PKI Lab

Task 1: Becoming a Certificate Authority (CA)

We are generating a self-signed certificate for our CA. This means that our CA is totally trusted, and its certificate will serve as the root certificate. This can be done by using the below command using openssl, this will generate certificate and key for the CA.

```
[04/09/2020 16:07] Rakshith-10.0.2.15@VM:~/PKI$openssl req -new -x509 -keyout ca.key -out ca.crt -config openssl.cnf
Generating a 2048 bit RSA private key
.....................................................+++
.....................+++
writing new private key to 'ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:NY
Locality Name (eg, city) []:SYR
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SU
Organizational Unit Name (eg, section) []:EECS
Common Name (e.g. server FQDN or YOUR name) []:RAK
Email Address []:rnallaha@syr.edu
[04/09/2020 16:08] Rakshith-10.0.2.15@VM:~/PKI$ls
ca.crt  ca.key  demoCA  openssl.cnf
[04/09/2020 16:09] Rakshith-10.0.2.15@VM:~/PKI$
```

RAK

Identity: RAK
Verified by: RAK
Expires: 05/09/2020

▾ **Details**

Subject Name

C (Country):	US
ST (State):	NY
L (Locality):	SYR
O (Organization):	SU
OU (Organizational Unit):	EECS
CN (Common Name):	RAK
EMAIL (Email Address):	rnallaha@syr.edu

Issuer Name

C (Country):	US
ST (State):	NY
L (Locality):	SYR
O (Organization):	SU
OU (Organizational Unit):	EECS
CN (Common Name):	RAK
EMAIL (Email Address):	rnallaha@syr.edu

Issued Certificate

Version:	3
Serial Number:	00 BE CD EC 65 27 03 EB 60
Not Valid Before:	2020-04-09
Not Valid After:	2020-05-09

Task 2: Creating a Certificate for SEEDWORKSHOP2020.com

Step 1: Generate public/private key pair.

We first create our own pair of public/private keys, we use the below command to generate the pair of RSA keys.

```
[04/09/2020 18:58] Rakshith-10.0.2.15@VM:~/PKI$openssl genrsa -aes128 -out server.key 1024
Generating RSA private key, 1024 bit long modulus
.....................................++++++
....++++++
e is 65537 (0x10001)
Enter pass phrase for server.key:
Verifying - Enter pass phrase for server.key:
[04/09/2020 18:58] Rakshith-10.0.2.15@VM:~/PKI$ls
ca.crt  ca.key  demoCA  openssl.cnf  server.key
[04/09/2020 18:59] Rakshith-10.0.2.15@VM:~/PKI$openssl rsa -in server.key -text
Enter pass phrase for server.key:
Private-Key: (1024 bit)
modulus:
    00:c6:1f:d6:5b:38:98:30:1d:06:c6:d2:30:75:2b:
    4e:8a:24:bf:27:d5:d6:e1:c8:07:bc:9b:e0:58:9f:
    f6:b9:cc:ef:63:e3:a8:d4:43:e9:72:76:4a:83:87:
    4e:b0:43:4f:09:91:a1:38:a1:9d:4c:7f:3f:20:79:
    51:49:35:c3:64:85:e1:1d:e7:61:6b:95:75:59:d0:
    bd:5c:f5:24:aa:13:e5:92:d4:31:d3:65:4d:92:85:
    76:bc:4f:cf:25:f2:18:84:f7:a4:be:e3:dd:9e:fd:
    35:a1:8f:e3:23:11:7e:f4:6e:d8:17:a9:01:32:10:
    2c:1e:8a:75:40:8c:05:3d:dd
publicExponent: 65537 (0x10001)
privateExponent:
    1f:79:31:49:3e:7d:56:af:55:c3:41:e2:b6:ca:51:
    68:ba:9b:af:4d:56:1c:79:f5:58:ad:fe:7e:b5:b2:
    b1:23:70:28:13:23:5f:cd:06:09:cb:e6:dc:6c:23:
    40:f7:00:58:49:cd:2d:bc:e6:cb:1f:2f:bd:0d:e8:
    ae:d3:9e:9d:52:de:23:22:d2:94:d6:98:84:fd:18:
    65:25:52:af:0d:4e:aa:9f:6d:2d:89:2b:3a:34:30:
    8c:ac:13:8a:f8:47:da:df:e9:1d:02:05:b9:9a:af:
    b1:34:38:77:aa:5d:04:6e:4e:6b:8a:8c:f2:c1:0f:
    a6:43:46:7c:b2:f2:1c:a1
prime1:
    00:e7:b6:9b:11:d7:38:27:16:42:2c:3c:df:1f:34:
    9b:9e:2b:88:77:77:08:6f:bc:42:7a:0b:69:7a:3b:
    b6:d2:e9:23:8f:8c:7b:75:82:e5:88:bd:5f:47:34:
    7b:1a:71:62:74:b6:f9:a0:ca:0b:bb:7c:4c:fb:2f:
    c5:81:96:7b:85
```

Step 2: Generate a Certificate Signing Request (CSR)

Once we have the key file, we generate a Certificate Signing Request (CSR), which basically includes the public key. The CSR will be sent to the CA, who will generate a certificate for the key.

```
[04/09/2020 19:01] Rakshith-10.0.2.15@VM:~/PKI$openssl req -new -key server.key -out server.csr -config openssl.cnf
Enter pass phrase for server.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:NY
Locality Name (eg, city) []:SYR
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SEED
Organizational Unit Name (eg, section) []:SEEDLABS
Common Name (e.g. server FQDN or YOUR name) []:SEEDWORKSHOP2020.com
Email Address []:rakshith2294@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:rak2294
An optional company name []:DEES
[04/09/2020 19:07] Rakshith-10.0.2.15@VM:~/PKI$
```

Step 3: Generating Certificates

The CSR file needs to have the CA's signature to form a certificate. In the real world, the CSR files are usually sent to a trusted CA for their signature. We will use our own trusted CA to generate certificates. The following command turns the certificate signing request (server.csr) into an X509 certificate (server.crt), using the CA's ca.crt and ca.key.

```
[04/09/2020 19:08] Rakshith-10.0.2.15@VM:~/PKI$openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key \-config openssl.cnf
Using configuration from openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4097 (0x1001)
        Validity
            Not Before: Apr  9 23:09:08 2020 GMT
            Not After : Apr  9 23:09:08 2021 GMT
        Subject:
            countryName               = US
            stateOrProvinceName       = NY
            localityName              = SYR
            organizationName          = SEED
            organizationalUnitName    = SEEDLABS
            commonName                = SEEDWORKSHOP2020.com
            emailAddress              = rakshith2294@gmail.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                48:50:BB:08:59:2E:66:33:4C:00:27:86:11:91:5C:92:5A:EA:71:3F
            X509v3 Authority Key Identifier:
                keyid:26:F0:34:FC:A0:7A:BA:3D:AA:AE:E7:AC:37:55:E4:14:74:CC:9C:E9

Certificate is to be certified until Apr  9 23:09:08 2021 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
[04/09/2020 19:09] Rakshith-10.0.2.15@VM:~/PKI$
```

Task 3: Deploying Certificate in an HTTPS Web Server

Step 1: Configuring DNS

To get our computers recognize this name, let us add the following entry to/etc/hosts; this entry basically maps the hostname SEEDWORKSHOP2020.com to our localhost.

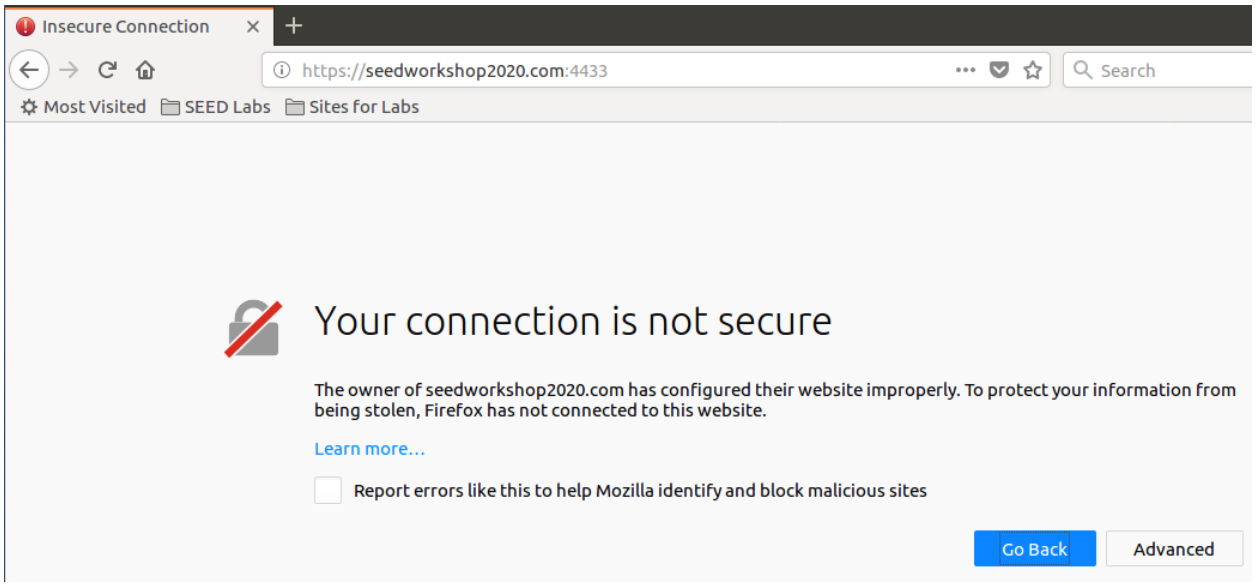
```
127.0.0.1       localhost
127.0.1.1       VM

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
127.0.0.1       SEEDWORKSHOP2020.com
```

Step 2: Configuring the web server

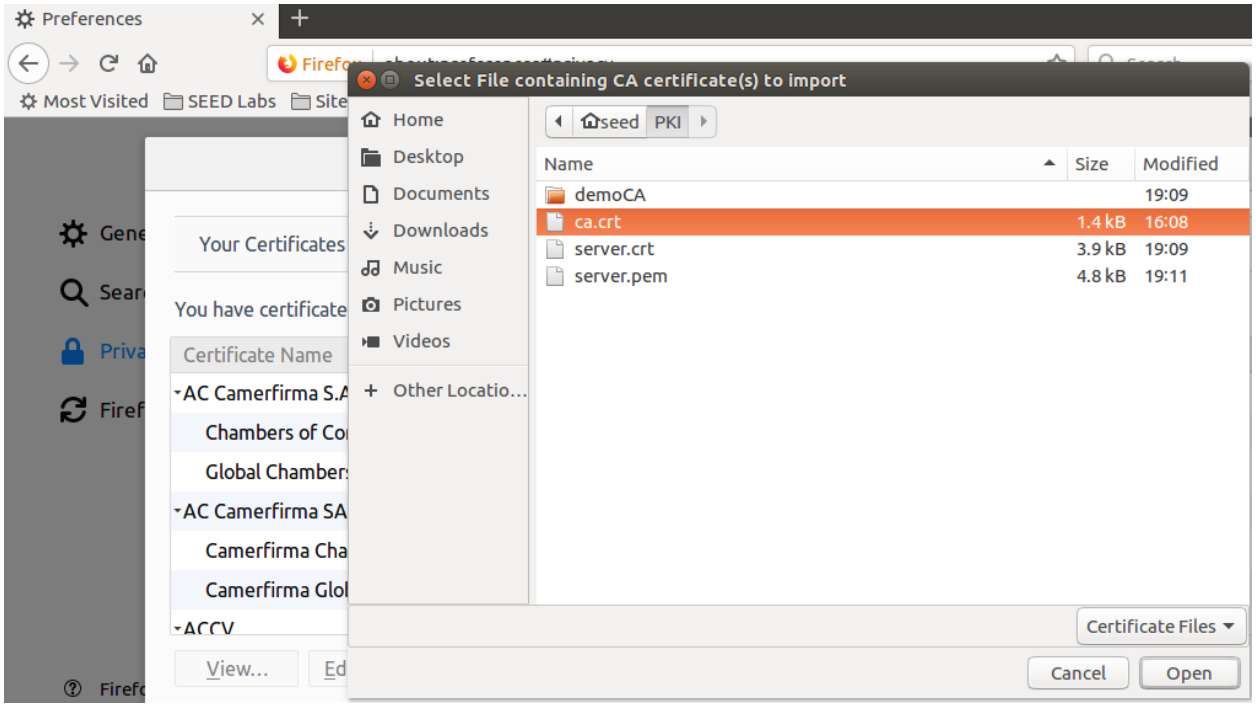
We are launching a simple web server with the certificate generated in the previous task. OpenSSL allows us to start a simple web server using the s_server command. By default, the server will listen on port 4433. We then try to load our page on port 4433, we get the following error because our browser does not recognize the certificate.

```
[04/09/2020 19:13] Rakshith-10.0.2.15@VM:~/PKI$openssl s_server -cert server.pem -www
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
```



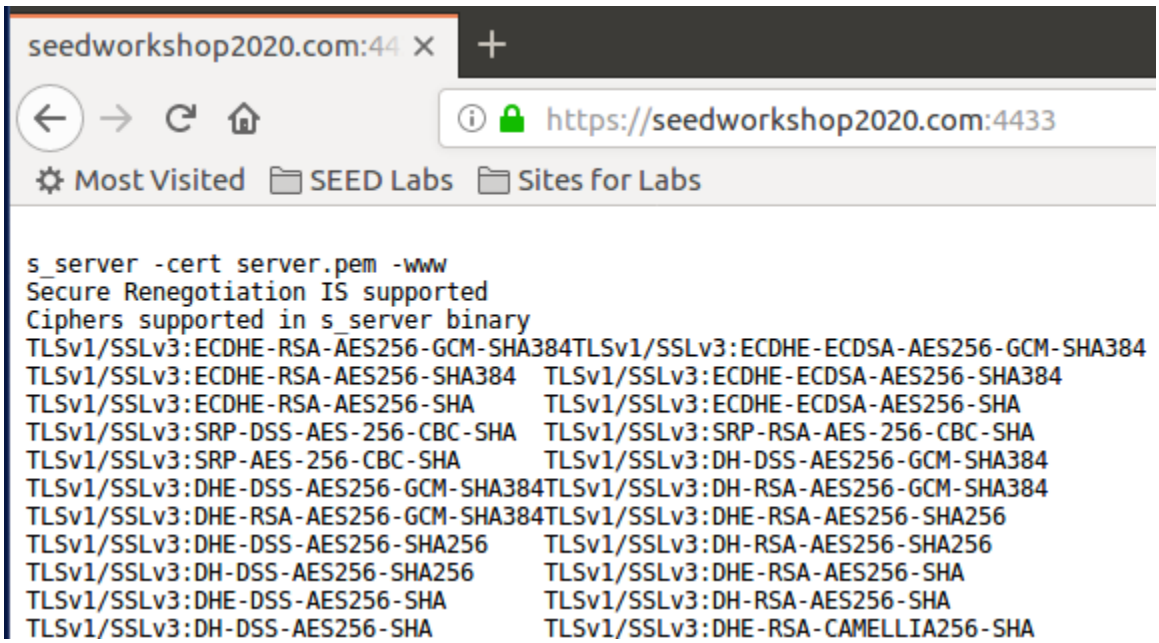
Step 3: Getting the browser to accept our CA certificate

We then make the browser accept our certificates, by importing the root CA that is ca.crt.



Step 4 testing our HTTPS website

Once the certificate is loaded into our browser we can see the content of the page.



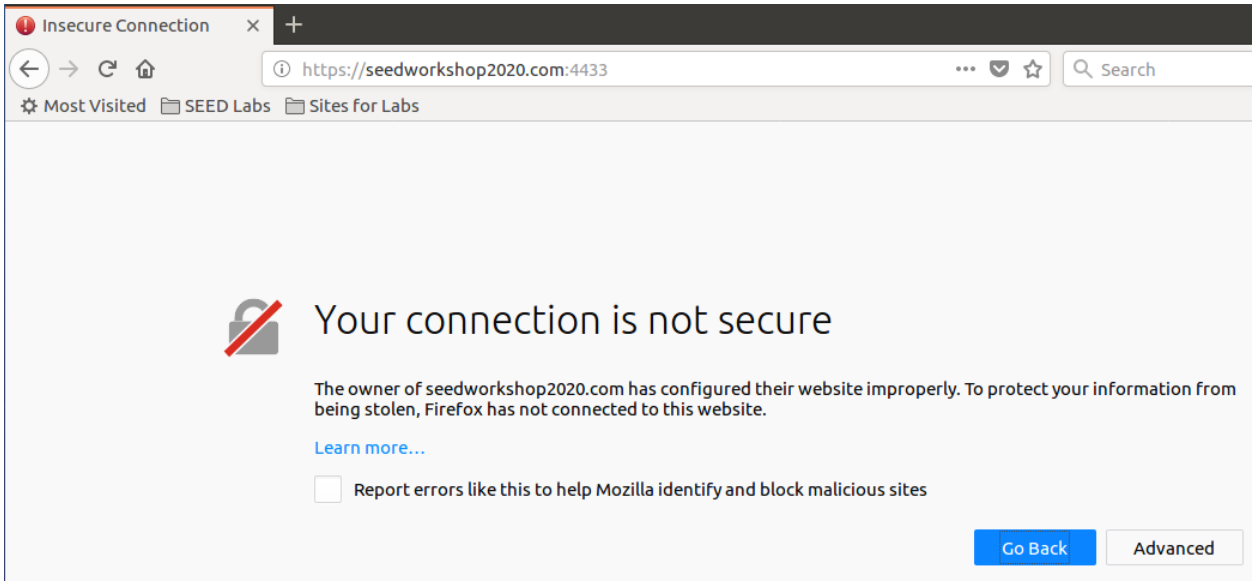
Step 4 Testing our HTTPS website

We modified one single bit of server.pem file and we can notice that our browser does not recognize this certificate, and output the below error.

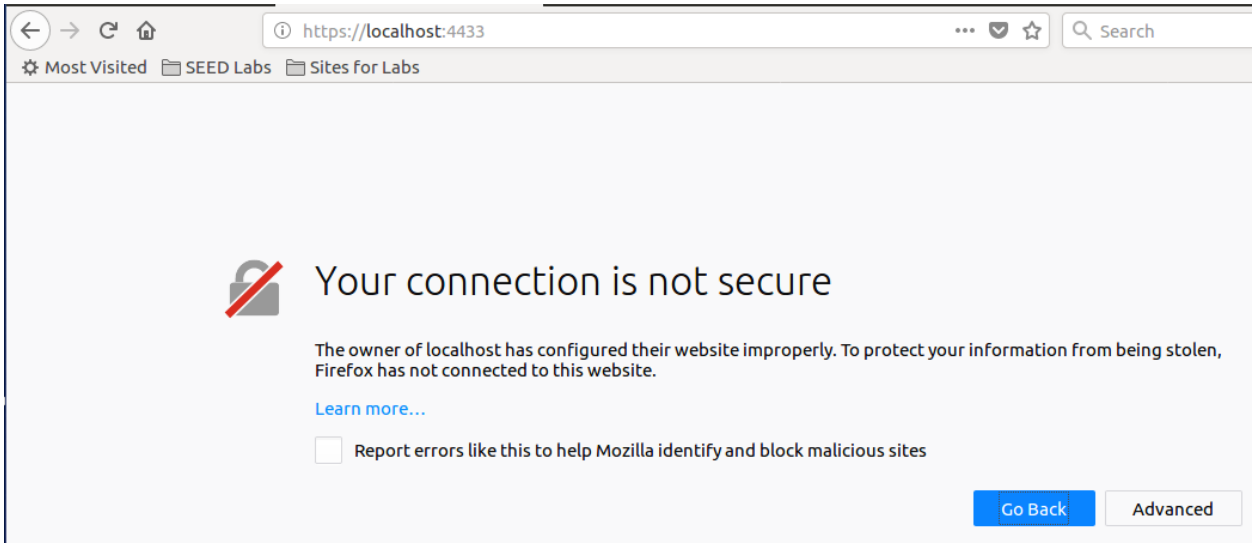
```
000001a4 64 6E 46 30 61 71 2B 4A 76 4B
000001c2 6A 45 44 47 46 38 4E 4D 38 39
000001e0 74 2B 41 46 4A 4F 65 33 45 2B
000001fe 70 69 43 61 62 42 4B 59 4C 4B
0000021c 6A 75 4A 6C 47 30 6A 43 54 50
0000023a 6E 35 31 54 78 47 64 71 4C 54
```

```
000001a4 64 6E 46 30 61 71 2B 4A
000001c2 6A 45 44 47 46 38 4E 4D
000001e0 74 2B 41 46 4A 4F 65 33
000001fe 70 69 44 61 62 42 4B 59
0000021c 6A 75 4A 6C 47 30 6A 43
```

```
[04/09/2020 19:23] Rakshith-10.0.2.15@VM:~/PKI$openssl s_server -cert server.pem -www
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
ACCEPT
```



SinceSEEDWORKSHOP2020.com points to the localhost, if we use https://localhost:4433 instead, we will be connecting to the same web server, but our page won't load because it does not recognize the certificate for localhost, hence we get the below error.



Task 4: Deploying Certificate in an Apache-Based HTTPS Website

We deploy our own webpage of SEEDWORKSHOP2020.com by creating a virtualhost in sites-enabled directory and default-conf file in /etc/apache2, and we add the following entries. Server_cert.pem is server certificate, server_key.pem is server key. We have created index.html file in a separate folder in /var/www. Once we restart the server and load our page we can see the content of our html file.

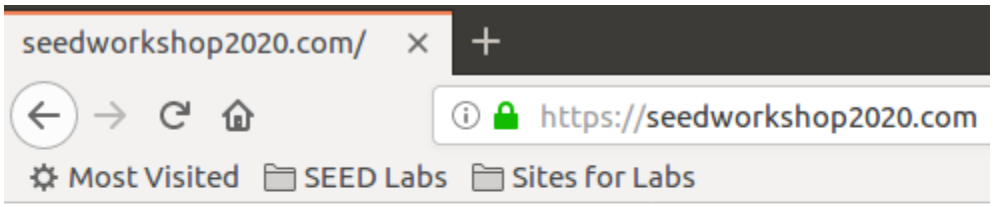
```
<VirtualHost *:443>
        ServerName SEEDWORKSHOP2020.com
        DocumentRoot /var/www/SEEDWORKSHOP2020_443
        DirectoryIndex index.html

        SSLEngine On
        SSLCertificateFile      /etc/apache2/ssl/server_cert.pem
        SSLCertificateKeyFile   /etc/apache2/ssl/server_key.pem
</VirtualHost>
```

```
[04/09/2020 20:55] Rakshith-10.0.2.15@VM:.../ssl$ls
server_cert.pem  server_key.pem
[04/09/2020 20:55] Rakshith-10.0.2.15@VM:.../ssl$
```

```
[04/09/2020 20:53] Rakshith-10.0.2.15@VM:.../www$cd SEEDWORKSHOP2020_443/
[04/09/2020 20:53] Rakshith-10.0.2.15@VM:.../SEEDWORKSHOP2020_443$ls
index.html
[04/09/2020 20:53] Rakshith-10.0.2.15@VM:.../SEEDWORKSHOP2020_443$
```

```
[04/09/2020 20:50] Rakshith-10.0.2.15@VM:.../sites-available$sudo apachectl configtest
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, u
s message
Syntax OK
[04/09/2020 20:50] Rakshith-10.0.2.15@VM:.../sites-available$sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
[04/09/2020 20:51] Rakshith-10.0.2.15@VM:.../sites-available$sudo a2ensite default-ssl
Site default-ssl already enabled
[04/09/2020 20:51] Rakshith-10.0.2.15@VM:.../sites-available$sudo service apache2 restart
Enter passphrase for SSL/TLS keys for SEEDWORKSHOP2020.com:443 (RSA): *******
[04/09/2020 20:51] Rakshith-10.0.2.15@VM:.../sites-available$
```



SEEDWORKSHOP2020

Assignment

Task 5: Launching a Man-In-The-Middle Attack

We are trying to launch the man-in-the-middle attack. We try to intercept communication between client and Syracuse.com using our own public key so that we can modify contents of the user and send it to real website, once we do that we will know the public key of Syracuse.com and we can generate a response to user using that.

Step 1: Setting up the malicious website.

Here, we generate the new page Syracuse.com (malicious), trying to intercept the connection and give fake information.

```
<VirtualHost *:443>
        ServerName syracuse.com
        DocumentRoot /var/www/SEEDWORKSHOP2020_443
        DirectoryIndex index.html

        SSLEngine On
        SSLCertificateFile      /etc/apache2/ssl/server_cert.pem
        SSLCertificateKeyFile   /etc/apache2/ssl/server_key.pem
</VirtualHost>
```

```
[04/09/2020 21:39] Rakshith-10.0.2.15@VM:.../sites-available$sudo apachectl configtest
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using
s message
Syntax OK
[04/09/2020 21:43] Rakshith-10.0.2.15@VM:.../sites-available$sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
[04/09/2020 21:43] Rakshith-10.0.2.15@VM:.../sites-available$sudo a2ensite default-ssl
Site default-ssl already enabled
[04/09/2020 21:43] Rakshith-10.0.2.15@VM:.../sites-available$sudo service apache2 restart
Enter passphrase for SSL/TLS keys for syracuse.com:443 (RSA): *******
[04/09/2020 21:44] Rakshith-10.0.2.15@VM:.../sites-available$
```

Step 2: Becoming the man in the middle

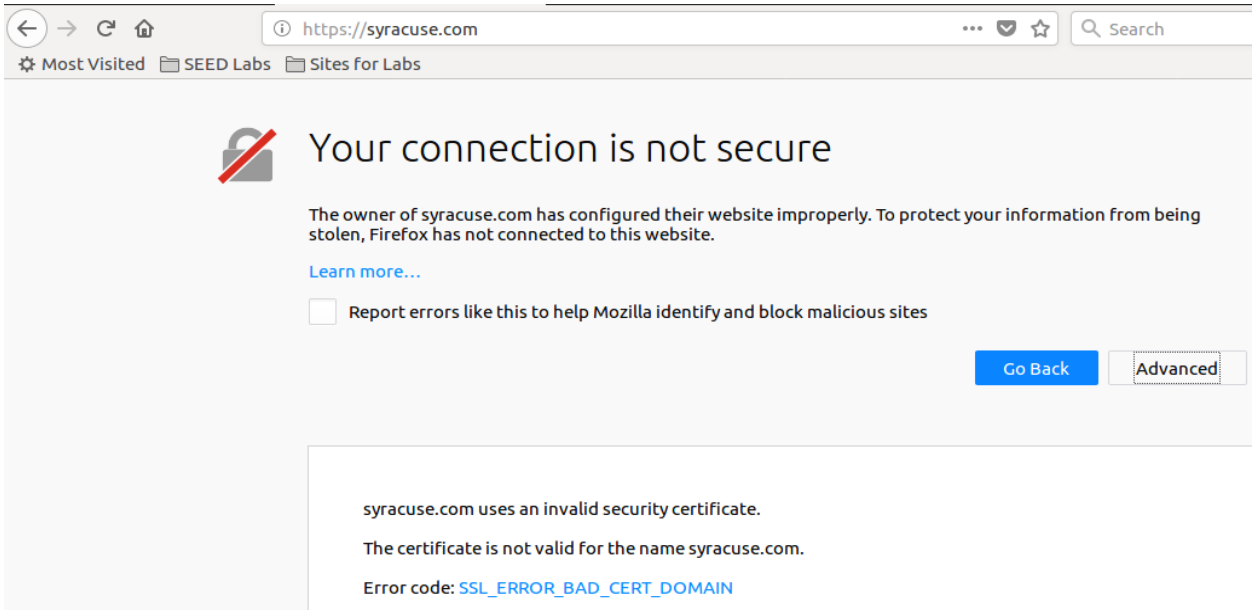
In order for us to accept all the request we have to become man-in-the middle we can do that by changing the DNS entries of Syracuse.com to our localhost so that we get all the requests.

```
[04/09/2020 21:41] Rakshith-10.0.2.15@VM:~/PKI$cat /etc/hosts
127.0.0.1       localhost
127.0.1.1       VM

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
127.0.0.1       syracuse.com
```

Step 3: Browse the target website

Once the user browses the website, our browser won't permit because the certificate we used for Syracuse.com was signed for a different domain, so access was not provided, and we see below warning.



2.6 Task 6: Launching a Man-In-The-Middle Attack with a Compromised CA

In order to accomplish the attack we generate a new certificate for Syracuse, but we can still use the old key. We get the certificate signed by the root, so that our browser can trust the certificate. Once we do this we can view the malicious content hosted in our virtualhost.

```
<VirtualHost *:443>
        ServerName syracuse.com
        DocumentRoot /var/www/html
        DirectoryIndex test.html

        SSLEngine On
        SSLCertificateFile      /etc/apache2/ssl/syracuse_cert.pem
        SSLCertificateKeyFile   /etc/apache2/ssl/syracuse_key.pem
</VirtualHost>


# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

```
[04/09/2020 22:36] Rakshith-10.0.2.15@VM:~/PKI$openssl req -new -key server.key -out server.csr -config openssl.cnf
Enter pass phrase for server.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:NY
Locality Name (eg, city) []:SYR
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SYRACUSE
Organizational Unit Name (eg, section) []:SU
Common Name (e.g. server FQDN or YOUR name) []:syracuse.com
Email Address []:rakshith2294@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:rak2294
An optional company name []:SYR
```

```
[04/09/2020 22:39] Rakshith-10.0.2.15@VM:~/PKI$openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key \-config openssl.cnf
Using configuration from openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4098 (0x1002)
        Validity
            Not Before: Apr 10 02:40:20 2020 GMT
            Not After : Apr 10 02:40:20 2021 GMT
        Subject:
            countryName               = US
            stateOrProvinceName       = NY
            localityName              = SYR
            organizationName          = SYRACUSE
            organizationalUnitName    = SU
            commonName                = syracuse.com
            emailAddress              = rakshith2294@gmail.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                48:50:BB:08:59:2E:66:33:4C:00:27:86:11:91:5C:92:5A:EA:71:3F
            X509v3 Authority Key Identifier:
                keyid:26:F0:34:FC:A0:7A:BA:3D:AA:AE:E7:AC:37:55:E4:14:74:CC:9C:E9

Certificate is to be certified until Apr 10 02:40:20 2021 GMT (365 days)
Sign the certificate? [y/n]:y
```

syracuse.com

Identity: syracuse.com
Verified by: RAK
Expires: 04/10/2021

Details

Subject Name

C (Country):	US
ST (State):	NY
L (Locality):	SYR
O (Organization):	SYRACUSE
OU (Organizational Unit):	SU
CN (Common Name):	syracuse.com
EMAIL (Email Address):	rakshith2294@gmail.com

Issuer Name

C (Country):	US
ST (State):	NY
L (Locality):	SYR
O (Organization):	SU
OU (Organizational Unit):	EECS
CN (Common Name):	RAK
EMAIL (Email Address):	rnallaha@syr.edu

Issued Certificate

Version:	3
Serial Number:	10 02
Not Valid Before:	2020-04-10
Not Valid After:	2021-04-10

Certificate Fingerprints