The University of Newcastle
School of Information and Physical Sciences

SENG2250/6250 System and Network Security
Week 10 Lab

Part 1: Create an SSL Protected Web Site

SSL, which stands for Secure Sockets Layer, is a crucial security protocol that plays a pivotal role in ensuring the confidentiality and integrity of data transmitted over the internet. It achieves this by encrypting the data exchanged between a web server and a client, such as a web browser. SSL provides a secure communication channel that safeguards sensitive information, like passwords, credit card numbers, and personal details, from unauthorized access or interception by malicious actors. With the widespread adoption of its successor, TLS (Transport Layer Security), SSL has formed the foundation of secure internet connections, enabling safe online transactions, secure email communication, and the protection of sensitive data in transit. It has become an integral component of internet security, assuring users that their online interactions are shielded from prying eyes.

The objective of this lab is twofold:

1. Create an SSL-protected website.
2. Inspect SSL handshake traffic.

Lab Softwares and how to access them:

1. This lab should be conducted under the Windows 10 VM. Access your virtual lab here:

   https://cybersec-vra04.newcastle.edu.au/

   Username: Student

   Password: $tud3nt

2. This lab requires XCA, XAMPP, Firefox, and Wireshark.

   - XCA 2.4.0 or later: https://hohnstaedt.de/xca/index.php/download
   - XAMPP 8.0.10 or later: https://www.apachefriends.org/download.html
   - Wireshark 3.4.8 or later: https://www.wireshark.org/#download
   - Firefox: https://www.mozilla.org/en-US/firefox/new/

(Tasks 1–4 were completed last week, but you'll need to repeat these steps to move forward)

Task 1: Creating a Certificate Authority (CA) using XCA

In this task, we will first create a CA. The role of CA is to issue public-key certificates to end entities (e.g., our web server). In other words, the CA digitally signs a public key of a web server and embeds it into the server's certificate. The CA uses its private key for signing. In practice, the CA certificate manager (a certificate server) should be installed on a secure machine (potentially disconnected from the network); the CA's private key should be kept highly secure.

The CA also issues a self-signed public-key certificate, whereby the CA digitally signs its own public key. This certificate is distributed, in a secure way (the integrity of the CA's certificate must be protected),

to all users who use a certificate issued by the CA to verify the authenticity of the certificate holder (e.g., a web server).

1. Open XCA (e.g., C:\Program Files\xca\xca.exe)
   Create a new database to store your keys and certificates. Select “File” and “New DataBase", create a new database named "mydatabase”. You will be asked for a password, but it is not necessary for this lab. You can simply click “Ok”.
2. Click the “New Certificate” button to start the X.509 certificate creation procedure.

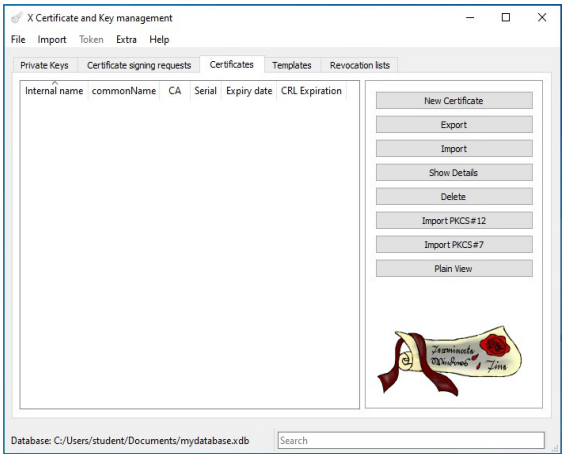

Figure 1. Starting new certificate generation.

3. Select CA template for this certificate.
4. Select the “Subject” tab and fill the fields, i.e., “Internal name”, “Country name”, etc. Please refer to Figure 2 as an example. Click “Generate a new key” button to generate a new private key (2048-bit RSA). This will be the signing key of the CA.
5. Select the “Extensions” tab and set the certificate “Type” to “Certificate Authority”, then click “OK” to finish the certificate creation procedure.

   You can check the details of the created CA certificate by double-clicking on it in the main XCA window under the “Certificate” tab.

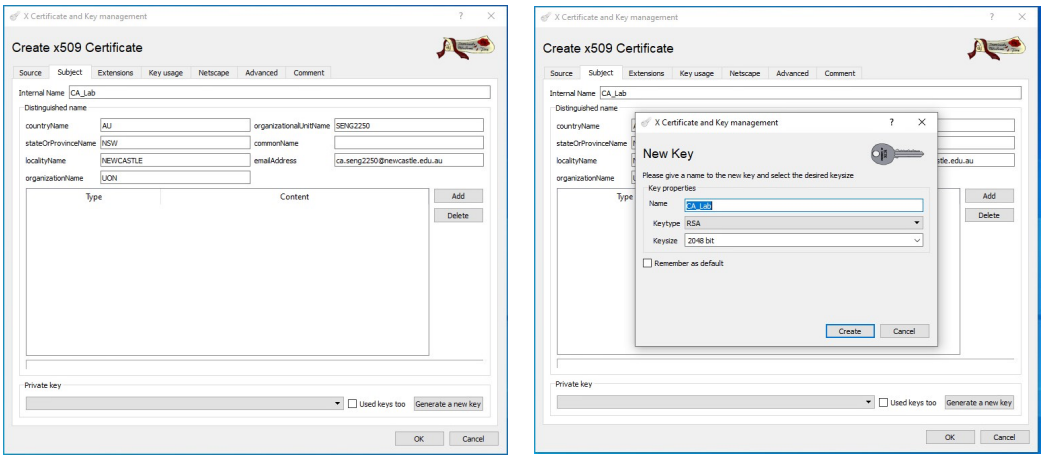

Figure 2. Create a CA certificate.

Task 2: Create a Web Server Certificate

In this task, we create a public-key certificate for a web server. This certificate will be digitally signed by the created Certification Authority (CA).

1. Click the “New Certificate” button in the main XCA window. Select the “Source” tab and configure the web server certificate properties as follows:
    a. Check “Use this Certificate for signing” and set it to the name of the CA created in the previous task.
    b. Choose “TLS_server” (or “HTTPS_server”, if applicable) from the list of available templates.
2. Select the "Subject" tab and fill the fields appropriately. Please refer to Figure 3 as an example. It is important to note that fill the commonName as “localhost”. This will be the IP address (or the corresponding URL) of your web server. Finally, click “Generate a new key” button to generate a new private key (2048-bit RSA).

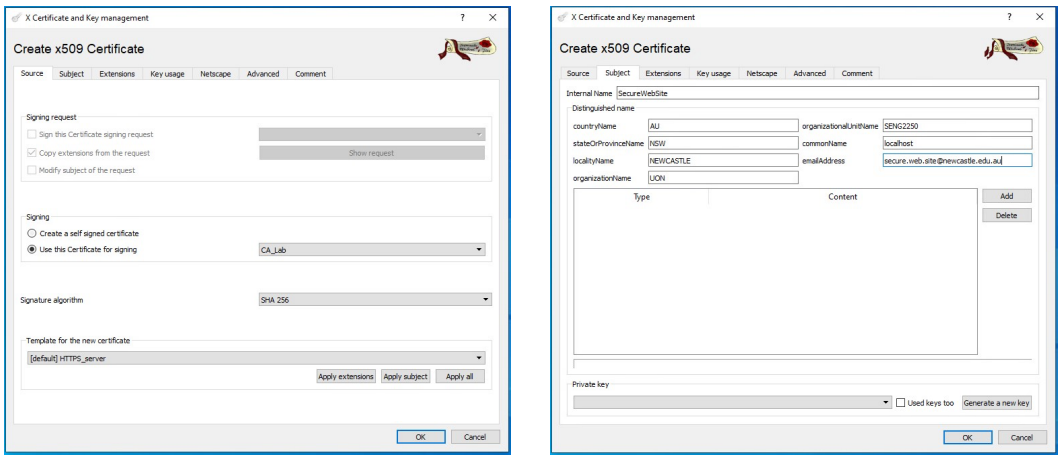

Figure 3. Create web server certificate.

3. Under the “Extensions” tab, set the type to “End Entity”, then click “OK” to finish the certificate creation procedure.

Task 3: Create a Client Certificate

In this task, we create a public-key certificate for a client. The CA will sign this certificate.

1. Click the “New Certificate” button in the main XCA window. Select the “Source” tab and configure the client certificate properties as follows:
    a. Check “Use this Certificate for signing” and set it to the name of the CA created in the previous task.
    b. Choose “TLS_client” (or “HTTPS_client”, if applicable) from the list of available templates.
2. Select the “Subject” tab and fill the fields properly. Please refer to Figure 4 as an example. Then, click “Generate a new key” button to generate a new private key (2048-bit RSA).

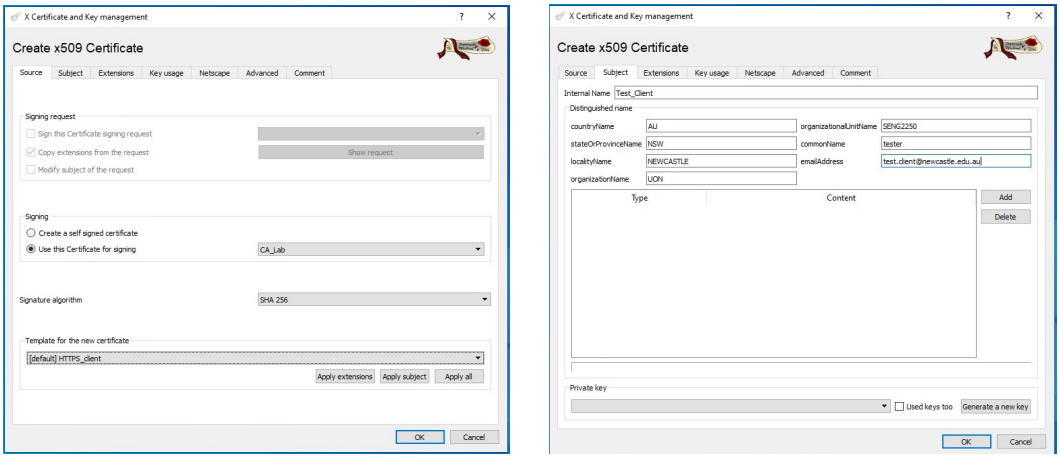

Figure 4. Create client certificate.

3. Under the “Extensions” tab, set the type to “End Entity”, then click “OK” to finish the certificate creation procedure.

Task 4: Export Certificates

In this task, we will export the certificates created in the previous tasks, that is, the CA public-key certificate (without the private key), the web server public-key certificate including its private key, as well as the client public-key certificate with its private key.

1. Open the “Certificates” tab in the main XCA window. Select the CA certificate and click “Export”. Select a destination (e.g., Documents) and filename where you want to store the certificate and click “OK”.
2. Repeat the above step to export web server and client certificates.
3. Finally, we export the web server and client private keys.
    a. For web server: click the “Private Keys” tab from the XCA main window. Select web server private key and click “Export”. Choose a destination (e.g., Documents) and filename where you want to store the private key. Finally, click OK to export the web server private key. (Figure 5)

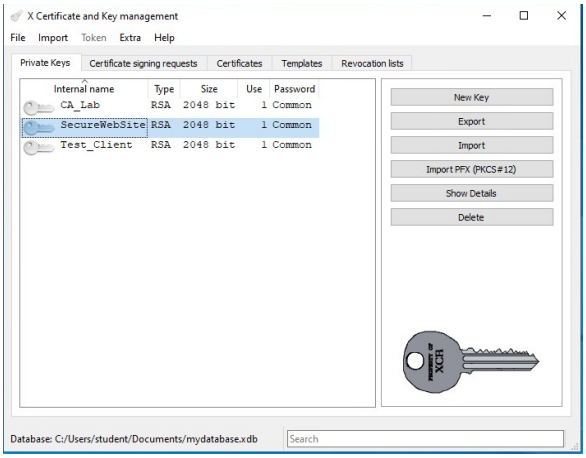

Figure 5. Export web server private key.

    b. For client: select the client certificate under the “Certificates” tab and click “Export”. Select a destination (e.g., Documents) and filename where you want to store the private key. Then, change the “Export Format” to “PKCS #12 (.p12)” (Figure 6). It will

be imported to the web browser. You will be asked to create a password; you may simply click “OK” to skip it.

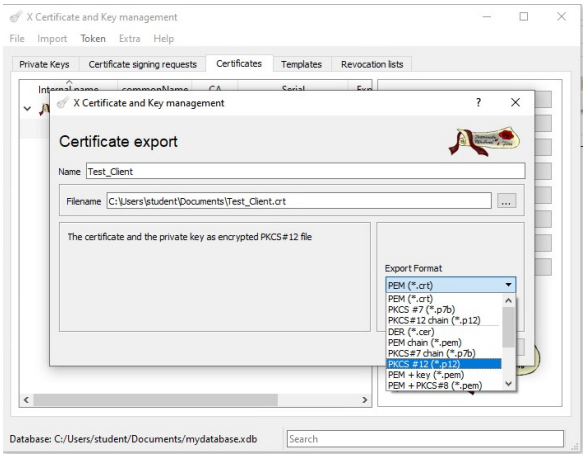

Figure 6. Export client private key.

You should have created files shown as in Figure 7.

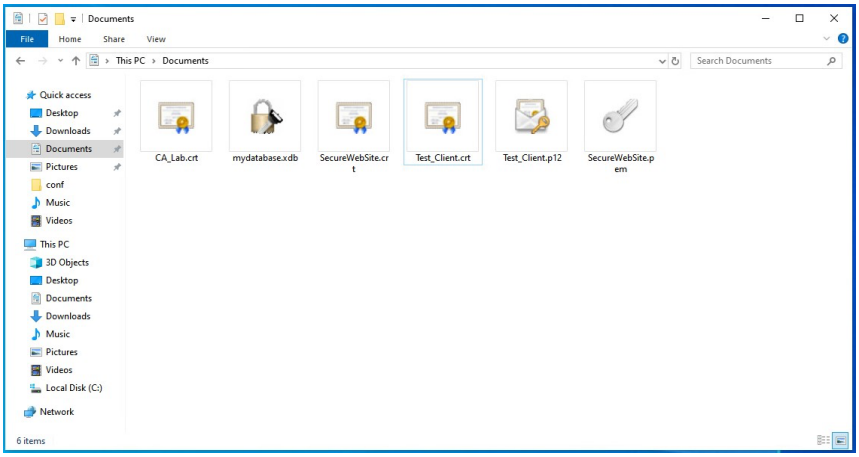

Figure 7. Created files.

Task 5: Configure Apache Web Server

1. To configure the Apache Web Server, we will need the CA_Lab.crt and SecureWebSite.crt public-key certificates as well as the private key SecureWebSite.pem. Copy these certificates and keys to the conf folder of the Apache Web Server (e.g., C:\xampp\apache\conf).
2. Edit httpd.conf (e.g. found within “C:\xampp\apache\conf” folder) and ensure the following lines are uncommented:

   LoadModule ssl_module modules/mod_ssl.so

   Include conf/extra/httpd-ssl.conf

3. Edit httpd-ssl.conf (e.g. found within “C:\xampp\apache\conf\extra” folder) and verify that the following holds (should be uncommented and the value is correct):
   a. SSLCertificateFile ‘‘C:/xampp/apache/conf/SecureWebSite.crt’’
   b. SSLCertificateKeyFile ‘‘C:/xampp/apache/conf/SecureWebSite.pem’’
   c. SSLCertificateChainFile ‘‘C:/xampp/apache/conf/CA_Lab.crt’’
   d. SSLCACertificateFile ‘‘C:/xampp/apache/conf/CA_Lab.crt’’

e. SSLVerifyClient require
f. SSLVerifyDepth 2

4. Start XAMPP Control Panel. Run the program: “C:\xampp\xampp-control.exe”.
5. Start/Restart the Apache server.

Task 6: Test Your Configuration

In this task, we will install a client public-key certificate in the browser and test our configuration. To accomplish this goal in a Firefox browser, we can place the client's public-key certificate in the "Your Certificates” directory. Here are the steps to install the client public-key certificate.

1. Go to “Settings” of Firefox.
2. Click “Privacy & Security”, on the left of the page.
3. Go to “View Certificates” to manage your SSL certificates and settings.
4. Click on the “Your Certificates” tab and select “Import...”.
5. Select the client’s .p12 file (e.g., Test_Client.p12) and click “Open”.
6. You will be asked to enter the password created while exporting client public-key certificate to pkcs12. If you didn’t create a password, leave it blank.
7. Click “Ok”.
8. Open a new tab in Firefox and enter the following address in the address bar:

   https://localhost/dashboard

   Do you get any warning messages? Why?

9. We would like to eliminate this warning message. What can we do in this regard?
10. Recall that the CA has digitally signed the web server public-key certificate. So if our web browser would have access to the CA certificate (i.e., if it would trust this certificate), the web browser could successfully verify the digital signature in the web server certificate and would not report any warning messages.
11. To accomplish this goal in a Firefox browser, we can place the CA’s certificate in the “Authorities” directory. Here are the steps to install the CA’s certificate:
    a. Go to “Settings” of Firefox.
    b. Click “Privacy & Security”, on the left of the page.
    c. Click “View Certificates”.
    d. Click on the “Authorities” tab and select “Import...”.
    e. Select the CA’s .crt file (e.g., CA_Lab.crt) and click “Open”.
    f. Tick both checkboxes.
    g. Click “Ok”.

12. Now that you have installed the CA certificate, try again to access:

    https://localhost/dashboard

    Do you get any warning messages? Why?

Part 2: Assignment 2 Help