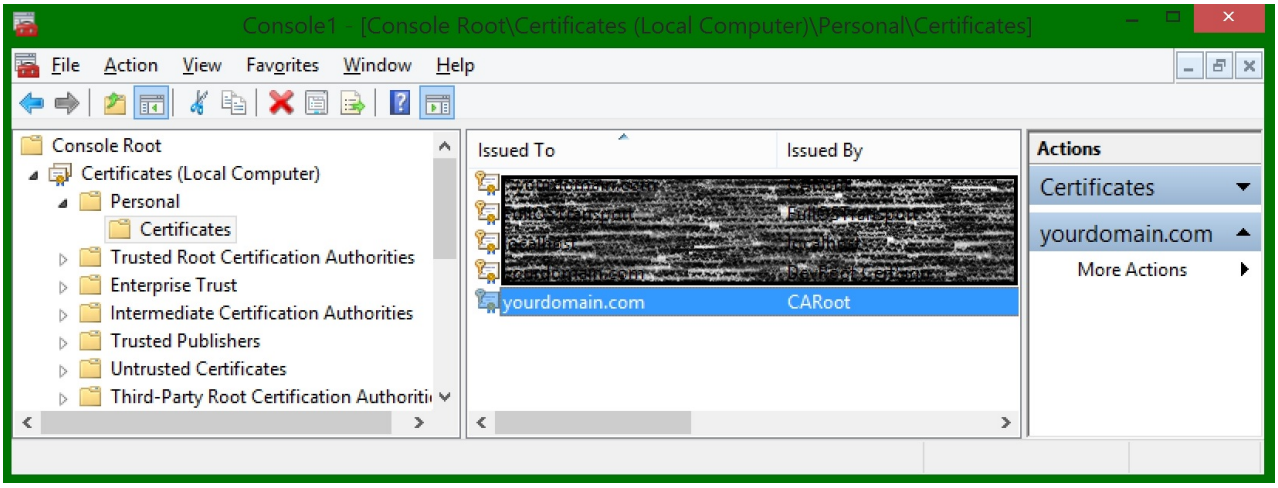
Again the Certificate Import Wizard pops up ➜ Go Next

This time you will Browse for the ServerSSL.pfx file



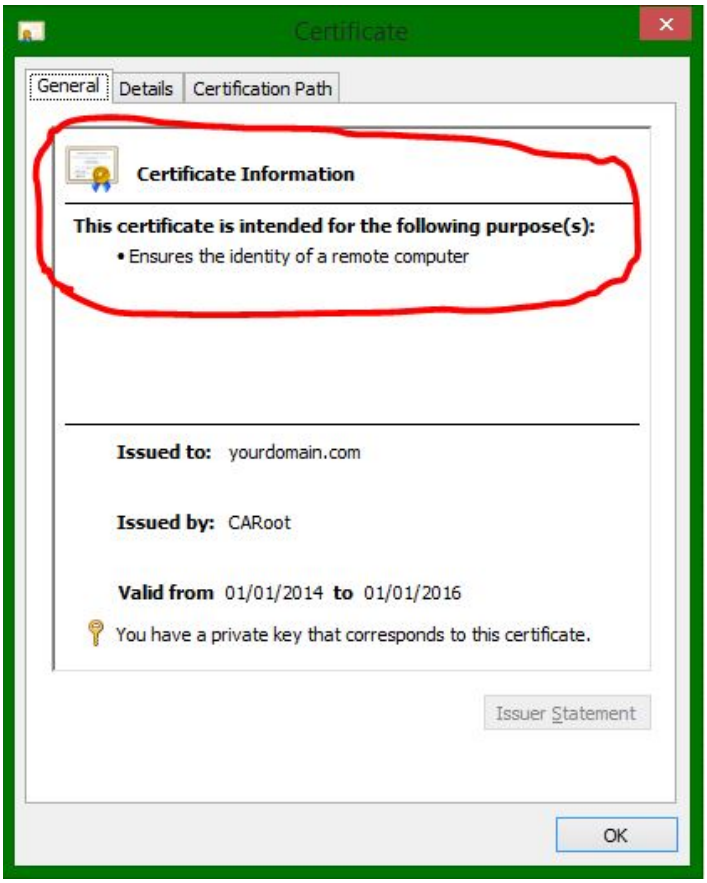
Go next ➜ Type in the password for your pfx file *(The -po parameter from the batch file)* ➜ Continue going next until finish and the message box with ”The import was successful” appears.

You should now see you newly imported certificate in your ➜ Personal Certificates folder



It is trusted automatically because your CARoot that signed it is trusted and has a private key corresponding to this certificate.

You can now configure your server to use this certificate.



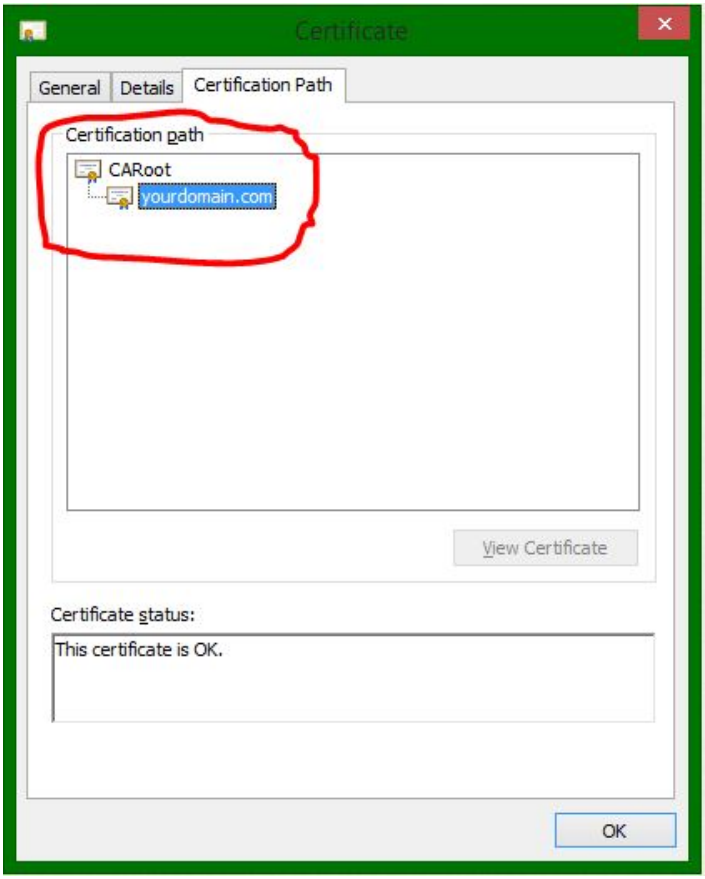
Client Certificates

Last but not least we will create the client certificate which can be used for client certificate authentication. We will again create a command batch file, now with the following parameters:

```
makecert.exe ^
-n "CN=%1" ^
-iv CARoot.pvk ^
-ic CARoot.cer ^
-pe ^
-a sha512 ^
-len 4096 ^
-b 01/01/2014 ^
-e 01/01/2016 ^
-sky exchange ^
-eku 1.3.6.1.5.5.7.3.2 ^
-sv %1.pvk ^
%1.cer

pvk2pfx.exe ^
-pvk %1.pvk ^
-spc %1.cer ^
-pfx %1.pfx ^
-po Test123
```

You may notice that this is almost identical to the server certificate parameters, all except:

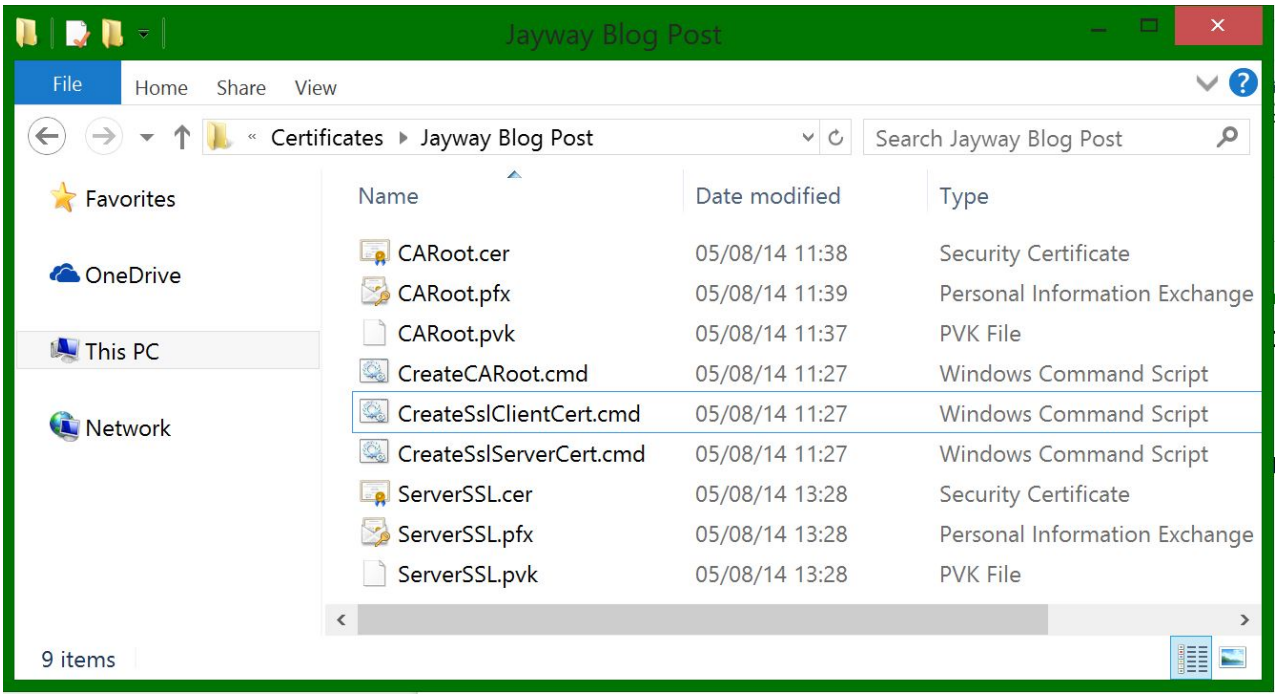


- "CN=%1" ➜ This can be whichever name you like and will be what you type in after .cmd
  You can also add more than one in the -n parameter for example: "-n "CA=%1,O=My Organization,OU=Dev,C=Denmark" and so on. Reference:
  - CN = commonName (for example, "CN=My Root CA")
  - OU = organizationalUnitName (for example, "OU=Dev")
  - O = organizationName (for example, "O=Jayway")
  - L = localityName (for example, "L=San Francisco")
  - S = stateOrProvinceName (for example, "S=CA")
  - C = countryName (for example, "C=US")
- -eku 1.3.6.1.5.5.7.3.2 ➜ The client authentication OID (Object Identifier).

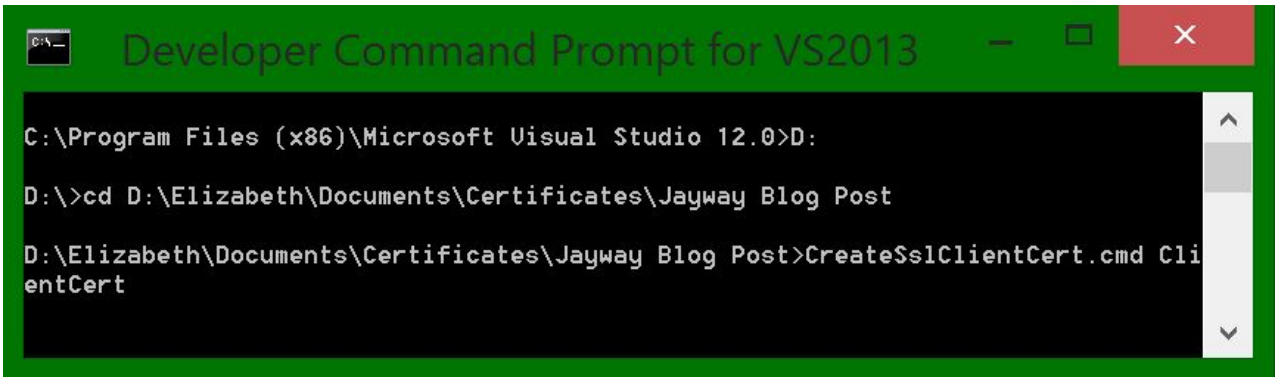
Optional: install client certificate directly into the CurrentUser Personal certificate store
NOTE: This will only install the .cer file into the MMC, in order to import the .pfx file you will have to do it manually.

- -sr CurrentUser ➜ The subject's certificate store location
- -ss My ➜ The certificate store name

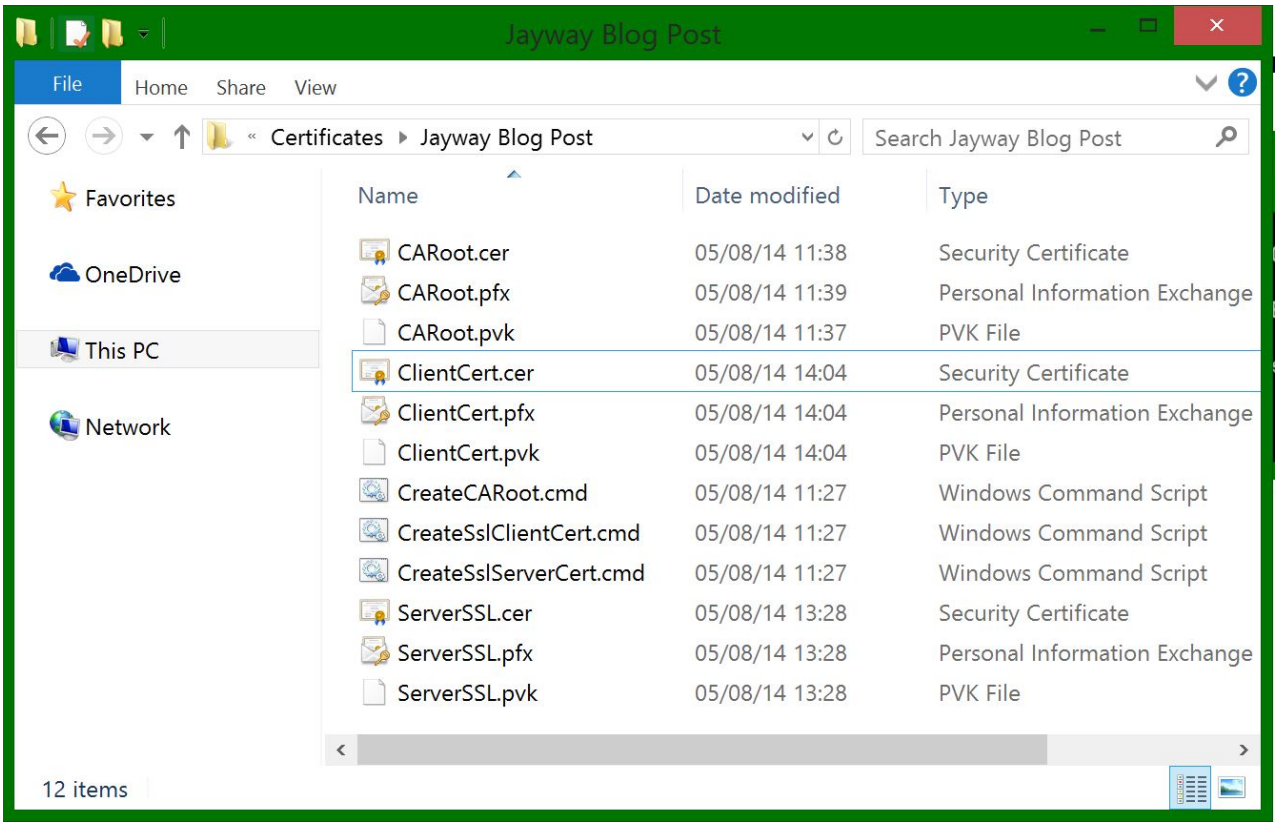
Your batch command will create a SSL certificate to use on your client and will be signed by your CARoot authority.



Execute the command batch file in the Developer Command Prompt, again with a name after the cmd. (Mine will be: CreateSslClientCert.cmd ClientCert)

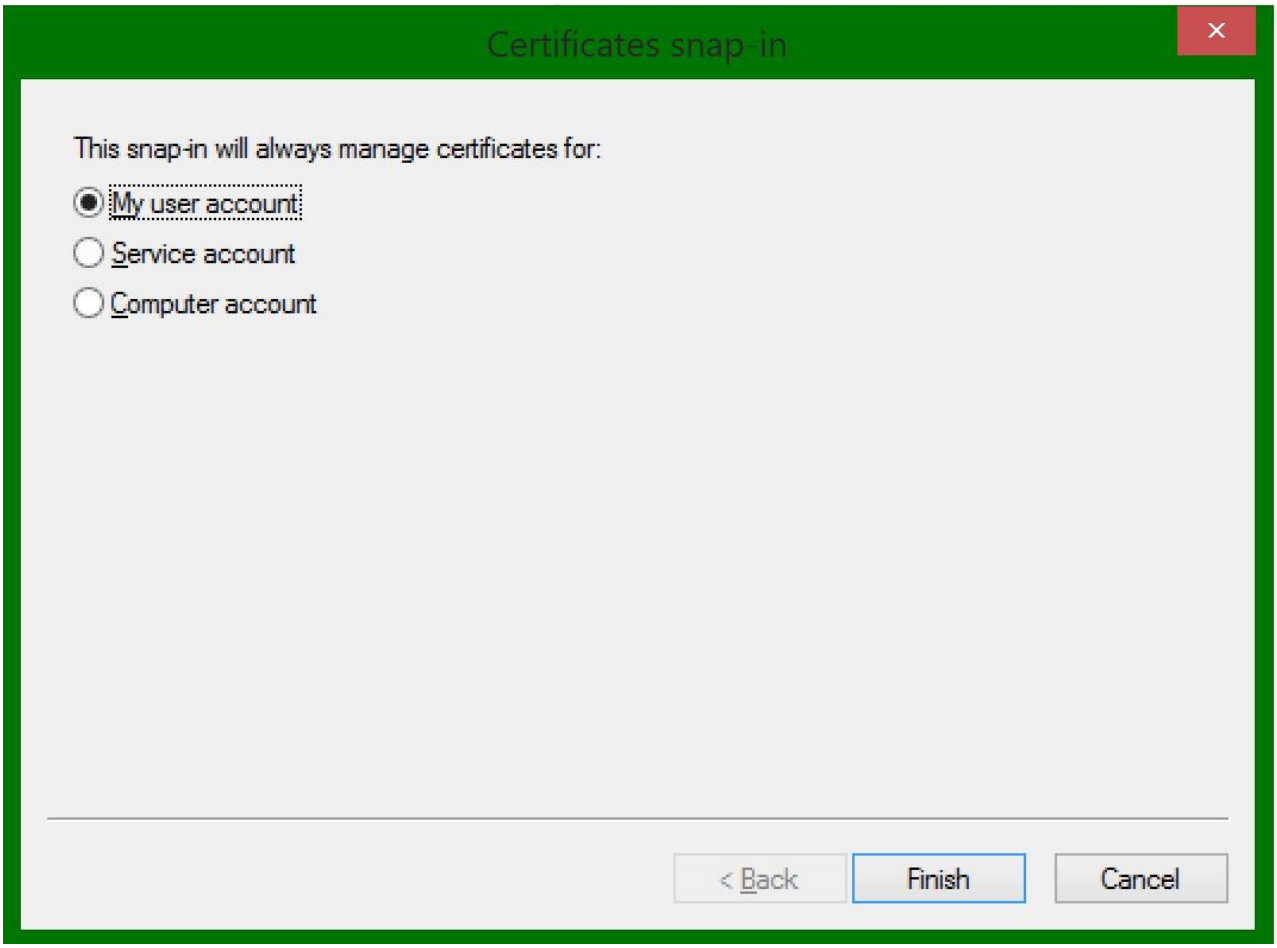


Enter the passwords in the same pattern as the server certificate and you now have your client certificate.



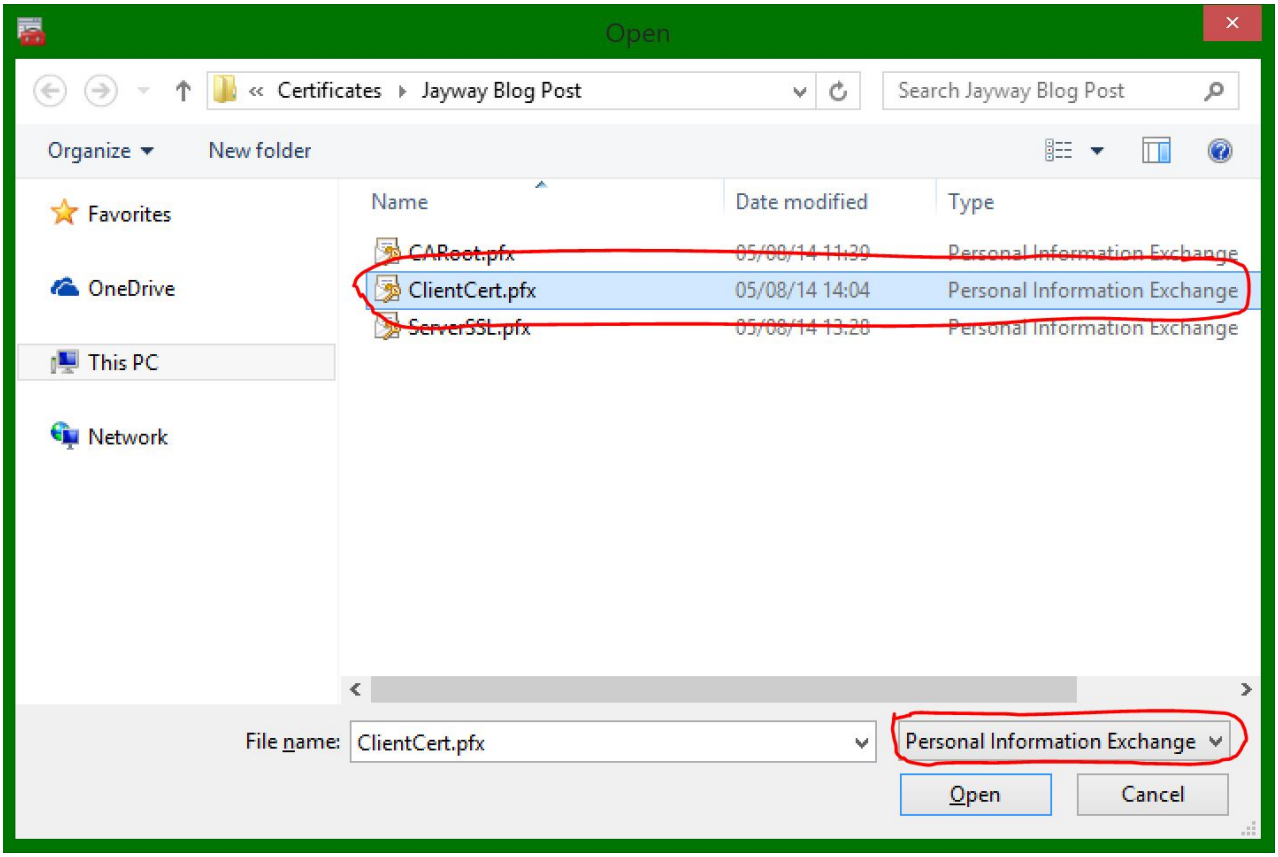
You can now add it to your Current User Personal Certificate store:
In the Microsoft Management Console, click File ➜ Add/Remove Snap-in

Double-click Certificates again, but this time choose My user account



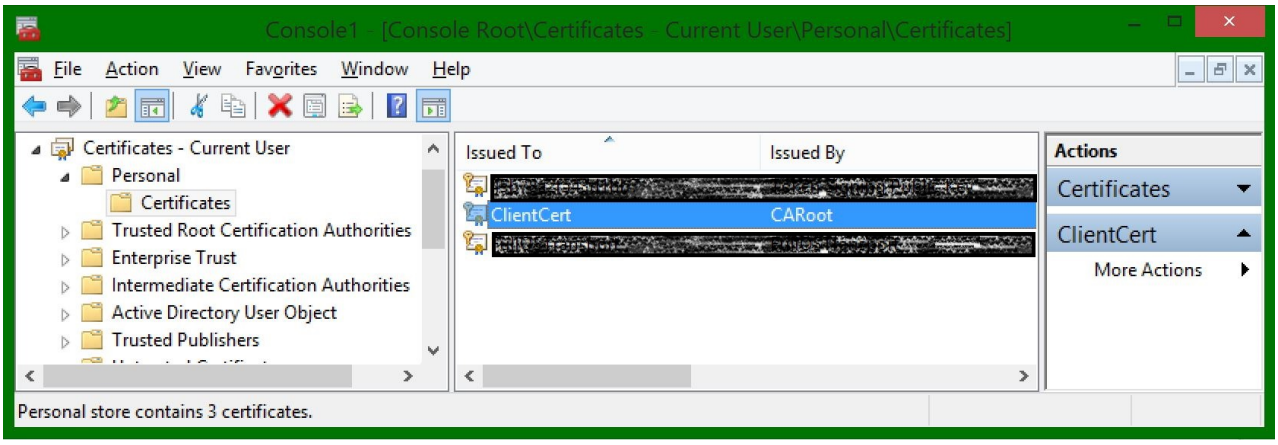
Open the Personal folder ➜ Right-click Certificates ➜ Import…

Browse for your ClientCert.pfx file



Go next ➜ Type in the password to your pfx file (-po parameter from the batch file) ➜ Continue going next until finish and ”The import was successful” message box appears.

You should now see you newly imported certificate in your Personal ➜ Certificates folder



Again the certificate is trusted because the CARoot is trusted by Windows.

You can now configure your client to use this certificate.

I hope the whole self signed certificate creation together with the makecert.exe generation tool feels more understandable and that you can use this knowledge for your development process. For a walk-through on setting up IIS to use your self-signed certificates check out

my next blog post: http://blog.jayway.com/2014/10/27/configure-iis-to-use-your-self-signed-certificates-with-your-application/

Check out my blog post for getting self signed certificates to work with a Windows Azure cloud service: http://blog.jayway.com/2015/04/21/configure-a-windows-azure-cloud-service-to-use-your-self-signed-certificates-for-iis-client-certificate-mapping-authentication/

Take care! =)