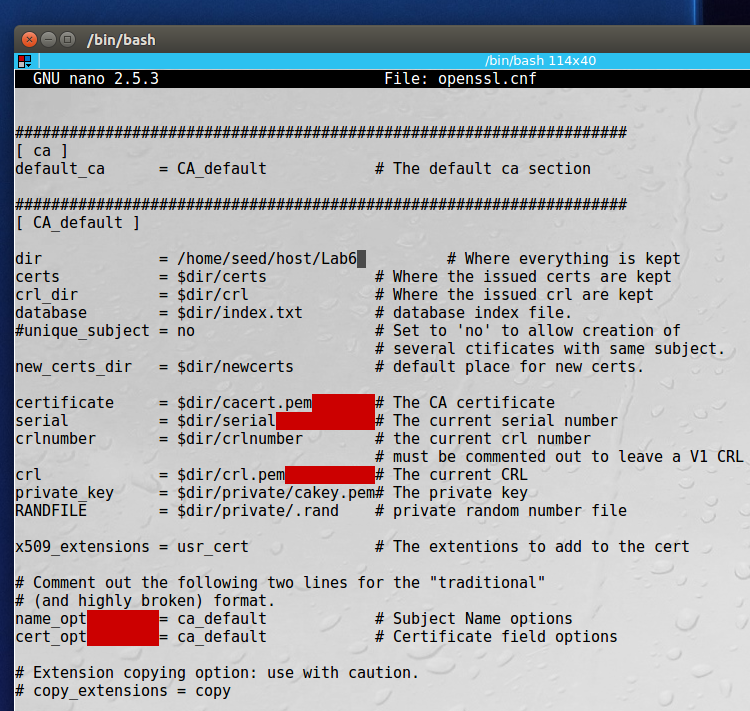
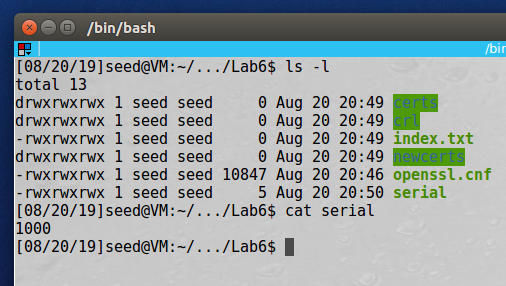
Internet Security - Lab 6 Report

# Task 1: Becoming a Certificate Authority (CA)

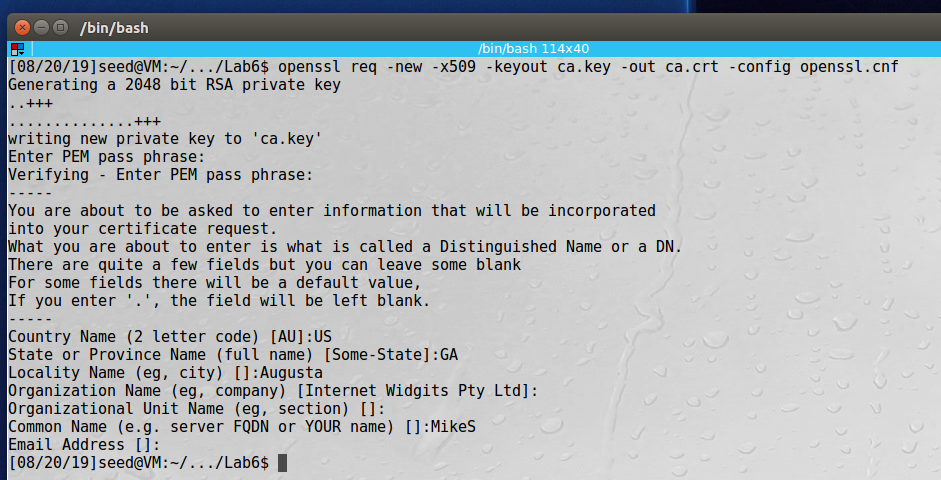
The first thing we do is copy the conf file template over to the working directory /home/seed/host/Lab6, and edit the conf file to show the base directory:



We then add the directories that we need, as well as the two files: index.txt and serial:



Next we generate the self-signed certificate and enter the relevant information, the passphrase is “deesdees”:

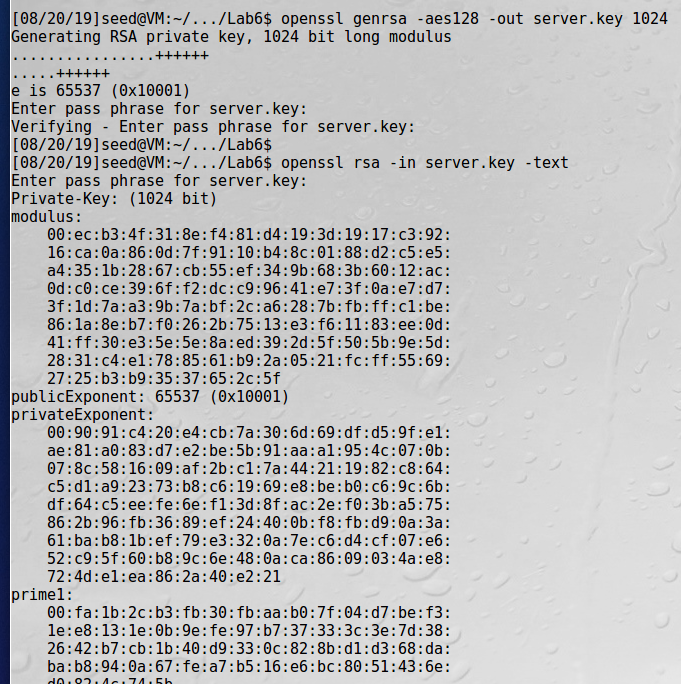


**Observation**: In this task we had to create the process to become a Certificate Authority. We first copying the template configuration file over into our working directory, then we created the necessary subfolders and files to support the CA. We then needed to create a self-signed certificate for the CA to use. This was acomplisthed using the openssl req command. The password used for this, and all other certs for this lab was “deesdees”.

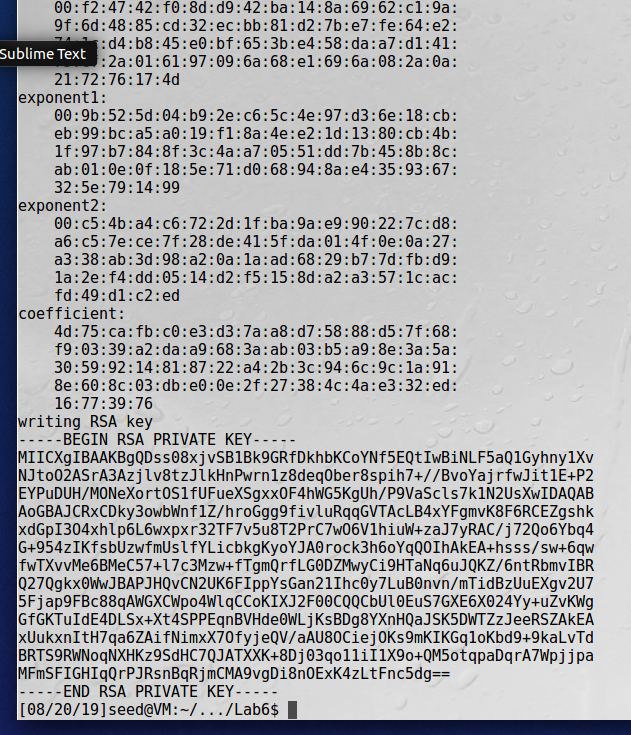
**Explanation**: A certificate is required part of the PKI encryption scheme. The CA is a trusted entity that verifies the authenticity of the user/server certificates. We didn’t want to pay a commercial provider, like verisign, to act as our CA so we created our own using root.

# Task 2: Creating a Certificate for SEEDPKILab2018.com

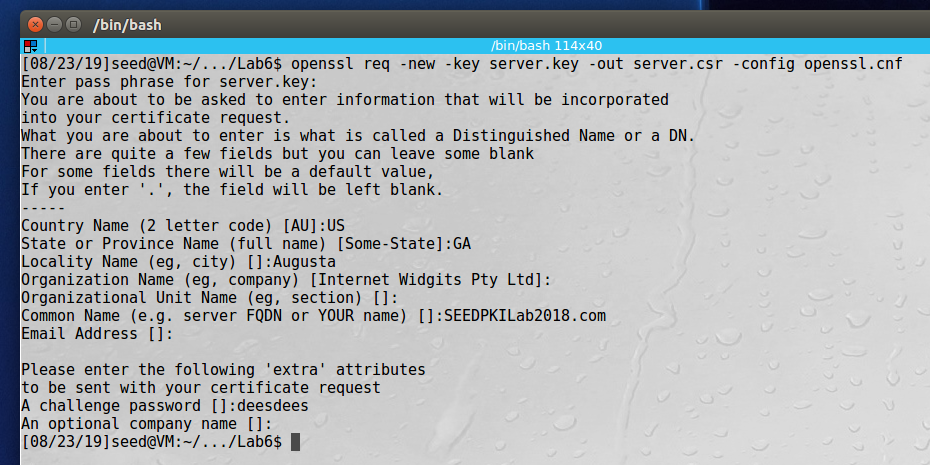
Next we create the RSA public/private key pair, again we are using “deesdees” for the passphrase:



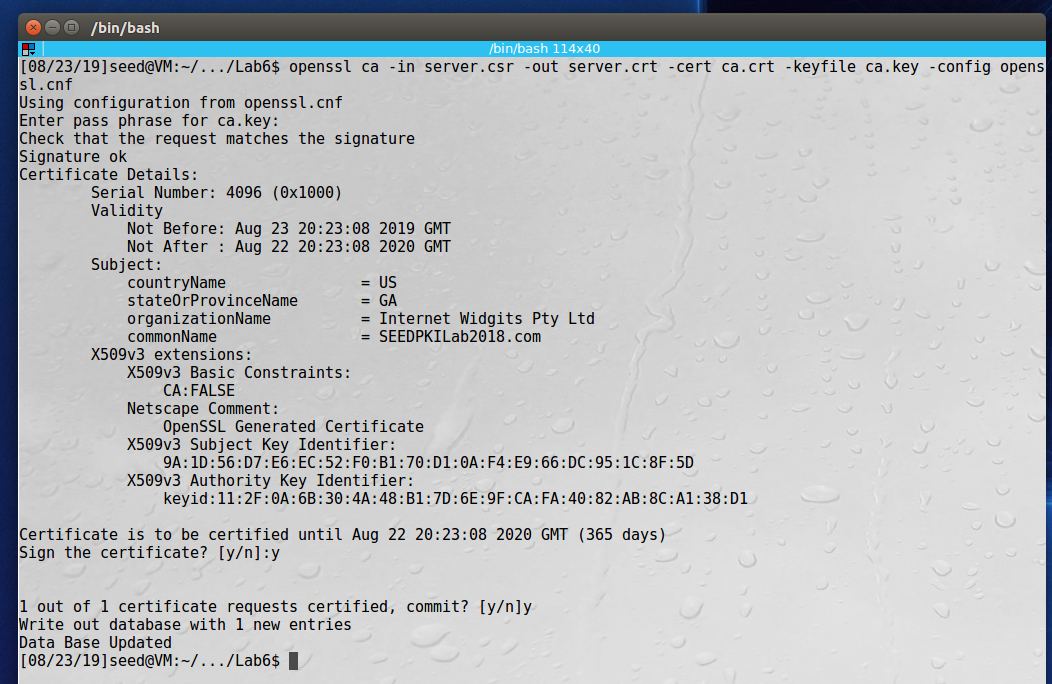
And the rest of the key:



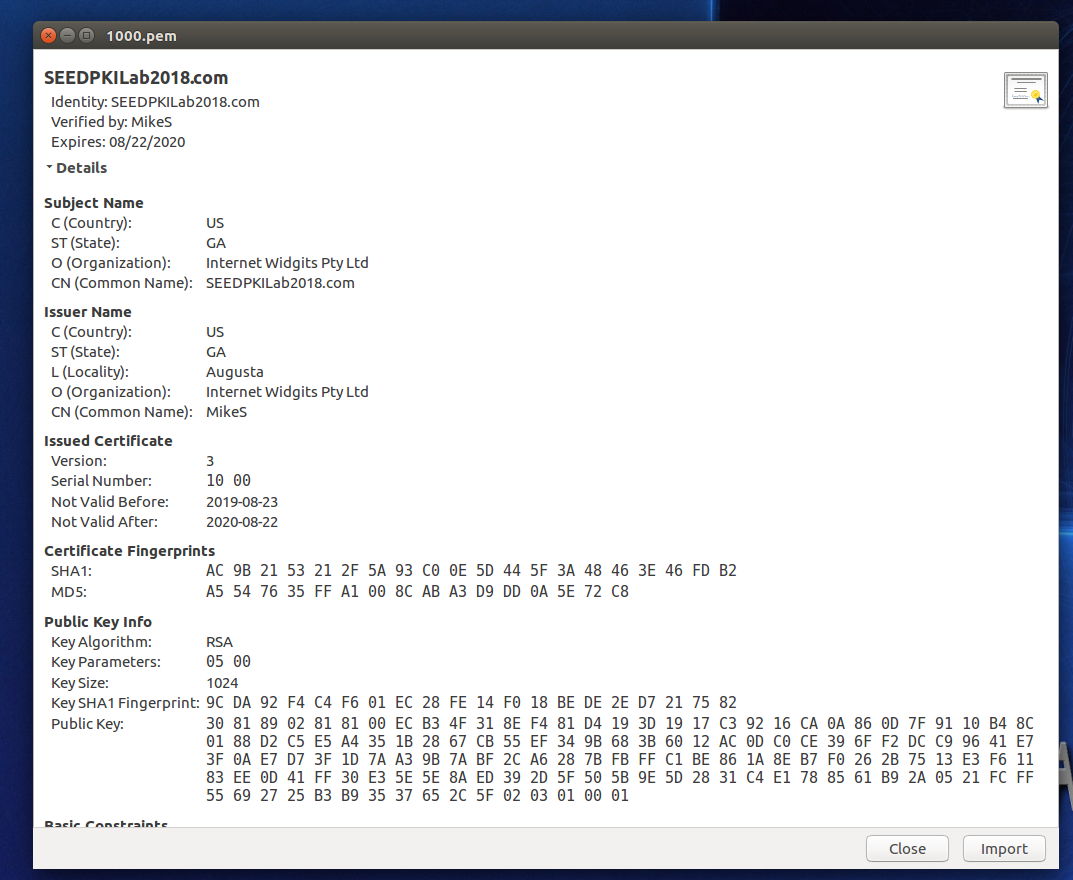
Next, we generate a certificate signing request (CSR):



And finally, we generate the certificates:



And this is our new signed certificate, 1000.pem:

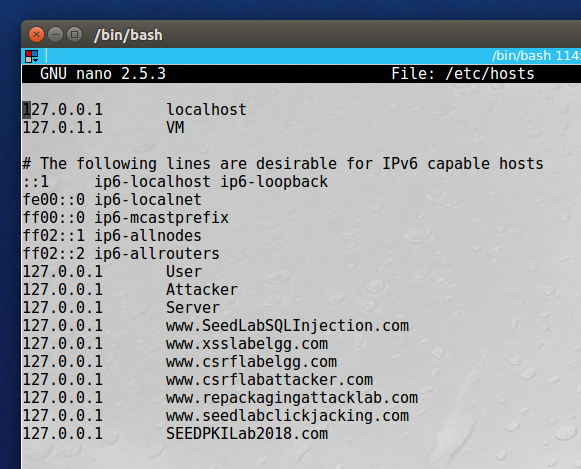


**Observation**: In this task we created the certificates needed to secure our website, SEEDPKILab2018.com using https. We used the authority that we created in the previous task to issue the certificate. We followed three steps to create the certificate, first we had to generate the public/private key pair, which was stored in server.key. Second, we generated a signing request which included the public key and is sent to the CA. Third, we use the signing request, server.csr to generate the certificate using our CA. We used the ‘openssl ca’ command for this.

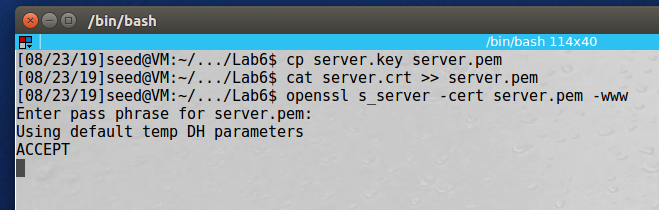
**Explanation**: The process for issuing valid certificates from a CA involves three parts, generating the public/private key pair, creating a certificate signing request using the public key, then issuing the certificate. With this certificate we will be able to secure our website in the next tasks.

# Task 3: Deploying Certificate in an HTTPS Web Server

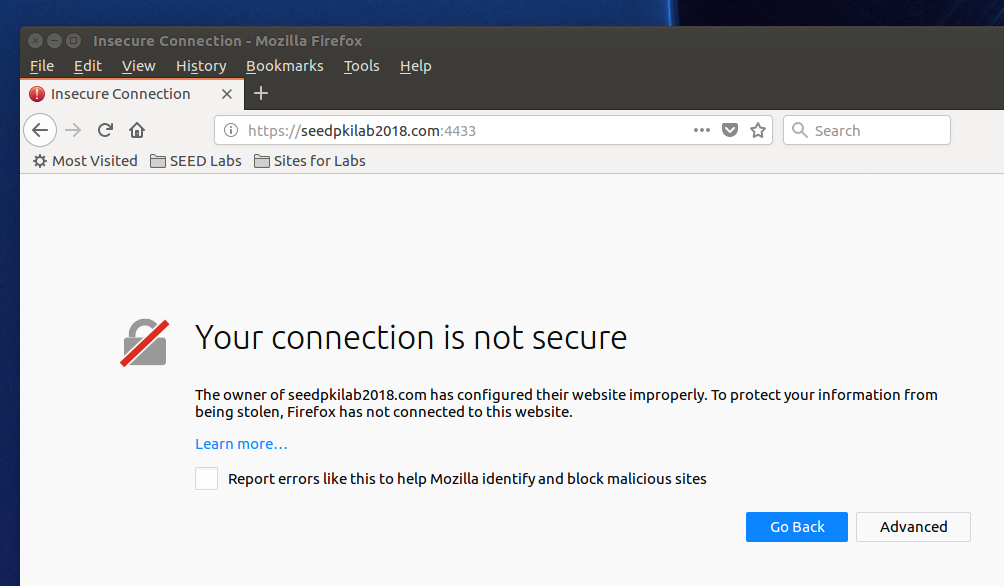
The first thing we do is edit the /etc/hosts file to redirect SEEDPKILab2018.com to our localhost:



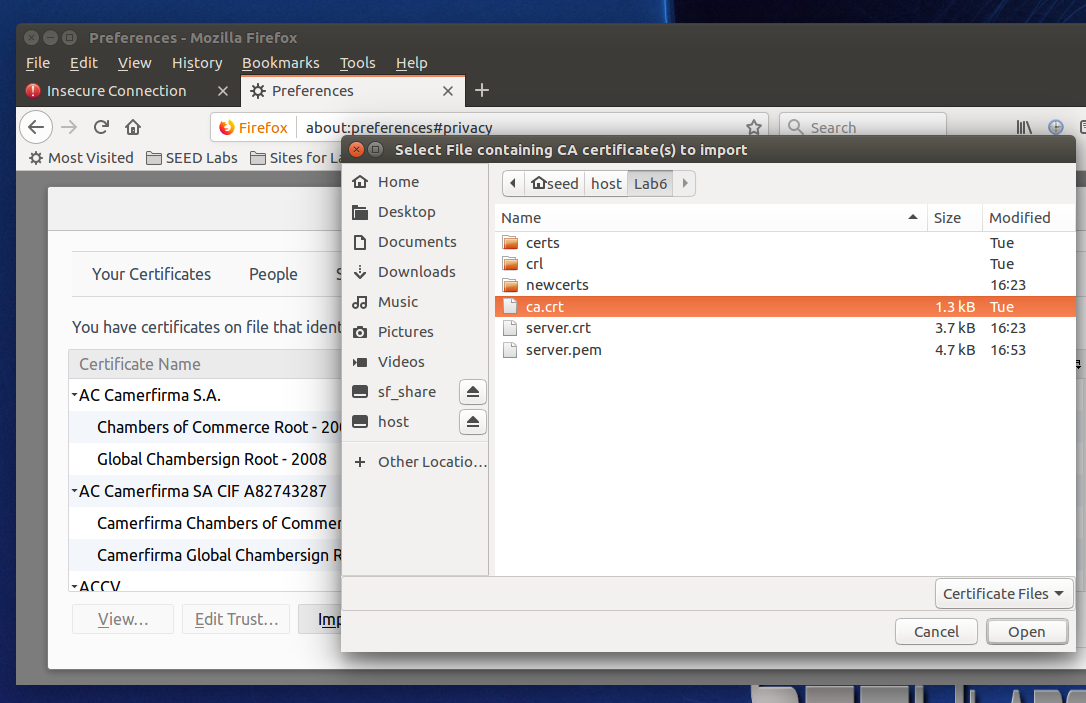
Next we configure our webserver to use our certificate and will be available at <https://SEEDPKILab2018.com>:



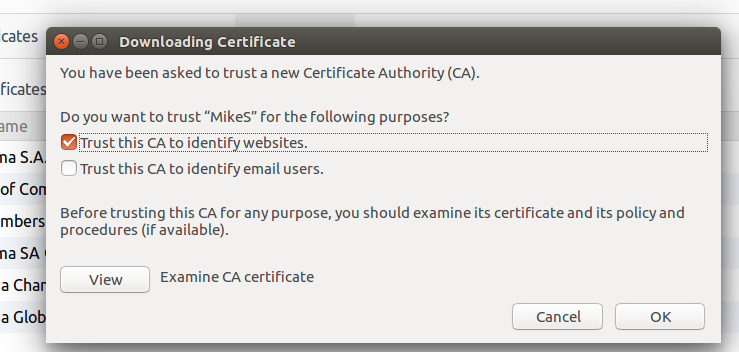
When we visit our website, we get a warning from Firefox that our connection is not secure since we still need to install our certificate repository:



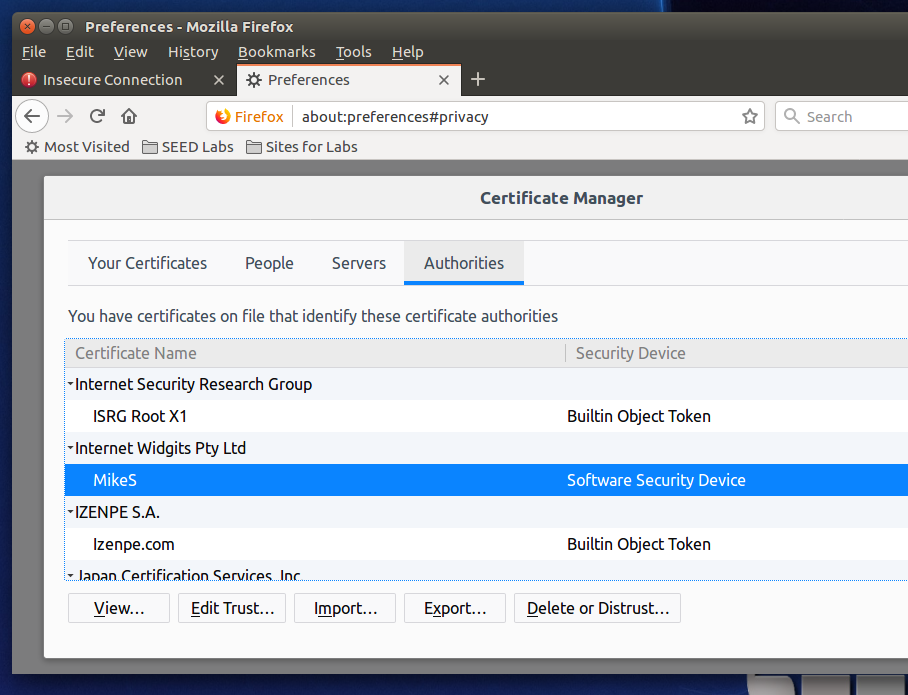
Now we installed the Certificate Authority that we made to Firefox:



Trusting the CA:



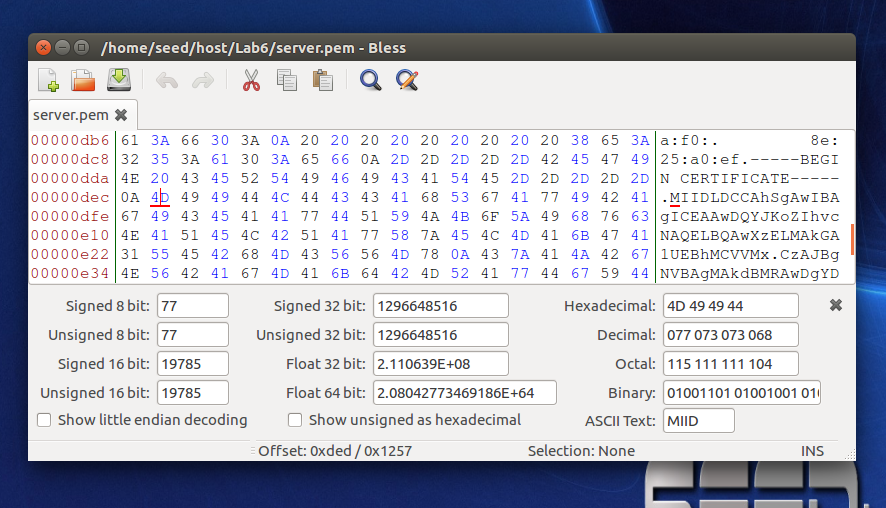
We can see the certificate now is in our list of trusted certificates with “MikeS” as the signer:



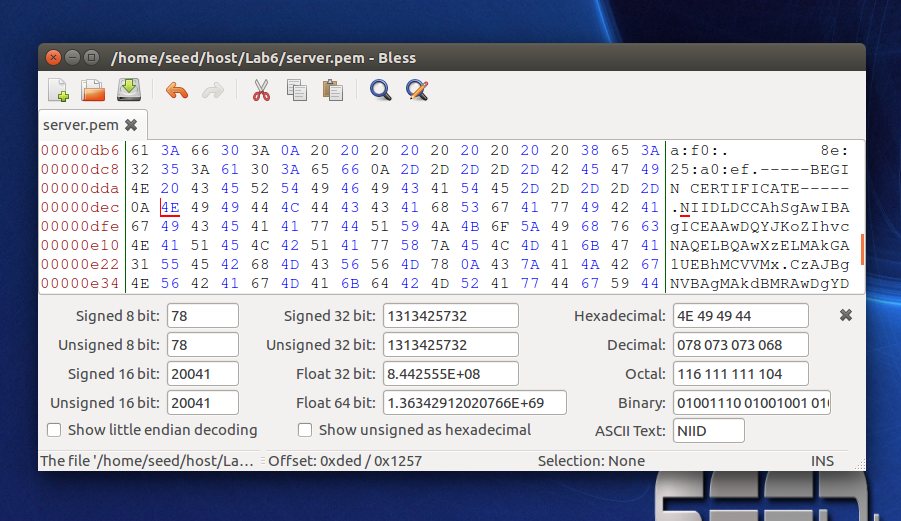
And when we try to visit our website again, we don’t get a security warning:



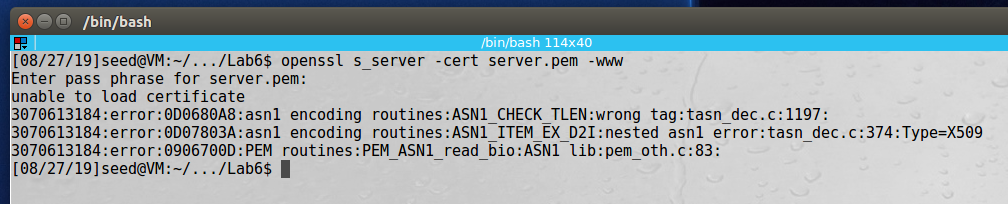
For the next part we will change one byte on server.pem, here is the file before we change the byte:



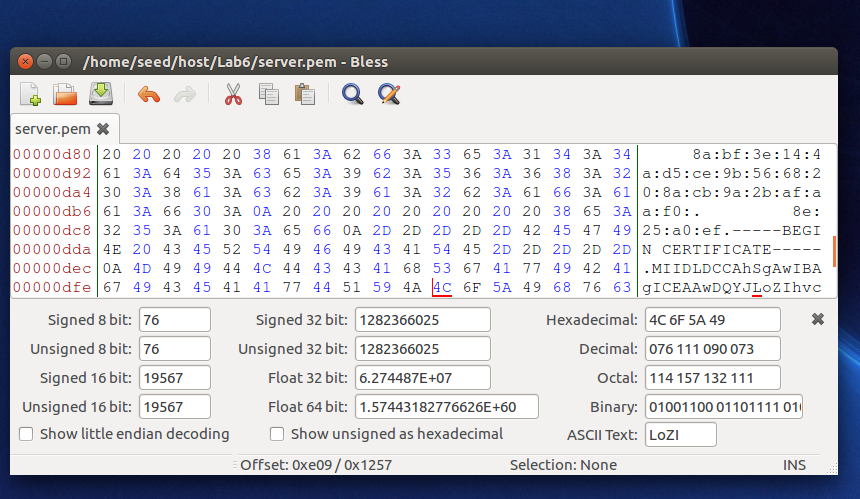
And now we change the byte from a “4D” to a “4E”, so from an “M” to an “N”:



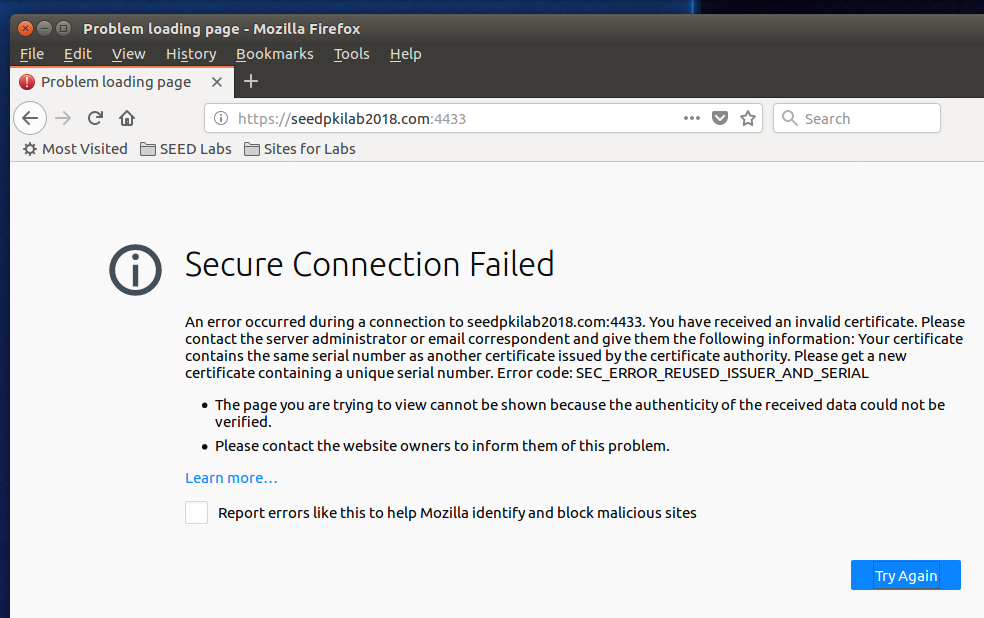
And when we try to start the server it will not start:



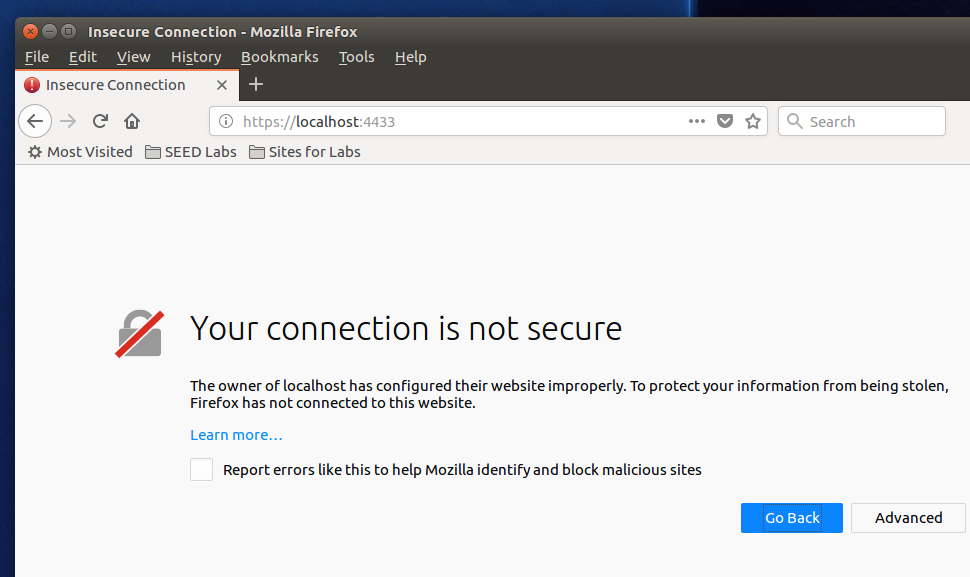
So we have to restore our good server.pem and try a different byte to alter, the highlighted byte was originally “4B” and we changed it to “4C”:



Now we can start our server, but when we browse to <https://seedpkilab2018.com:4433>, we get a failure:



If we attempt to go to <https://localhost:4433> we get a security warning because the certificate was registered for <https://seedpkilab2018.com:4433> and not for localhost:



**Observation**: In this task we deploy and test our certificate and then try to corrupt it to see the result. We first had to add the site to our /etc/hosts file, and then we combined the secret key and certificate into one file, server.pem and launched our web server using s\_server. When we first visited the site in Firefox, we were prompted with a security warning because we hadn’t added our CA to Firefox’s trusted CAs. After we added the CA to Firefox to trust, we no longer get a security warning and can now see our website working properly. We then go into the server’s certificate to corrupt a byte, restart our webserver to see if it still works, but Firefox tells us that our connection is no longer secure.

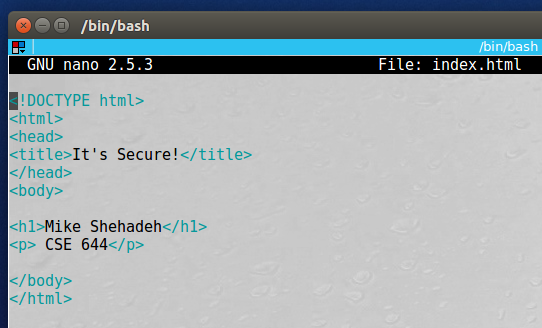
**Explanation**: Deploying the certificate to the website allowed us to visit our site securely using https. We first needed to at the CA that we created to Firefox since we didn’t use a commercial provider like Verisign. When we corrupted our certificate file and tried again, after only changing one byte, Firefox gave us a security warning.

# Task 4: Deploying Certificate in an Apache-Based HTTPS Web Server

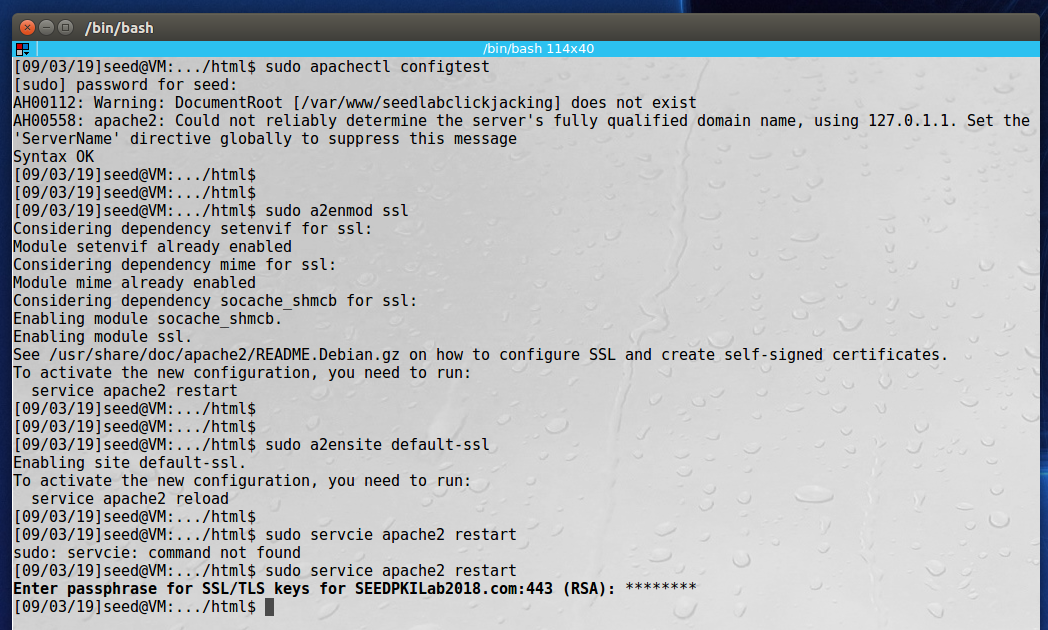
First, we modify the default-ssl.conf file to include information about our website:



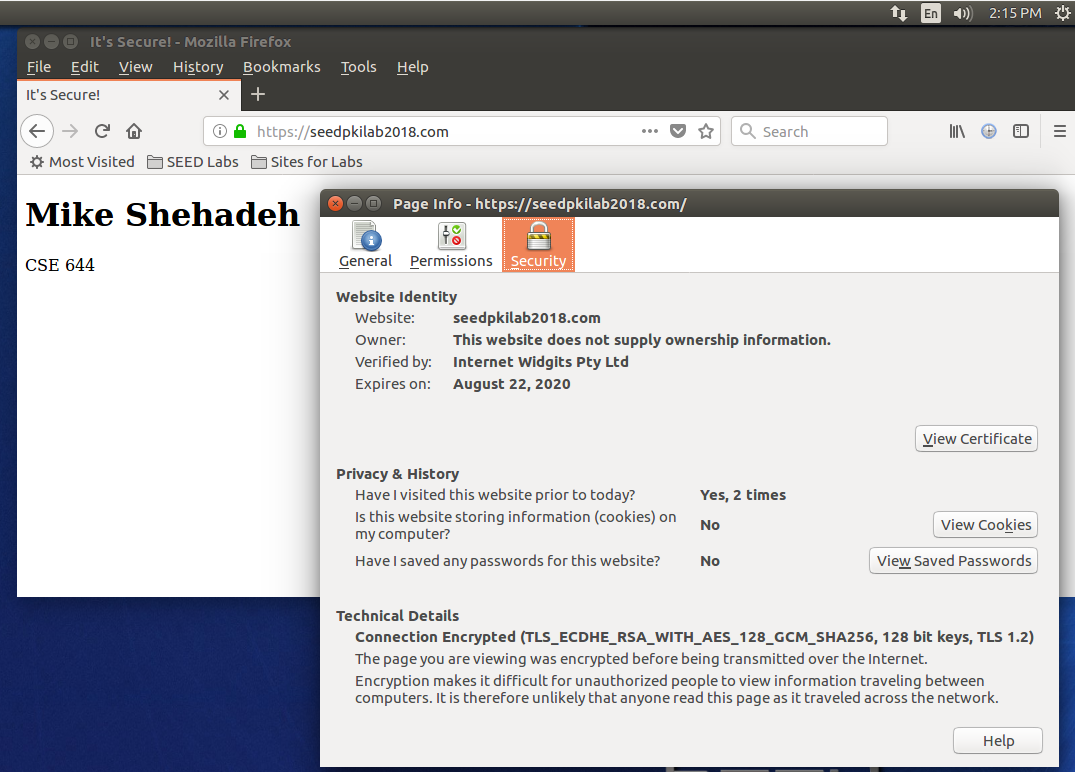
Next we create a simple website located at /var/www/SEEDPKILab2018:



Next we run some commands to check that our site’s security is properly configured:



After restarting our Apache2 webserver we can see that our site is secure:



**Observation**: This task was similar to the previous except we used Apache as our webserver. Apache is a popular webserver and is widely used in the real world. We first had to edit the file at /etc/apache2/default-ssl.conf file to point to our website and certificates. Then we created a simple html file in the /var/www/SEEDPKILab2018 folder. We then ran some command to check if everything was configured properly, restarted our apache server, and visited the site in Firefox. When we visited the site, it showed as secure in the browser, with valid certificates since we previously installed the CA.

**Explanation**: Apache is an easy to configure and deploy web server that we used to host our website SEEDPKILabs2018.com. We were able to have our secure website up and running in only a few minutes after configuring our certificates and browser to trust them.