===========================================================

**I. Set up your own CA using OpenSSL**

===========================================================

1. Create a working directory for your CA. In the following, we use [dir] to denote your working directory.

2. Download **openssl.cnf** from Elearn to [dir].

3. Run the following command to generate your self-signed root certificate.

**openssl req -config openssl.cnf -x509 -newkey rsa:2048 -keyout mycakey.pem -out mycacert.pem -days 365**

During the process, your will see “Enter PEM pass phase:”. Please type your chosen pin, e.g,. *asdf*. Your keystrokes will **not** be echoed back on the screen, but they will be accepted as the pin to encrypt the RSA private key. Remember your pin which is used to decrypt the CA private key, mycakey.pem.

For “Common Name” input, choose a name for your CA, for example, “CS440 CA”. You may leave “Email Address” empty

4. If successful, you get **mycakey.pem** file which is your CA’s private key and **mycacert.pem** which is your CA’s public key certificate.

5. Run the following command to check the Subject and Issuer of your CA certificate.

**openssl x509 -in mycacert.pem –text**

Is your CA certificate a self-signed root CA? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. You need to the following things to run a CA to use mycakey.pem and mycacert.pem

a) Create a folder called “**private**” under your current working directory (i.e. [dir]) . This folder contains the private key of your CA. Run the command line:

**mkdir private**

b) copy **mycakey.pem** to “ **[dir]/private**”

c) Create a folder called “**newcerts**” under your current working directory; It contains new certificates issued in the future.

**mkdir newcerts**

d) download “**serial.txt**” and “**index.txt**” from eLearn to your working directory;

Now, you are ready to issue public key certificates!

===========================================================

**II. Create your RSA key, certificate request and issue a certificate using your own CA.**

1. You can generate your RSA private key with 2048 bits.

**openssl genrsa -out myrsakey.pem 2048**

2. Generate a certificate request using your newly generated RSA key.

**openssl req -config openssl.cnf -new -key myrsakey.pem –out req.pem**

The command will prompt you to input your personal information. If you want to use this private key for your email, you must use your true email address and real name when generating the certificate request; Otherwise, any information is fine. The command will ask you to choose a password called "PEM pass phase", which is used for encrypting your private key. Remember this password. Your new certificate request is the file **req.pem**.

**If you want to use it for your email signing (in the Lab IV), please input your real email address when generating the request.**

3.. Then, to use your CA to issue the certificate:

**openssl ca -config openssl.cnf -in req.pem -out myrsacert.pem**

The command requires you to input the previous pin to use CA's private key **mycakey.pem**. The CA should be able to issue a certificate for you. Check the file **myrsacert.pem** is created with a non-zero length. You can read it by any text editor.

==================================

**III. File Signing and Verification**

==================================

1. Choose any file (e.g. "message.doc") to sign

**openssl dgst -sha1 -sign myrsakey.pem -out signature.bin message.doc.**

This command signs your chosen file with your private key **myrsakey.pem**. The resulting signature is the file "**signature.bin**".

2. Extract the public key from your certificate.

**openssl x509 -in myrsacert.pem -pubkey -noout > mypublickey.pem**

The resulting "**mypublickey.pem**" contains your RSA public key.

3. Verify the signature with the public key

**openssl dgst -sha1 -verify mypublickey.pem -signature signature.bin message.doc**

The signature "**signature.bin**" should be verified correctly for the file "**message.doc**". If you modify message.doc, the verification will fail.

**Experiment:** send to your classmate the file and the signature from you, as well as other data the receiver needs (you figure out what to send ☺). Check whether your receiver can also verify the file with the signature.

===================================

**IV. Verification**

====================================

1. Verify the certificate

**openssl verify -CAfile mycacert.pem myrsacert.pem**

This verifies the your certificate with your root CA's certificate "**mycacert.pem**".

2. Verify your CA certificate

**openssl verify mycacert.pem**

Is a self-signed certificate?

3. Verify your certificate request

**openssl req -in req.pem –verify –text –noout**

Does the certificate request contain a signature? What’s the key used to generate this signature?

===========================================================

**IV. Have fun with your home-made certificate.**

===========================================================

You can import the credential as your personal certificate to Outlook so that you can sign your outgoing emails. Do a Google search on how to install your personal certificate to various applications on different platforms.

Below are the instructions for installing certificates and credentials to Outlook.

1. To create a PKCS12 credential.

**openssl pkcs12 -export -inkey myrsakey.pem -in myrsacert.pem -CAfile mycacert.pem -out mycrediential.p12**

The command will first ask you to input the password to open your RSA private key. Then, it will also ask you to choose a password for encrypting your credential **mycredential.p12**. Remember your selection. Keep in mind that **mycredential.p12** is your personal credential which includes your public key and private key.

2. Import your credential to Outlook.

Use the “Alternate method” in <https://kb.iu.edu/d/bcta#install>

(a) Click “File | Options”, then click “Trust Center”

Graphical user interface, text, application

Description automatically generated

(b) Click “Trust Center Settings…” , followed by selecting “Email Security”

Graphical user interface, text, application

Description automatically generated

(c) Click “Import/Export” under “Digital IDs (Certificates) to import your personal certificate created priorly.

Graphical user interface, application

Description automatically generated

(d) Click “Browse …” button to load your file. In the “Password:” box, input your pin used in generating the PKCS12 file.

(e) Click OK and complete import.

3. If you try to send an email, you are not able to sign it. The reason is that Windows cannot verify your certificate generated from OpenSSL. To fix this issue, you have to import your self-signed CA certificate to Windows.

(Starting from Step 7 in <https://windowsreport.com/install-windows-10-root-certificates/>)

1. Press Win key +R hotkey and enter “certmgr.msc” in Run’s text box to open the certificate manager.
2. Click **Trusted Root Certification Authorities** and right-click **Certificates** to open a context menu.
3. Select **All Tasks** > **Import** on the context menu to open the window shown below
4. Press the **Next** button, click **Browse,** and then select the digital certificate root file (i.e., mycacert.pem). In the **Open** dialog, make sure you select “All Files (\*.\*)” as the filter.
5. Select the radio button for “Place all certificate in the following store” and then click **Next**. In the end, click **Finish** button to complete CA certificate import.
6. Under “Certificates | Personal | Certificates”, you will find your personal certificate on the right panel. You can double click the certificate to see the details. Under the “Certification Path” tab, you should see “This certificate is OK” message in the bottom. It means your personal certificate is trusted by Windows now.
7. Congratulations! You can now digitally sign your email!

4. In your Outlook, compose a new email. Click “Options” in the email’s window. You should see the Sign button. If it is set, Outlook will sign your outgoing email.

Graphical user interface, application

Description automatically generated