BIRZEIT UNIVERSITY

Faculty of Engineering and Technology

Master of Software Engineering (SWEN)

MSCE6342: NETWORK SECURITY PROTOCOLS **Second Semester 2021/2022**

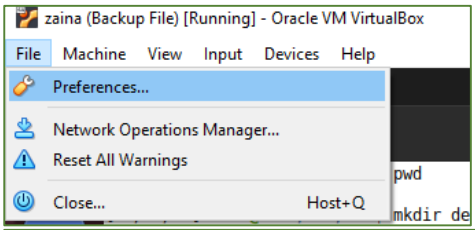
Student name: Zaina S. Abuabed **Instructor name: Dr. Ahmad S. Alsadeh**

Public Key Infrastructure (PKI) Lab

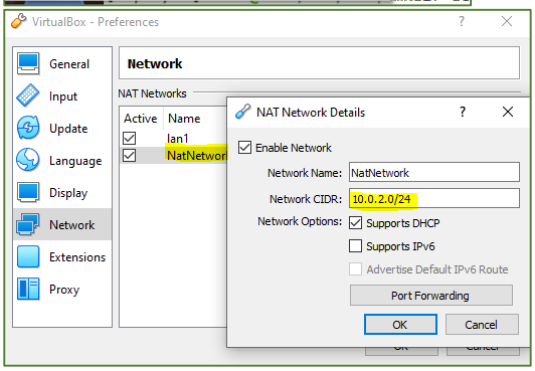
1. Lab Environment Setup:

➔ Network setup:

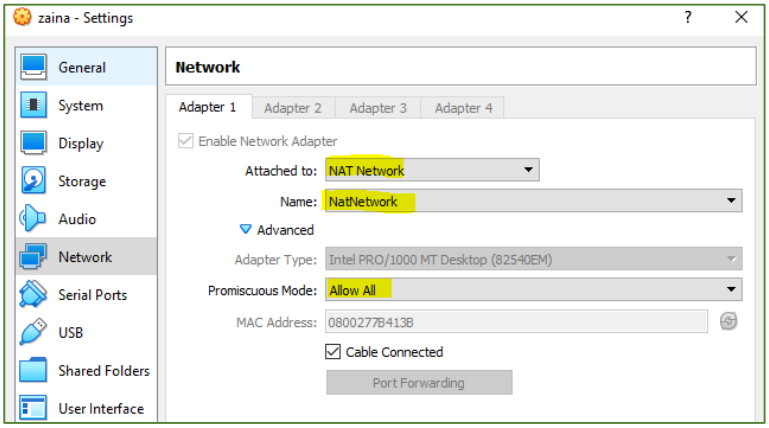
1) Before starting the machine: go to File → preferences



2) Create new NAT network as follow



3) Go to the machine settings and setup the network settings as follow:



➔ Docker setup:

1) Use the command docker-compose build to build the container image

```
[05/23/22]zaina@VM:~/.../Labsetup$ docker-compose build
Building web-server
Step 1/7 : FROM handsonsecurity/seed-server:apache-php
 ---> 2365d0ed3ad9
Step 2/7 : ARG WWWDIR=/var/www/bank32
 ---> Using cache
 ---> 4290251d0c0c
Step 3/7 : COPY ./index.html ./index_red.html $WWWDIR/
 ---> Using cache
 ---> 5346e945a1a5
Step 4/7 : COPY ./bank32_apache_ssl.conf /etc/apache2/sites-available
 ---> Using cache
 ---> 4e526e39f285
Step 5/7 : COPY ./certs/bank32.crt ./certs/bank32.key  /certs/
 ---> Using cache
 ---> acf6e81d436a
Step 6/7 : RUN  chmod 400 /certs/bank32.key      && chmod 644 $WWWDIR/index.html
      && chmod 644 $WWWDIR/index_red.html      && a2ensite bank32_apache_ssl
 ---> Using cache
 ---> 87c8f7cb152c
Step 7/7 : CMD  tail -f /dev/null
 ---> Using cache
 ---> 3d59b30b7bd4

Successfully built 3d59b30b7bd4
Successfully tagged seed-image-www-pki:latest
```

2) Use the command docker-compose up (alias→ dcup) to start the container

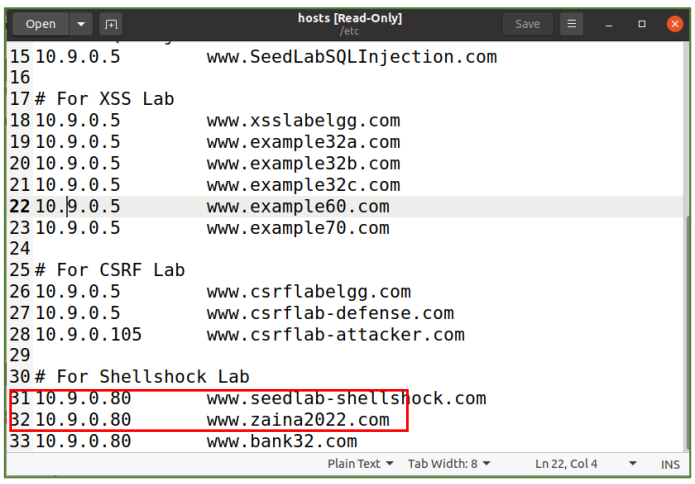
```
[05/23/22]zaina@VM:~/.../Labsetup$ dcup
Creating network "net-10.9.0.0" with the default driver
WARNING: Found orphan containers (oracle-10.9.0.80) for this project. If you rem
oved or renamed this service in your compose file, you can run this command with
 the --remove-orphans flag to clean it up.
Creating www-10.9.0.80 ... done
Attaching to www-10.9.0.80
```

3) Use the command dockps to get the running containers id name and ip

```
[05/22/22]zaina@VM:~/.../Labsetup$ dockps
84ef96c7b17a  www-10.9.0.80
```

4) For DNS setup, we update /etc/hosts file by mapping www.bank32.com name to the container ip, and the name www.zaina2022.com name to the same container ip as follow, then ping the alias name.

```
[05/22/22]zaina@VM:~/.../Labsetup$ sudo nano /etc/hosts
```




```
[05/22/22]zaina@VM:~/.../Labsetup$ ping bank32.com
PING bank32.com (34.102.136.180) 56(84) bytes of data.
64 bytes from 180.136.102.34.bc.googleusercontent.com (34.102.136.180): icmp_seq
=1 ttl=113 time=98.2 ms
64 bytes from 180.136.102.34.bc.googleusercontent.com (34.102.136.180): icmp_seq
=2 ttl=113 time=88.1 ms
64 bytes from 180.136.102.34.bc.googleusercontent.com (34.102.136.180): icmp_seq
=3 ttl=113 time=83.2 ms
64 bytes from 180.136.102.34.bc.googleusercontent.com (34.102.136.180): icmp_seq
=4 ttl=113 time=83.9 ms
64 bytes from 180.136.102.34.bc.googleusercontent.com (34.102.136.180): icmp_seq
=5 ttl=113 time=119 ms
^C
--- bank32.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 83.245/94.485/118.958/13.351 ms
```

2. Task 1: Becoming a Certificate Authority (CA)

➔ The default CA settings that we need to prepare:

```
[05/22/22]zaina@VM:~/.../PKI$ pwd
/home/zaina/Desktop/PKI
```
Prints the current working directory path, starting from the root (/)

```
[05/22/22]zaina@VM:~/.../PKI$ mkdir demoCA
[05/22/22]zaina@VM:~/.../PKI$ ls
demoCA  Labsetup
```
Create demoCA directory in the current directory and list the current directory content to ensure step correctiness

```
[05/22/22]zaina@VM:~/.../PKI$ cd demoCA
[05/22/22]zaina@VM:~/.../demoCA$ mkdir certs crl newcerts
[05/22/22]zaina@VM:~/.../demoCA$ ls
certs  crl  newcerts
```
Enter the demoCA folder and create three folders inside : certs, crl m and newcerts. then list the directory contents

```
[05/22/22]zaina@VM:~/.../demoCA$ echo 1000>serial
[05/22/22]zaina@VM:~/.../demoCA$ ls
certs  crl  newcerts  serial
```
Create the serial file which contains the current serial number =1000, then lit the directory contents.

```
[05/22/22]zaina@VM:~/.../demoCA$ gedit index.txt
[05/22/22]zaina@VM:~/.../demoCA$ ls
certs  crl  index.txt  newcerts  serial
```
Create the database file index.txt as an empty file, then list the directory contents

➔ The configuration File:

We need to copy the openssl.conf from /usr/lib/ssl/openssl.conf to PKI folder that we are working inside. So, we need to exit the demoCA directory, print the current path, then perform the copy command

```
[05/22/22]zaina@VM:~/.../demoCA$ cd
[05/22/22]zaina@VM:~/.../PKI$ pwd
/home/zaina/Desktop/PKI
[05/22/22]zaina@VM:~/.../PKI$ cp "/usr/lib/ssl/openssl.cnf" "/home/zaina/Desktop/PKI/"
```

➔ Certificate Authority (CA)

`$ openssl req -x509 -newkey rsa:4096 -sha256 -days 3560 -keyout ca.key -out ca.crt -config openssl.cnf`

This command is used to generate self-signed certificates for Certificate Authority. It authenticates the certificate by prompting information related to the certificate.

openssl commands meaning:	
req	Creates a signing request.
-newkey	Creates new requests.
-x509	Inspects signed certificate by loading x509 modules.
rsa:4096	RSA key processing command with 4096 key size
-sha256	Use SHA-2 256 Message Digest Commands
-days 3560	CA Valid for 10 years → 365*10
-keyout	Used to give a path to a filename, where it writes newly created private keys.
-out	Specifies the output file where the result will be stored.
-config	Configuration file to be specified.

After starting the command, a set of settings must be configured by adding the CA information to establish the certificate authority. Screenshot:

```
[05/22/22]zaina@VM:~/.../PKI$ openssl req -x509 -newkey rsa:4096 -sha256 -days 3560 -keyo
ut ca.key -out ca.crt -config openssl.cnf
Generating a 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]:PS
State or Province Name (full name) [Some-State]:Westbank
Locality Name (eg, city) []:Birzeit
Organization Name (eg, company) [Internet Widgits Pty Ltd]:BZU
Organizational Unit Name (eg, section) []:SWEN
Common Name (e.g. server FQDN or YOUR name) []:zaina
Email Address []:zaina.shtaiwi.zs@gmail.com
[05/22/22]zaina@VM:~/.../PKI$
```

password and verify password-->seed

2 letter country code PS for Palestine

Westbank for west bank state

birzeit --> city

BZU --> organization name

SWEN --> section within the organization

name and email

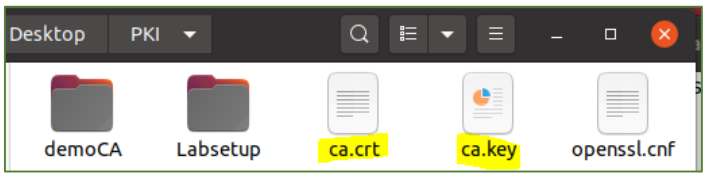
To view the decoded certificate, use the following command:

$ openssl x509 -in ca.crt -text -noout

```
[05/22/22]zaina@VM:~/.../PKI$ openssl x509 -in ca.crt -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            4e:9b:b2:2c:d7:98:25:01:c3:6e:1c:56:01:59:0b:62:4c:79:53:94
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = PS, ST = Westbank, L = Birzeit, O = BZU, OU = SWEN,
 CN = zaina, emailAddress = zaina.shtaiwi.zs@gmail.com
        Validity
            Not Before: May 22 22:49:07 2022 GMT
            Not After : May 13 22:49:07 2023 GMT
        Subject: C = PS, ST = Westbank, L = Birzeit, O = BZU, OU = SWEN
, CN = zaina, emailAddress = zaina.shtaiwi.zs@gmail.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (4096 bit)
                Modulus:
                    00:cb:42:36:09:33:3a:43:ee:ee:2e:0d:9c:29:df:
                    39:9b:b4:93:df:95:4e:96:bf:e7:ad:48:ae:05:a6:
                    a1:74:e1:42:75:ad:53:a0:4b:2b:51:d4:ec:c9:5a:
                    50:41:be:f6:5f:ae:c1:b4:ed:1e:0d:6b:81:a4:b6:
                    a8:70:bc:dc:df:8f:d8:47:6d:e2:a7:ce:60:96:21:
                    37:8f:8d:3a:9b:62:e6:35:ba:82:07:1e:7d:fe:f8:
                    35:27:86:6b:e1:fe:cd:fe:94:62:8d:35:c2:a8:d7:
                    4f:cd:54:4f:7b:f7:9b:2e:06:b9:64:d2:aa:b6:f1:
                    d4:6e:a3:82:9e:b4:0d:9f:11:8f:0f:fb:93:e6:6b:
                    43:61:16:b2:a8:34:ed:8c:be:eb:60:86:75:9f:a8:
                    74:29:69:ab:7f:11:30:85:a0:87:e6:fa:8f:15:1f:
                    50:e4:4d:d4:bb:29:8a:3c:7b:11:fe:bc:ab:6e:25:
                    09:61:55:6a:c3:71:1c:e9:74:d8:89:0a:4f:e5:80:
                    77:fd:55:fb:85:de:f4:da:fb:af:2f:03:4d:ae:1a:
                    51:62:a3:15:a8:51:a9:a1:36:7f:af:6c:76:44:eb:
                    0c:4f:2c:cb:02:03:9a:ec:1e:38:8c:7d:ff:15:96:
                    f9:70:02:f1:9f:d8:6d:60:d7:f1:5e:70:36:b4:27:
                    fe:aa:b0:6a:88:03:d6:0f:2c:b6:47:c6:f7:71:7a:
                    3b:4b:1c:f1:e8:b6:04:a2:0a:5a:2f:c3:cb:a6:81:
                    f5:36:7e:fa:38:1f:63:8d:fa:3e:86:87:1d:f0:84:
                    69:19:b4:a8:dc:cc:af:58:30:10:d0:18:6b:c1:c7:
                    c1:9a:6a:f9:cb:79:4e:e4:64:57:cf:b5:cd:87:f4:
                    41:37:9d:e0:1e:0b:49:7a:11:d7:7f:cc:86:fa:d8:
                    66:8f:5a:22:f1:52:98:65:e2:ad:99:5c:e6:5e:36:
                    49:e4:a0:be:7d:d0:92:4c:67:e7:d3:9d:23:85:ed:
                    ba:d8:e7:1d:6c:6d:54:f6:7f:c8:5f:b3:5c:1c:6e:
                    1f:42:56:6c:ae:35:9b:16:c9:3f:71:a9:db:0b:f2:
                    c6:7d:0a:2e:9c:c1:fa:c2:50:38:74:46:69:6d:c4:
                    2c:b0:f0:45:71:3b:82:89:a4:91:37:23:04:f2:27:
                    52:0a:71:25:4b:f1:e3:b0:6b:bd:ba:ad:a6:9e:48:
                    39:63:7b:f8:ef:75:9d:e2:b0:73:f1:c4:05:90:04:
                    80:5e:3a:bf:58:28:ec:9b:14:5d:a6:2b:a5:7e:48:
                    d7:d0:70:92:e9:07:84:17:f7:21:b6:38:08:2c:54:
                    cc:c2:0e:ff:ab:39:07:a9:42:18:39:c9:bf:f3:02:
                    48:48:e9
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                A9:ED:99:5A:7E:99:63:FB:A7:AD:8A:44:D6:9A:99:5D:19:AB:3
6:44
            X509v3 Authority Key Identifier:
                keyid:A9:ED:99:5A:7E:99:63:FB:A7:AD:8A:44:D6:9A:99:5D:1
9:AB:36:44

            X509v3 Basic Constraints: critical
                CA:TRUE
    Signature Algorithm: sha256WithRSAEncryption
         71:31:57:8a:eb:c1:3d:5f:38:3e:dd:c9:31:dd:e2:fd:f2:04:
         84:7b:9c:57:00:18:2e:81:8c:cd:ce:3c:69:8a:b1:96:b3:d4:
         57:0a:05:d0:87:5e:8a:07:8c:0d:2c:4c:a5:88:6b:c6:9b:a8:
         41:76:ac:51:83:f2:b9:9d:8c:94:a2:f2:a2:4c:b0:13:d6:3e:
         4f:cc:d2:b7:16:e9:69:60:2c:ae:27:fe:f2:19:c8:4c:68:8d:
         4d:48:75:3e:2f:f6:9d:98:68:47:e1:8c:41:7f:06:a2:75:64:
         90:40:9f:bf:81:32:80:ae:ad:2f:e4:c2:59:a5:52:0e:16:97:
         3c:f2:ac:b2:02:8a:ca:ba:af:f7:68:a5:14:24:cb:53:23:f3:
         f5:0c:17:fa:f1:8a:fa:5a:69:a0:a4:76:c9:a0:3f:cd:0c:fc:
         04:b8:b8:5d:56:2f:64:56:f1:e6:c9:5f:8d:84:d2:2f:d7:0f:
         39:46:b6:bf:cb:89:e1:5f:43:7e:54:d7:2c:91:d5:fe:5a:06:
         6d:80:30:db:03:c1:a5:98:e5:55:2a:b2:76:23:b5:8a:69:ca:
         b0:d6:98:e2:49:7c:a5:11:96:be:ca:af:32:fd:48:c2:2f:07:
         ea:d1:07:7d:17:5c:0e:65:b5:2a:35:35:98:8d:db:b7:ed:99:
         11:02:c8:51:2a:13:eb:fe:3e:38:1d:0a:8a:db:c8:0f:ce:37:
         6a:b5:f1:67:9c:40:74:16:95:72:4e:d7:f5:af:03:c1:8e:c9:
         d0:e5:94:a7:48:4f:70:1c:f6:fd:32:d6:3e:4b:e5:df:d9:25:
         cf:00:16:ba:8b:18:c7:b0:59:ba:13:7a:c9:ea:e7:72:f8:48:
         c5:45:95:70:01:1c:a9:c1:e3:46:29:83:35:55:02:f8:b9:09:
         f0:54:23:49:33:6e:74:18:f8:7e:41:74:21:a1:65:7d:6f:57:
         ac:c0:6b:13:09:f9:95:b0:81:1f:71:12:84:f6:b3:a7:66:42:
         d8:10:cb:4c:19:cb:89:b2:e4:6a:cc:3d:a1:99:df:0c:6d:8a:
         d2:f1:d7:ac:2a:9a:c0:25:71:a0:2f:17:2b:b0:11:2e:33:60:
         68:00:ce:72:d7:8b:12:bb:04:68:10:25:44:cd:fe:4e:71:c0:
         32:62:f2:91:19:44:cc:4b:33:6d:ac:2b:71:e5:1b:8c:81:cc:
         83:6f:3b:7d:2b:f1:82:6f:a7:f7:b0:89:f4:36:df:e4:04:e9:
         dd:c7:33:15:41:28:25:34:77:4f:56:9b:e3:a4:68:af:99:2a:
         62:96:c5:43:63:1e:9e:ee:55:fe:b5:90:c1:34:57:32:32:2b:
         33:88:c4:e6:9f:c0:c0:98
```

The output shows:



- ➔ The Signature Algorithm used
- ➔ The dates of certificate Validity
- ➔ The Public-Key bit length
- ➔ The Issuer, is the entity that signed the certificate
- ➔ The Subject, which refers to the certificate itself

To view the decoded RSA key, use the following command: rsa means Runs the RSA encryption algorithm while –text means prints out the certificate in the text format. –out means printed encrypted.

```
$ openssl rsa -in ca.key -text -noout
```

```
[05/22/22]zaina@VM:~/.../PKI$ openssl rsa -in ca.key -text -noout
Enter pass phrase for ca.key:
RSA Private-Key: (4096 bit, 2 primes)
modulus:
    00:cb:42:36:09:33:3a:43:ee:ee:2e:0d:9c:29:df:
    39:9b:b4:93:df:95:4e:96:bf:e7:ad:48:ae:05:a6:
    a1:74:e1:42:75:ad:53:a0:4b:2b:51:d4:ec:c9:5a:
    50:41:be:f6:5f:ae:c1:b4:ed:1e:0d:6b:81:a4:b6:
    a8:70:bc:dc:df:8f:d8:47:6d:e2:a7:ce:60:96:21:
    37:8f:8d:3a:9b:62:e6:35:ba:82:07:1e:7d:fe:f8:
    35:27:86:6b:e1:fe:cd:fe:94:62:8d:35:c2:a8:d7:
    4f:cd:54:4f:7b:f7:9b:2e:06:b9:64:d2:aa:b6:f1:
    d4:6e:a3:82:9e:b4:0d:9f:11:8f:0f:fb:93:e6:6b:
    43:61:16:b2:a8:34:ed:8c:be:eb:60:86:75:9f:a8:
    74:29:69:ab:7f:11:30:85:a0:87:e6:fa:8f:15:1f:
    50:e4:4d:d4:bb:29:8a:3c:7b:11:fe:bc:ab:6e:25:
    09:61:55:6a:c3:71:1c:e9:74:d8:89:0a:4f:e5:80:
    77:fd:55:fb:85:de:f4:da:fb:af:2f:03:4d:ae:1a:
    51:62:a3:15:a8:51:a9:a1:36:7f:af:6c:76:44:eb:
    0c:4f:2c:cb:02:03:9a:ec:1e:38:8c:7d:ff:15:96:
    f9:70:02:f1:9f:d8:6d:60:d7:f1:5e:70:36:b4:27:
    fe:aa:b0:6a:88:03:d6:0f:2c:b6:47:c6:f7:71:7a:
    3b:4b:1c:f1:e8:b6:04:a2:0a:5a:2f:c3:cb:a6:81:
    f5:36:7e:fa:38:1f:63:8d:fa:3e:86:87:1d:f0:84:
    69:19:b4:a8:dc:cc:af:58:30:10:d0:18:6b:c1:c7:
    c1:9a:6a:f9:cb:79:4e:e4:64:57:cf:b5:cd:87:f4:
    41:37:9d:e0:1e:0b:49:7a:11:d7:7f:cc:86:fa:d8:
    66:8f:5a:22:f1:52:98:65:e2:ad:99:5c:e6:5e:36:
    49:e4:a0:be:7d:d0:92:4c:67:e7:d3:9d:23:85:ed:
    ba:d8:e7:1d:6c:6d:54:f6:7f:c8:5f:b3:5c:1c:6e:
    1f:42:56:6c:ae:35:9b:16:c9:3f:71:a9:db:0b:f2:
    c6:7d:0a:2e:9c:c1:fa:c2:50:38:74:46:69:6d:c4:
    2c:b0:f0:45:71:3b:82:89:a4:91:37:23:04:f2:27:
    52:0a:71:25:4b:f1:e3:b0:6b:bd:ba:ad:a6:9e:48:
    39:63:7b:f8:ef:75:9d:e2:b0:73:f1:c4:05:90:04:
    80:5e:3a:bf:58:28:ec:9b:14:5d:a6:2b:a5:7e:48:
    d7:d0:70:92:e9:07:84:17:f7:21:b6:38:08:2c:54:
    cc:c2:0e:ff:ab:39:07:a9:42:18:39:c9:bf:f3:02:
    48:48:e9
publicExponent: 65537 (0x10001)
privateExponent:
    00:a0:78:a4:aa:bc:cc:0a:68:2f:99:22:5b:a1:5f:
    40:5f:22:c1:00:6b:23:81:b6:fe:d4:fb:25:91:06:
    8e:3f:f5:bb:ff:a6:18:f8:db:7f:d6:fa:70:fb:43:
    17:4b:e0:d5:28:93:93:11:21:87:71:76:e8:02:bf:
    c6:da:42:f1:6a:7b:69:78:73:53:d7:c2:02:d5:64:
    62:36:2e:7d:0f:c2:94:17:e5:28:d5:30:08:af:6d:
    0c:a2:28:e9:31:a2:76:c5:c0:26:d2:6f:98:40:ae:
    c4:ad:4a:1b:fa:b6:70:c8:74:bc:97:d2:0e:bf:a2:
    cb:61:92:2c:04:da:08:4b:06:94:d9:3c:78:f7:18:
    c6:38:77:b0:13:46:be:29:39:6e:21:65:e6:a5:67:
    5c:56:79:bb:d0:f5:24:46:cb:87:a0:1f:3f:29:a0:
    f0:bf:5e:87:4e:40:ac:87:b5:6e:6e:9c:4c:08:08:
    ab:90:31:dc:b9:72:ab:2d:d1:77:47:32:62:03:c2:
    5a:79:23:12:12:0e:05:68:81:fe:3b:f2:b2:b8:7a:
    c4:b8:48:23:48:21:38:f8:33:4b:b9:96:bb:05:0c:
    41:4a:9d:05:a5:90:6d:16:6f:81:9b:c1:b9:cb:34:
    c7:b7:dc:4b:21:34:c0:9e:0d:be:dd:1d:60:c2:f9:
    34:7f:ea:73:3c:0c:fd:f8:b1:48:40:04:6c:48:fd:
    9f:74:5a:b4:28:b9:d4:98:e6:70:fc:5a:37:56:d3:
    85:72:d6:73:d1:89:26:40:a1:55:88:a4:ed:b8:0d:
    86:59:5f:ae:14:1a:88:b3:34:6b:bb:65:55:a3:15:
    97:b1:39:ac:8f:a4:c9:bf:55:1f:34:e5:ce:aa:dd:
    aa:37:a6:e5:d7:f3:7d:0e:11:1b:1e:44:00:b8:af:
    8d:ba:79:64:7e:82:e0:47:4e:bd:53:50:70:24:07:
    39:0c:f7:06:8a:11:20:22:ff:e7:33:99:a9:bb:c1:
    36:2f:fe:01:8d:5b:d7:a3:f3:96:5c:82:a4:d9:61:
    71:65:89:f4:54:14:58:03:f9:9f:e2:36:26:ff:3c:
    cc:b0:78:9b:09:6f:ab:6f:f8:43:db:69:55:f4:1f:
    fb:72:ea:27:bc:16:a6:ac:f1:4e:9e:da:ff:02:cf:
    f1:48:03:8b:4d:c0:9a:05:db:36:91:3d:af:fd:b2:
    24:45:fe:58:16:e1:19:ff:63:7b:24:d2:6e:87:0e:
    33:55:14:9a:5d:d7:3f:63:f2:49:c9:51:65:9c:88:
    95:8d:bb:80:77:2c:f9:2d:47:af:0c:d9:07:6b:52:
    20:a1:7b:fd:af:65:29:81:1b:03:ed:23:d4:59:00:
    35:5c:81
prime1:
    00:f9:23:86:9f:ad:2e:c0:16:00:8b:a8:ac:a1:cb:
    d1:02:8b:b7:9f:01:c7:c9:3f:3f:82:8f:bf:45:75:
    10:d3:a4:c2:53:2f:dc:b4:d4:52:00:14:0b:d7:bf:
    8b:37:a7:b7:10:d0:d5:23:91:71:93:5f:cc:88:e1:
    81:31:b1:e6:fe:7d:f1:f5:4c:fc:28:cb:47:77:58:
    af:b8:e4:67:cc:d6:45:3a:4e:d5:fe:3e:eb:5b:60:
    47:13:6a:85:26:4d:93:bf:df:87:2e:cb:d7:c6:1d:
    ce:a1:e3:e5:f3:e0:66:b7:1d:74:69:36:2a:cf:0b:
    c2:65:9d:da:9a:dd:7a:4a:f8:9c:0e:b5:4c:72:c2:
    01:2e:a4:8d:38:32:92:a9:f5:3b:98:f2:82:46:fa:
    0a:ac:5c:e6:f5:1c:0c:2c:6e:f0:66:1d:50:66:2c:
    23:da:8b:41:67:2d:f5:d9:65:be:f7:e0:fd:42:aa:
    ad:40:2c:6f:d6:d3:bd:65:ae:50:83:c8:08:20:ba:
    89:cf:47:a8:7a:8c:f8:13:10:53:aa:dc:9a:12:c6:
    40:ba:df:4c:84:72:91:47:ec:db:0e:0f:8e:c4:c2:
    4a:0f:f9:48:7f:67:2a:41:30:de:2e:fb:2c:15:b8:
    bf:a3:e6:44:a9:a4:98:2e:48:af:d8:3a:ab:77:8b:
    22:b9
prime2:
    00:d0:db:38:c8:e8:f5:68:06:65:d0:ad:89:17:a0:
    23:4d:f2:6a:fc:f8:3d:b4:0b:d9:56:02:d0:f6:5e:
    c7:27:49:2b:89:98:c5:41:cd:5e:14:c3:a9:13:f5:
    77:0b:bd:31:08:0b:96:41:59:72:23:18:b6:0d:5c:
    15:98:63:10:55:a9:6d:d0:d2:48:a2:33:b8:88:66:
    c6:7a:44:b3:8f:85:f9:ae:65:c7:b0:7d:c7:89:8a:
    4c:19:a9:63:0c:2e:bf:1f:2c:d6:e2:01:63:6f:aa:
    ca:89:97:25:3a:e2:37:4e:d8:2d:b1:11:76:5b:7a:
    13:b8:25:ec:cd:c6:bb:dc:cf:54:0e:a5:14:a2:90:
    4e:20:67:c5:97:9e:4b:d6:b7:a9:18:6d:87:d3:9b:
    bc:2f:5c:ce:22:76:11:a8:c4:95:78:24:1c:30:a0:
    f3:a7:2f:e3:0f:05:7a:e9:60:b7:49:da:03:8d:b2:
    76:44:67:ba:12:20:b9:8a:36:33:dc:6e:7a:ac:8a:
    34:63:52:84:75:cc:c2:5a:53:39:6a:0d:b2:45:4a:
    c6:79:70:e0:be:22:6b:0a:ed:3b:04:6a:bf:4f:31:
    48:f6:05:31:2e:5a:cb:29:55:b8:67:78:c5:3d:fc:
    26:04:e2:28:4c:7a:bd:7f:77:ce:1a:bc:d7:db:f7:
    ff:b1
exponent1:
    00:ba:ee:59:f3:73:08:d1:dd:70:59:59:9c:f5:e5:
    2c:94:e1:6a:4b:e7:15:15:3d:dc:94:44:f0:f9:d7:
    85:61:97:8c:aa:c8:0d:ac:ac:66:3e:62:cc:23:fc:
    37:23:f2:36:1c:6a:e4:91:44:6e:d8:6a:17:d8:58:
    83:0c:5b:74:7b:5c:da:6f:f8:6c:76:db:57:6b:07:
    e3:09:32:6f:63:b5:1d:ca:4b:1e:ab:1e:60:8f:76:
    24:be:3f:bd:59:e8:f9:44:0d:1e:ac:11:0c:7c:3d:
    1f:2e:3f:05:e9:bf:64:b2:5e:6a:5b:0a:e6:81:e5:
    4b:dd:dc:dc:0a:38:c5:aa:7c:f0:f2:e4:3b:51:1b:
    99:b1:7a:8b:ef:2f:9a:f5:e4:e7:eb:b1:81:e8:69:
    72:1a:b3:b6:23:11:01:5d:51:7c:14:d2:47:e1:f1:
    9a:91:3e:cd:3a:f2:b4:35:82:ef:0f:a4:03:c2:c2:
    b7:2d:a0:69:e5:e1:87:89:44:8a:6a:e6:6a:9a:cb:
    fb:dc:c7:e2:74:d9:52:c5:53:09:c3:f8:b3:ae:04:
    8c:ca:ea:02:ca:99:ff:c1:dc:c8:db:ba:f6:9f:51:
    3c:2a:d2:1b:00:64:d2:c2:89:3d:1c:39:1f:71:c8:
    e6:19:bf:0b:a3:d1:b8:d1:79:7e:5f:1a:5e:4f:5a:
    d9:19
exponent2:
    00:b1:81:e0:5d:41:41:71:cc:1e:34:a1:88:e6:bf:
    7f:c1:fb:ca:da:7d:d1:53:78:ce:9a:bf:82:02:22:
    c3:78:ac:43:88:8a:82:fc:4e:23:2b:da:af:94:fb:
    06:d6:bf:51:30:d9:48:36:fa:62:7b:59:c1:40:d1:
    86:b9:21:58:fc:b5:27:54:57:a4:cb:d6:d1:7b:7c:
    7e:2b:e4:c0:07:b7:82:61:aa:b7:71:55:35:85:bc:
    38:b9:86:b1:5b:e5:4f:22:9b:e5:75:e8:60:2e:c4:
    25:a7:93:02:5d:6d:6e:3c:60:97:54:ea:d6:81:20:
    dc:29:a2:72:04:e2:67:40:60:25:bc:dc:a9:11:11:
    3c:95:66:1d:15:a3:c0:29:19:a7:69:cf:8d:a5:57:
    b3:35:3d:d7:24:8a:dc:af:67:84:f1:bb:1f:0a:65:
    81:30:13:0f:13:f1:40:69:a1:7d:a6:12:d8:fa:51:
    37:1b:24:5d:8c:92:ab:9e:c0:3a:8b:9a:8a:40:a2:
    a5:ab:d4:b2:27:bc:83:62:69:61:5e:b5:8d:52:3f:
    1f:38:8f:ab:33:6d:e5:34:40:33:ec:a9:1d:37:9d:
    57:b9:60:e8:4f:d1:72:e2:0a:ac:d3:a4:5d:5f:35:
    32:ca:eb:bb:fc:64:45:32:55:0c:a2:c4:e5:61:5b:
    01:c1
coefficient:
    00:eb:f2:78:18:b4:90:8b:ab:be:95:91:18:22:5b:
    40:a1:39:21:57:e0:04:53:50:9b:56:63:11:61:ee:
    21:b8:a8:39:3c:ad:65:c5:da:97:ed:70:91:d8:e4:
    ee:f7:e3:46:2a:10:a7:aa:d2:73:7a:3b:6b:ff:8f:
    7b:99:d0:8b:85:bf:9a:a5:f3:d0:31:07:6b:cb:c9:
    59:be:a6:58:4d:bc:b9:61:e2:58:90:87:e8:03:70:
    69:75:b0:52:5d:66:00:48:3e:54:01:8d:ab:cc:89:
    36:86:5d:3f:5e:dd:9e:56:cd:65:5d:05:88:81:e2:
    7d:e1:91:88:e2:16:5a:55:b3:31:61:b2:c0:d8:de:
    db:9d:a6:9e:30:d7:cb:18:e1:92:00:6a:f8:54:7a:
    34:79:9d:0a:72:d4:38:92:46:53:52:0d:62:14:38:
    ed:b1:07:72:f2:a8:e1:18:fe:8c:c0:31:d6:85:91:
    90:c2:5b:d7:84:c7:96:72:9d:11:33:a0:e5:33:31:
    67:a4:66:c7:7c:96:bb:18:45:8a:5c:e6:05:44:43:
    6a:d9:f7:13:8a:fc:2d:08:fe:e9:f5:81:6a:36:90:
    61:a7:c6:cd:ff:fe:a4:bb:47:08:e6:6b:af:62:df:
    99:b3:33:b3:d8:83:43:6b:53:da:b6:83:12:47:a0:
    62:b3
```

Q1 : What part of the certificate indicates this is a CA's certificate?

CA: True means that this is a certificate authority (CA) certificate. If CA:false then the certificate is for other entity.Screenshot:

```
X509v3 Basic Constraints: critical
    CA:TRUE
```

Q2: What part of the certificate indicates this is a self-signed certificate?

The *Issuer* and *Subject* are identical as the certificate is self-signed. Note that all root certificates are self-signed.

```
[05/22/22]zaina@VM:~/.../PKI$ openssl x509 -in ca.crt -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            4e:9b:b2:2c:d7:98:25:01:c3:6e:1c:56:01:59:0b:62:4c:79:53:94
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = PS, ST = Westbank, L = Birzeit, O = BZU, OU = SWEN, CN = zaina, emailAddress = zaina.sh
taiwi.zs@gmail.com
        Validity
            Not Before: May 22 22:49:07 2022 GMT
            Not After : May 13 22:49:07 2023 GMT
        Subject: C = PS, ST = Westbank, L = Birzeit, O = BZU, OU = SWEN, CN = zaina, emailAddress = zaina.s
htaiwi.zs@gmail.com
```

Q3: In the RSA algorithm, we have a public exponent e, a private exponent d, a modulus n, and two secret numbers p and q, such that n = pq. Please identify the values for these elements in your certificate and key files.

modulus → n

```
modulus:
    00:cb:42:36:09:33:3a:43:ee:ee:2e:0d:9c:29:df:
    39:9b:b4:93:df:95:4e:96:bf:e7:ad:48:ae:05:a6:
    a1:74:e1:42:75:ad:53:a0:4b:2b:51:d4:ec:c9:5a:
    50:41:be:f6:5f:ae:c1:b4:ed:1e:0d:6b:81:a4:b6:
    a8:70:bc:dc:df:8f:d8:47:6d:e2:a7:ce:60:96:21:
    37:8f:8d:3a:9b:62:e6:35:ba:82:07:1e:7d:fe:f8:
    35:27:86:6b:e1:fe:cd:fe:94:62:8d:35:c2:a8:d7:
    4f:cd:54:4f:7b:f7:9b:2e:06:b9:64:d2:aa:b6:f1:
    d4:6e:a3:82:9e:b4:0d:9f:11:8f:0f:fb:93:e6:6b:
    43:61:16:b2:a8:34:ed:8c:be:eb:60:86:75:9f:a8:
    74:29:69:ab:7f:11:30:85:a0:87:e6:fa:8f:15:1f:
    50:e4:4d:d4:bb:29:8a:3c:7b:11:fe:bc:ab:6e:25:
    09:61:55:6a:c3:71:1c:e9:74:d8:89:0a:4f:e5:80:
    77:fd:55:fb:85:de:f4:da:fb:af:2f:03:4d:ae:1a:
    51:62:a3:15:a8:51:a9:a1:36:7f:af:6c:76:44:eb:
    0c:4f:2c:cb:02:03:9a:ec:1e:38:8c:7d:ff:15:96:
    f9:70:02:f1:9f:d8:6d:60:d7:f1:5e:70:36:b4:27:
    fe:aa:b0:6a:88:03:d6:0f:2c:b6:47:c6:f7:71:7a:
    3b:4b:1c:f1:e8:b6:04:a2:0a:5a:2f:c3:cb:a6:81:
    f5:36:7e:fa:38:1f:63:8d:fa:3e:86:87:1d:f0:84:
    69:19:b4:a8:dc:cc:af:58:30:10:d0:18:6b:c1:c7:
    c1:9a:6a:f9:cb:79:4e:e4:64:57:cf:b5:cd:87:f4:
    41:37:9d:e0:1e:0b:49:7a:11:d7:7f:cc:86:fa:d8:
    66:8f:5a:22:f1:52:98:65:e2:ad:99:5c:e6:5e:36:
    49:e4:a0:be:7d:d0:92:4c:67:e7:d3:9d:23:85:ed:
    ba:d8:e7:1d:6c:6d:54:f6:7f:c8:5f:b3:5c:1c:6e:
    1f:42:56:6c:ae:35:9b:16:c9:3f:71:a9:db:0b:f2:
    c6:7d:0a:2e:9c:c1:fa:c2:50:38:74:46:69:6d:c4:
    2c:b0:f0:45:71:3b:82:89:a4:91:37:23:04:f2:27:
    52:0a:71:25:4b:f1:e3:b0:6b:bd:ba:ad:a6:9e:48:
    39:63:7b:f8:ef:75:9d:e2:b0:73:f1:c4:05:90:04:
    80:5e:3a:bf:58:28:ec:9b:14:5d:a6:2b:a5:7e:48:
    d7:d0:70:92:e9:07:84:17:f7:21:b6:38:08:2c:54:
    cc:c2:0e:ff:ab:39:07:a9:42:18:39:c9:bf:f3:02:
    48:48:e9
```

publicExponent → e	publicExponent: 65537 (0x10001)
privateExponent → d	privateExponent: 00:a0:78:a4:aa:bc:cc:0a:68:2f:99:22:5b:a1:5f: 40:5f:22:c1:00:6b:23:81:b6:fe:d4:fb:25:91:06: 8e:3f:f5:bb:ff:a6:18:f8:db:7f:d6:fa:70:fb:43: 17:4b:e0:d5:28:93:93:11:21:87:71:76:e8:02:bf: c6:da:42:f1:6a:7b:69:78:73:53:d7:c2:02:d5:64: 62:36:2e:7d:0f:c2:94:17:e5:28:d5:30:08:af:6d: 0c:a2:28:e9:31:a2:76:c5:c0:26:d2:6f:98:40:ae: c4:ad:4a:1b:fa:b6:70:c8:74:bc:97:d2:0e:bf:a2: cb:61:92:2c:04:da:08:4b:06:94:d9:3c:78:f7:18: c6:38:77:b0:13:46:be:29:39:6e:21:65:e6:a5:67: 5c:56:79:bb:d0:f5:24:46:cb:87:a0:1f:3f:29:a0: f0:bf:5e:87:4e:40:ac:87:b5:6e:6e:9c:4c:08:08: ab:90:31:dc:b9:72:ab:2d:d1:77:47:32:62:03:c2: 5a:79:23:12:12:0e:05:68:81:fe:3b:f2:b2:b8:7a: c4:b8:48:23:48:21:38:f8:33:4b:b9:96:bb:05:0c: 41:4a:9d:05:a5:90:6d:16:6f:81:9b:c1:b9:cb:34: c7:b7:dc:4b:21:34:c0:9e:0d:be:dd:1d:60:c2:f9: 34:7f:ea:73:3c:0c:fd:f8:b1:48:40:04:6c:48:fd: 9f:74:5a:b4:28:b9:d4:98:e6:70:fc:5a:37:56:d3: 85:72:d6:73:d1:89:26:40:a1:55:88:a4:ed:b8:0d: 86:59:5f:ae:14:1a:88:b3:34:6b:bb:65:55:a3:15: 97:b1:39:ac:8f:a4:c9:bf:55:1f:34:e5:ce:aa:dd: aa:37:a6:e5:d7:f3:7d:0e:11:1b:1e:44:00:b8:af: 8d:ba:79:64:7e:82:e0:47:4e:bd:53:50:70:24:07: 39:0c:f7:06:8a:11:20:22:ff:e7:33:99:a9:bb:c1: 36:2f:fe:01:8d:5b:d7:a3:f3:96:5c:82:a4:d9:61: 71:65:89:f4:54:14:58:03:f9:9f:e2:36:26:ff:3c: cc:b0:78:9b:09:6f:ab:6f:f8:43:db:69:55:f4:1f: fb:72:ea:27:bc:16:a6:ac:f1:4e:9e:da:ff:02:cf: f1:48:03:8b:4d:c0:9a:05:db:36:91:3d:af:fd:b2: 24:45:fe:58:16:e1:19:ff:63:7b:24:d2:6e:87:0e: 33:55:14:9a:5d:d7:3f:63:f2:49:c9:51:65:9c:88: 95:8d:bb:80:77:2c:f9:2d:47:af:0c:d9:07:6b:52: 20:a1:7b:fd:af:65:29:81:1b:03:ed:23:d4:59:00: 35:5c:81
prime1 → p	prime1: 00:f9:23:86:9f:ad:2e:c0:16:00:8b:a8:ac:a1:cb: d1:02:8b:b7:9f:01:c7:c9:3f:3f:82:8f:bf:45:75: 10:d3:a4:c2:53:2f:dc:b4:d4:52:00:14:0b:d7:bf: 8b:37:a7:b7:10:d0:d5:23:91:71:93:5f:cc:88:e1: 81:31:b1:e6:fe:7d:f1:f5:4c:fc:28:cb:47:77:58: af:b8:e4:67:cc:d6:45:3a:4e:d5:fe:3e:eb:5b:60: 47:13:6a:85:26:4d:93:bf:df:87:2e:cb:d7:c6:1d: ce:a1:e3:e5:f3:e0:66:b7:1d:74:69:36:2a:cf:0b: c2:65:9d:da:9a:dd:7a:4a:f8:9c:0e:b5:4c:72:c2: 01:2e:a4:8d:38:32:92:a9:f5:3b:98:f2:82:46:fa: 0a:ac:5c:e6:f5:1c:0c:2c:6e:f0:66:1d:50:66:2c: 23:da:8b:41:67:2d:f5:d9:65:be:f7:e0:fd:42:aa: ad:40:2c:6f:d6:d3:bd:65:ae:50:83:c8:08:20:ba: 89:cf:47:a8:7a:8c:f8:13:10:53:aa:dc:9a:12:c6: 40:ba:df:4c:84:72:91:47:ec:db:0e:0f:8e:c4:c2: 4a:0f:f9:48:7f:67:2a:41:30:de:2e:fb:2c:15:b8: bf:a3:e6:44:a9:a4:98:2e:48:af:d8:3a:ab:77:8b: 22:b9

prime2	→ q	prime2: 00:d0:db:38:c8:e8:f5:68:06:65:d0:ad:89:17:a0: 23:4d:f2:6a:fc:f8:3d:b4:0b:d9:56:02:d0:f6:5e: c7:27:49:2b:89:98:c5:41:cd:5e:14:c3:a9:13:f5: 77:0b:bd:31:08:0b:96:41:59:72:23:18:b6:0d:5c: 15:98:63:10:55:a9:6d:d0:d2:48:a2:33:b8:88:66: c6:7a:44:b3:8f:85:f9:ae:65:c7:b0:7d:c7:89:8a: 4c:19:a9:63:0c:2e:bf:1f:2c:d6:e2:01:63:6f:aa: ca:89:97:25:3a:e2:37:4e:d8:2d:b1:11:76:5b:7a: 13:b8:25:ec:cd:c6:bb:dc:cf:54:0e:a5:14:a2:90: 4e:20:67:c5:97:9e:4b:d6:b7:a9:18:6d:87:d3:9b: bc:2f:5c:ce:22:76:11:a8:c4:95:78:24:1c:30:a0: f3:a7:2f:e3:0f:05:7a:e9:60:b7:49:da:03:8d:b2: 76:44:67:ba:12:20:b9:8a:36:33:dc:6e:7a:ac:8a: 34:63:52:84:75:cc:c2:5a:53:39:6a:0d:b2:45:4a: c6:79:70:e0:be:22:6b:0a:ed:3b:04:6a:bf:4f:31: 48:f6:05:31:2e:5a:cb:29:55:b8:67:78:c5:3d:fc: 26:04:e2:28:4c:7a:bd:7f:77:ce:1a:bc:d7:db:f7: ff:b1
exponent1	→ d mod (p-1)	exponent1: 00:ba:ee:59:f3:73:08:d1:dd:70:59:59:9c:f5:e5: 2c:94:e1:6a:4b:e7:15:15:3d:dc:94:44:f0:f9:d7: 85:61:97:8c:aa:c8:0d:ac:ac:66:3e:62:cc:23:fc: 37:23:f2:36:1c:6a:e4:91:44:6e:d8:6a:17:d8:58: 83:0c:5b:74:7b:5c:da:6f:f8:6c:76:db:57:6b:07: e3:09:32:6f:63:b5:1d:ca:4b:1e:ab:1e:60:8f:76: 24:be:3f:bd:59:e8:f9:44:0d:1e:ac:11:0c:7c:3d: 1f:2e:3f:05:e9:bf:64:b2:5e:6a:5b:0a:e6:81:e5: 4b:dd:dc:dc:0a:38:c5:aa:7c:f0:f2:e4:3b:51:1b: 99:b1:7a:8b:ef:2f:9a:f5:e4:e7:eb:b1:81:e8:69: 72:1a:b3:b6:23:11:01:5d:51:7c:14:d2:47:e1:f1: 9a:91:3e:cd:3a:f2:b4:35:82:ef:0f:a4:03:c2:c2: b7:2d:a0:69:e5:e1:87:89:44:8a:6a:e6:6a:9a:cb: fb:dc:c7:e2:74:d9:52:c5:53:09:c3:f8:b3:ae:04: 8c:ca:ea:02:ca:99:ff:c1:dc:c8:db:ba:f6:9f:51: 3c:2a:d2:1b:00:64:d2:c2:89:3d:1c:39:1f:71:c8: e6:19:bf:0b:a3:d1:b8:d1:79:7e:5f:1a:5e:4f:5a: d9:19
exponent2	→ d mod (q-1)	exponent2: 00:b1:81:e0:5d:41:41:71:cc:1e:34:a1:88:e6:bf: 7f:c1:fb:ca:da:7d:d1:53:78:ce:9a:bf:82:02:22: c3:78:ac:43:88:8a:82:fc:4e:23:2b:da:af:94:fb: 06:d6:bf:51:30:d9:48:36:fa:62:7b:59:c1:40:d1: 86:b9:21:58:fc:b5:27:54:57:a4:cb:d6:d1:7b:7c: 7e:2b:e4:c0:07:b7:82:61:aa:b7:71:55:35:85:bc: 38:b9:86:b1:5b:e5:4f:22:9b:e5:75:e8:60:2e:c4: 25:a7:93:02:5d:6d:6e:3c:60:97:54:ea:d6:81:20: dc:29:a2:72:04:e2:67:40:60:25:bc:dc:a9:11:11: 3c:95:66:1d:15:a3:c0:29:19:a7:69:cf:8d:a5:57: b3:35:3d:d7:24:8a:dc:af:67:84:f1:bb:1f:0a:65: 81:30:13:0f:13:f1:40:69:a1:7d:a6:12:d8:fa:51: 37:1b:24:5d:8c:92:ab:9e:c0:3a:8b:9a:8a:40:a2: a5:ab:d4:b2:27:bc:83:62:69:61:5e:b5:8d:52:3f: 1f:38:8f:ab:33:6d:e5:34:40:33:ec:a9:1d:37:9d: 57:b9:60:e8:4f:d1:72:e2:0a:ac:d3:a4:5d:5f:35: 32:ca:eb:bb:fc:64:45:32:55:0c:a2:c4:e5:61:5b: 01:c1
coefficient	→ (inverse of q) mod p	coefficient: 00:eb:f2:78:18:b4:90:8b:ab:be:95:91:18:22:5b: 40:a1:39:21:57:e0:04:53:50:9b:56:63:11:61:ee: 21:b8:a8:39:3c:ad:65:c5:da:97:ed:70:91:d8:e4: ee:f7:e3:46:2a:10:a7:aa:d2:73:7a:3b:6b:ff:8f: 7b:99:d0:8b:85:bf:9a:a5:f3:d0:31:07:6b:cb:c9: 59:be:a6:58:4d:bc:b9:61:e2:58:90:87:e8:03:70: 69:75:b0:52:5d:66:00:48:3e:54:01:8d:ab:cc:89: 36:86:5d:3f:5e:dd:9e:56:cd:65:5d:05:88:81:e2: 7d:e1:91:88:e2:16:5a:55:b3:31:61:b2:c0:d8:de: db:9d:a6:9e:30:d7:cb:18:e1:92:00:6a:f8:54:7a: 34:79:9d:0a:72:d4:38:92:46:53:52:0d:62:14:38: ed:b1:07:72:f2:a8:e1:18:fe:8c:c0:31:d6:85:91: 90:c2:5b:d7:84:c7:96:72:9d:11:33:a0:e5:33:31: 67:a4:66:c7:7c:96:bb:18:45:8a:5c:e6:05:44:43: 6a:d9:f7:13:8a:fc:2d:08:fe:e9:f5:81:6a:36:90: 61:a7:c6:cd:ff:fe:a4:bb:47:08:e6:6b:af:62:df: 99:b3:33:b3:d8:83:43:6b:53:da:b6:83:12:47:a0: 62:b3

Task 2: Generating a Certificate Request for Your Web Server

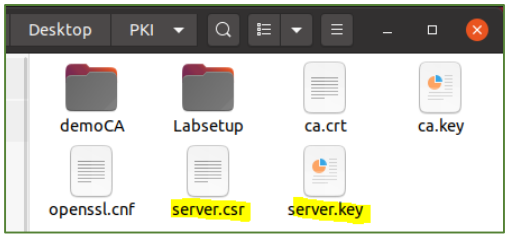
1) Generate CRS for personal web page

In this task, we want to generate a certificate signing request (CSR)for our company. In the docker configuration, we add an alias for the container which is “www.zaina2022.com” which represents a fake company server that we wish to request the certificate for.

```
$openssl req -newkey rsa:2048 -sha256  -keyout server.key -out server.csr  -subj
"/CN=www.zaina2022.com/O=ZA Inc./C=PS"  -passout pass:seed
```

Screenshot:

```
[05/23/22]zaina@VM:~/.../PKI$ openssl req -newkey rsa:2048 -sha256  -keyout
 server.key -out server.csr  -subj "/CN=www.zaina2022.com/O=ZA Inc./C=PS"
-passout pass:seed
Generating a RSA private key
.....................................................................+++++
.................................+++++
writing new private key to 'server.key'
-----
```



The output is a pair of public and private key. To view the decoded CSR we use the following command:

```
$ openssl req -in server.csr -text -noout
```

```
[05/23/22]zaina@VM:~/.../PKI$ openssl req -in server.csr -text -noout
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: CN = www.zaina2022.com, O = ZA Inc., C = PS
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:a1:f0:ac:e7:f8:26:e5:38:33:6d:d1:7e:0b:3d:
                    ac:ec:5c:7c:37:54:bb:88:94:a2:39:d9:54:6d:58:
                    94:15:5b:2f:e5:12:5f:d4:64:e2:d3:1b:42:ce:09:
                    b6:de:12:47:58:f5:7a:28:16:05:d1:75:00:c6:8f:
                    65:c6:8d:49:4d:ab:35:18:db:b8:36:4d:5a:ce:60:
                    bc:43:fd:a3:d3:7d:7b:b3:15:ad:56:71:62:ac:fd:
                    70:f7:76:04:ae:74:00:07:05:e9:92:01:ee:75:30:
                    09:d3:95:bb:91:2c:51:ce:ad:e1:8f:b7:7d:8f:af:
                    2e:e5:81:af:e9:ac:5f:ea:2e:4d:27:96:b9:e6:ce:
                    0d:8a:d6:bc:e0:c4:bf:9f:27:53:95:5c:4b:28:b4:
                    52:73:12:23:20:0a:7a:70:93:2e:87:d2:84:ed:10:
                    3b:3d:da:10:cd:59:4f:06:be:38:68:0c:44:30:51:
                    14:bc:84:0e:4e:f8:c6:a5:a8:33:9e:2b:f4:e8:52:
                    ed:f3:a5:5f:fc:d4:ec:7a:cd:01:40:de:48:7a:a4:
                    35:12:33:2a:22:26:10:5d:23:49:94:32:d3:70:cf:
                    57:71:f8:e8:c5:51:80:7a:30:77:3f:d7:33:a8:28:
                    0c:39:5f:67:ce:74:ae:5a:6a:40:b7:fd:c7:40:b9:
                    44:ff
                Exponent: 65537 (0x10001)
        Attributes:
            a0:00
    Signature Algorithm: sha256WithRSAEncryption
         7a:a3:10:da:62:f7:24:da:b4:72:c3:73:06:57:af:f4:92:52:
         ec:e0:b9:4e:9b:ab:a6:89:e1:e8:b7:d0:b7:37:b7:fb:ea:11:
         61:9c:c6:5c:0f:34:d1:98:ca:ce:bb:8e:8f:eb:cf:9b:30:b0:
         80:2e:f9:e0:2b:b8:7e:1b:c1:9b:d4:bf:34:fc:1a:05:43:55:
         2a:bd:62:8c:ed:6c:b1:a2:46:39:b3:45:c0:1e:aa:82:99:a0:
         60:41:07:40:a8:c3:a0:38:42:aa:b2:88:3b:1f:3a:69:58:ef:
         da:cf:29:93:2d:ba:5a:29:d0:23:1f:74:df:b6:f9:40:df:40:
         d1:64:64:7f:fb:26:4a:4f:b3:5b:17:bb:40:c7:7e:27:0d:17:
         d7:5f:d3:34:a5:4e:97:19:a1:f9:86:8f:34:76:3b:e4:30:cf:
         b5:85:a9:19:54:ea:6a:db:98:9d:53:8f:7e:65:44:91:3b:bc:
         a7:64:c3:af:74:ec:95:2c:a8:eb:ec:cb:6d:3d:2a:27:62:c6:
         8a:4c:85:e8:1d:2b:46:94:ec:f7:71:0a:dd:bf:39:30:bd:f8:
         57:8c:44:73:c2:c0:aa:73:e6:f1:57:4f:74:5e:d0:08:cc:cf:
         07:6e:f8:dd:77:b0:db:36:44:9e:9a:e1:75:85:cc:84:38:37:
         8e:34:35:b8
```

To view the decoded private key, we use the following command:

$openssl rsa -in server.key -text -noout

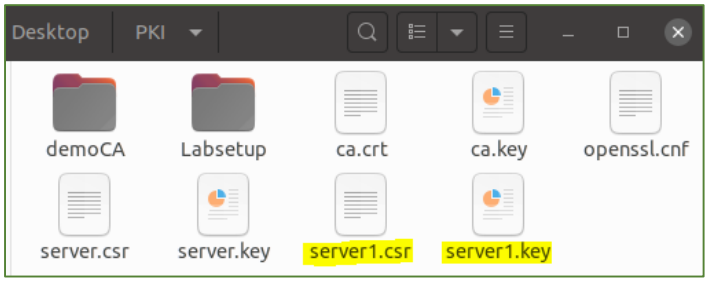
```
[05/23/22]zaina@VM:~/.../PKI$ openssl rsa -in server.key -text -noout
Enter pass phrase for server.key:
RSA Private-Key: (2048 bit, 2 primes)
modulus:
    00:a1:f0:ac:e7:f8:26:e5:38:33:6d:d1:7e:0b:3d:
    ac:ec:5c:7c:37:54:bb:88:94:a2:39:d9:54:6d:58:
    94:15:5b:2f:e5:12:5f:d4:64:e2:d3:1b:42:ce:09:
    b6:de:12:47:58:f5:7a:28:16:05:d1:75:00:c6:8f:
    65:c6:8d:49:4d:ab:35:18:db:b8:36:4d:5a:ce:60:
    bc:43:fd:a3:d3:7d:7b:b3:15:ad:56:71:62:ac:fd:
    70:f7:76:04:ae:74:00:07:05:e9:92:01:ee:75:30:
    09:d3:95:bb:91:2c:51:ce:ad:e1:8f:b7:7d:8f:af:
    2e:e5:81:af:e9:ac:5f:ea:2e:4d:27:96:b9:e6:ce:
    0d:8a:d6:bc:e0:c4:bf:9f:27:53:95:5c:4b:28:b4:
    52:73:12:23:20:0a:7a:70:93:2e:87:d2:84:ed:10:
    3b:3d:da:10:cd:59:4f:06:be:38:68:0c:44:30:51:
    14:bc:84:0e:4e:f8:c6:a5:a8:33:9e:2b:f4:e8:52:
    ed:f3:a5:5f:fc:d4:ec:7a:cd:01:40:de:48:7a:a4:
    35:12:33:2a:22:26:10:5d:23:49:94:32:d3:70:cf:
    57:71:f8:e8:c5:51:80:7a:30:77:3f:d7:33:a8:28:
    0c:39:5f:67:ce:74:ae:5a:6a:40:b7:fd:c7:40:b9:
    44:ff
publicExponent: 65537 (0x10001)
privateExponent:
    25:f1:18:e1:e4:bf:60:37:51:bd:6a:37:08:f5:33:
    5d:d1:b1:7c:d9:4a:73:0e:b8:0f:ab:be:df:3b:c0:
    be:5c:36:15:8a:bf:8f:28:f0:08:1c:3c:03:c0:3c:
    eb:3d:07:20:6d:ab:68:56:f8:98:01:bc:e3:c8:d7:
    44:b4:f7:b6:cd:e5:4b:e4:b6:fd:6a:14:5b:84:48:
    b7:f9:66:77:f7:cf:92:bf:02:74:dd:6e:0c:2f:f9:
    56:d1:7c:f9:4e:ce:6e:d6:d0:c9:d7:d0:01:fe:f6:
    5e:6f:97:99:c4:f3:89:e4:2d:94:19:9e:be:9a:81:
    f7:5d:f2:54:62:81:ba:81:2d:43:44:dc:aa:11:2e:
    06:76:d6:5d:e6:f3:9a:99:0f:f7:e4:4a:2e:1e:a8:
    3c:38:ca:d3:1f:cf:52:b3:2f:23:25:f3:27:9d:4c:
    c6:b9:a9:b8:32:09:1e:02:c3:54:ff:50:3b:da:cf:
    ce:e6:d4:2a:b8:39:37:30:dd:e3:7d:07:cb:c6:1e:
    f2:91:1c:4f:19:0b:25:8e:5e:c4:a1:66:4a:d6:54:
    05:cb:4f:03:91:c1:18:d6:c5:b4:13:fa:db:b9:ac:
    8c:15:ac:9a:4c:18:61:0c:92:9c:de:ae:3f:af:7c:
    cd:03:24:e5:19:84:e0:52:82:73:87:ce:3d:ca:68:
    11
prime1:
    00:cc:66:ea:af:8e:db:1e:2c:a6:ec:4a:c2:84:04:
    c4:f0:10:46:c6:a4:16:99:55:a6:3d:33:69:85:29:
    a6:e4:0d:0a:30:41:89:5f:69:46:16:75:89:2c:13:
    97:c9:3a:da:9a:19:79:04:bb:d1:cf:33:34:42:a7:
    0a:c9:aa:40:d7:68:2c:55:2a:79:70:6f:2b:31:a6:
    9b:88:07:6a:a4:b3:e1:42:46:3d:2d:d7:14:b9:dc:
    51:5e:5b:ad:98:15:4e:9a:54:76:7b:4b:92:80:30:
    19:ca:ed:4e:66:60:24:55:5d:53:3a:0a:84:a9:58:
    0f:ef:1c:e3:ab:74:a7:01:89
prime2:
    00:ca:d1:bd:ff:53:11:0f:59:d6:70:42:4a:f6:f8:
    b8:79:58:2a:45:f0:da:a3:13:91:5c:0c:29:69:68:
    06:7c:4a:50:b8:57:d3:46:ee:76:64:e9:d4:d0:f0:
    62:72:b7:0b:76:fc:88:e0:2c:dd:e0:51:87:54:81:
    6c:3a:d9:24:f6:70:ee:37:68:ca:0e:53:9a:da:ad:
    e3:8a:f2:64:78:2c:c8:cd:7b:87:78:2b:3e:9b:b2:
    fd:0e:a2:82:9d:8b:57:c9:9b:50:3b:b1:73:dd:8a:
    d8:02:24:99:0e:21:30:89:96:24:74:73:74:58:ac:
    22:8d:ec:c1:f3:4a:6b:18:47
exponent1:
    1f:b7:44:69:13:38:8a:b9:10:f9:8f:2c:19:82:07:
    f0:dc:05:af:62:dc:a2:d4:38:64:3c:aa:fd:df:95:
    a6:c3:23:36:93:a4:b6:2a:46:96:cb:8c:0a:4b:cb:
    fc:8c:66:49:0c:f4:cb:38:3d:78:f2:10:04:db:98:
    7f:ec:2b:fd:80:8b:88:77:7b:ba:67:75:83:51:bf:
    c6:5e:a8:3c:80:30:ce:b0:c9:63:68:81:2d:c9:b2:
    d2:7a:55:38:48:ee:16:62:4e:b4:18:e8:d5:7d:21:
    33:6c:99:02:a7:9e:5b:a4:8c:7a:70:11:5f:99:6b:
    77:85:bb:09:ea:a2:d3:49
exponent2:
    6c:89:07:52:de:21:f0:04:14:64:28:a5:b3:3a:70:
    85:c7:2f:80:14:1e:fc:4b:d9:6a:d7:bb:45:83:fc:
    8c:80:bc:67:99:e0:1f:42:37:2b:53:92:a6:69:2d:
    55:d4:2c:97:75:16:ce:58:54:37:d9:89:41:28:ad:
    41:64:b8:8a:04:ef:05:c9:d6:ef:09:c0:81:ab:b2:
    6b:c5:c1:0c:2a:73:1e:bb:2f:48:98:63:bd:b0:70:
    09:67:df:db:ed:18:82:73:d1:e0:cf:da:80:40:ef:
    cd:5d:c6:66:30:4e:a8:22:27:ca:05:8a:ca:74:fb:
    5c:d5:c9:08:a7:d7:5d:bb
coefficient:
    0e:20:19:aa:43:55:95:70:08:72:f1:54:2a:56:29:
    53:73:6d:76:91:c9:26:3b:3c:0b:12:4d:1d:2b:ac:
    62:56:56:90:5b:7b:5c:d1:57:ef:dd:80:37:98:58:
    26:a0:54:b7:6f:8f:a4:8a:e6:52:47:5e:20:7b:2f:
    e6:5e:86:40:dd:8f:7d:e7:e9:f8:4c:db:ec:50:d2:
    02:65:f9:cb:e0:5c:b7:ab:f6:1b:df:37:bd:1d:02:
    27:f1:21:8c:67:69:13:fb:09:f5:4e:20:a3:03:44:
    02:0a:2c:4d:35:63:ba:09:80:f2:9e:70:85:2b:c3:
    4d:a1:f2:a6:d6:88:2c:b9
```

Observation:

We note that the values in server.key file is shorter that the values in ca.key file in task 1 because the key size in rsa is 2048 for task 2 while the key size in task 1 is 4096.

2) Adding alternative names:

```
[05/23/22]zaina@VM:~/.../PKI$ openssl req -newkey rsa:2048 -sha256  -keyout ser
ver1.key -out server1.csr  -subj "/CN=www.zaina2022.com/O=ZA Inc./C=PS" -addext
 "subjectAltName=DNS:www.zaina2022.com, DNS:www.zaina2022A.com, DNS:www.zaina20
22B.com"  -passout pass:seed
Generating a RSA private key
.....................................................+++++
..........+++++
writing new private key to 'server1.key'
-----
```



The decoded certificate shows the DNS alternatives:

$openssl req -in server1.csr -text –noout

```
[05/23/22]zaina@VM:~/.../PKI$ openssl req -in server1.csr -text -noout
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: CN = www.zaina2022.com, O = ZA Inc., C = PS
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:b1:92:88:a0:60:52:f9:0f:30:aa:69:d7:19:ad:
                    d9:6f:a2:89:49:6d:fb:51:43:85:86:54:ef:76:e7:
                    5a:a8:3c:e4:ab:be:f4:2c:51:39:03:36:95:58:ba:
                    28:66:cf:23:10:5e:3d:05:0e:f8:da:73:fd:ee:36:
                    d1:60:08:51:e9:2c:32:76:e1:e4:e0:70:ae:f6:6d:
                    f0:fd:5a:1e:bf:01:a7:4e:07:b0:ec:3d:cb:9f:db:
                    28:33:4e:09:df:2f:5a:5f:ea:2d:1e:7f:03:72:8c:
                    27:ff:42:8d:ac:b4:0d:d1:6a:17:94:23:e9:ca:1f:
                    7b:fb:7e:4e:b0:9e:99:19:fc:2d:27:67:0e:82:cc:
                    4f:79:67:fb:82:b1:c3:4e:1b:13:2e:d9:cb:ed:b1:
                    aa:2e:de:86:08:ec:b7:b3:a2:26:5c:f8:8e:48:91:
                    6a:55:6d:2c:f5:65:18:a8:2b:c4:39:74:c7:d9:81:
                    10:b4:34:65:46:2d:a1:a5:d0:54:26:99:be:c5:12:
                    23:31:ce:ad:f1:3a:68:52:7b:69:c2:14:41:b7:1c:
                    a3:e6:d0:b5:2c:65:a2:08:4f:60:ce:6f:da:36:b8:
                    39:4e:d9:d2:29:b9:bd:89:d8:43:b5:15:76:b2:ca:
                    e3:c9:24:da:42:8b:fd:34:94:3e:b9:ac:ad:57:7a:
                    ec:4f
                Exponent: 65537 (0x10001)
        Attributes:
        Requested Extensions:
            X509v3 Subject Alternative Name:
                DNS:www.zaina2022.com, DNS:www.zaina2022A.com, DNS:www.zaina2022B.com
    Signature Algorithm: sha256WithRSAEncryption
```

Task 3: Generating a Certificate for your server

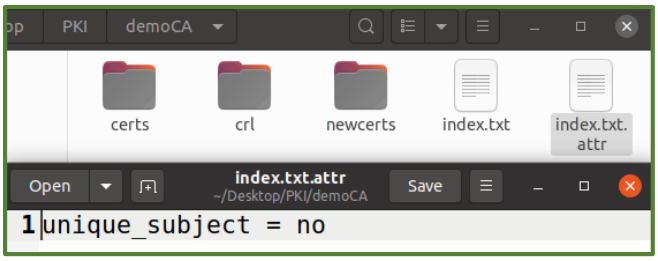
Before beginning:

1) Uncomment line 68 command to copy the extension field from the request to the final certificate

```
Open   openssl.cnf  ~/Desktop/PKI
62 # Comment out the following two lines for the "traditional"
63 # (and highly broken) format.
64 name_opt        = ca_default            # Subject Name options
65 cert_opt        = ca_default            # Certificate field options
66
67 # Extension copying option: use with caution.
68 copy_extensions = copy
```

2) uncomment line 49 command the unique subject line to allow creation of certifications with the same subject

```
Open   openssl.cnf  ~/Desktop/PKI   Save
45 dir             = ./demoCA              # Where everything is kept
46 certs           = $dir/certs            # Where the issued certs are kept
47 crl_dir                 = $dir/crl              # Where the issued crl are kept
48 database        = $dir/index.txt        # database index file.
49 unique_subject  = no                    # Set to 'no' to allow creation of
50                                         # several certs with same subject.
```



Observation: the index.txt.attr is created automatically by adding the unique subject configuration with “no”.

3) Ensure that we use the policy anything policy defined in the configuration file

```
 94 # For the 'anything' policy
 95 # At this point in time, you must list all acceptable 'object'
 96 # types.
 97 [ policy_anything ]
 98 countryName             = optional
 99 stateOrProvinceName     = optional
100 localityName            = optional
101 organizationName        = optional
102 organizationalUnitName  = optional
103 commonName              = supplied
104 emailAddress            = optional
105
106 ####################################################################
```

Create the certificate:

```
$openssl ca -config openssl.cnf -policy policy_anything -md sha256 -days 3650  -in server.csr -out server.crt -batch  -cert ca.crt -keyfile ca.key
```

```
[05/23/22]zaina@VM:~/.../PKI$ openssl ca -config openssl.cnf -policy policy_anything -md sha256 -days
3650  -in server.csr -out server.crt -batch  -cert ca.crt -keyfile ca.key
Using configuration from openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4096 (0x1000)
        Validity
            Not Before: May 23 16:38:47 2022 GMT
            Not After : May 20 16:38:47 2032 GMT
        Subject:
            countryName               = PS
            organizationName          = ZA Inc.
            commonName                = www.zaina2022.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                C9:C6:F9:68:10:B8:F9:C1:E8:11:83:21:35:8E:5E:12:E1:62:8D:E5
            X509v3 Authority Key Identifier:
                keyid:A9:ED:99:5A:7E:99:63:FB:A7:AD:8A:44:D6:9A:99:5D:19:AB:36:44

Certificate is to be certified until May 20 16:38:47 2032 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated
```

Validity to 10 years => 3650 days

Subject information fetched from the CSR file

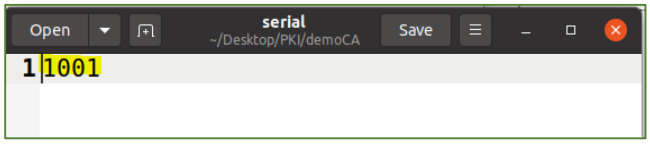
Indicated that the certificate is not for this CA but for another subject

Indicates that the certificate is for another subject because the subject identifier is different from the authority identifier

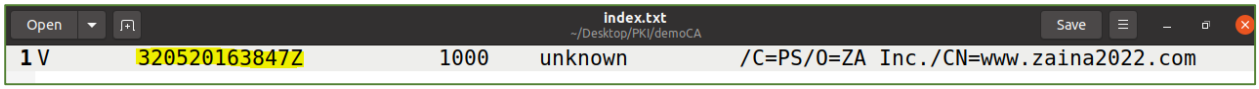
Update the index.txt file which represents the database

Observation:

Serial file value Increased by 1 for the next record

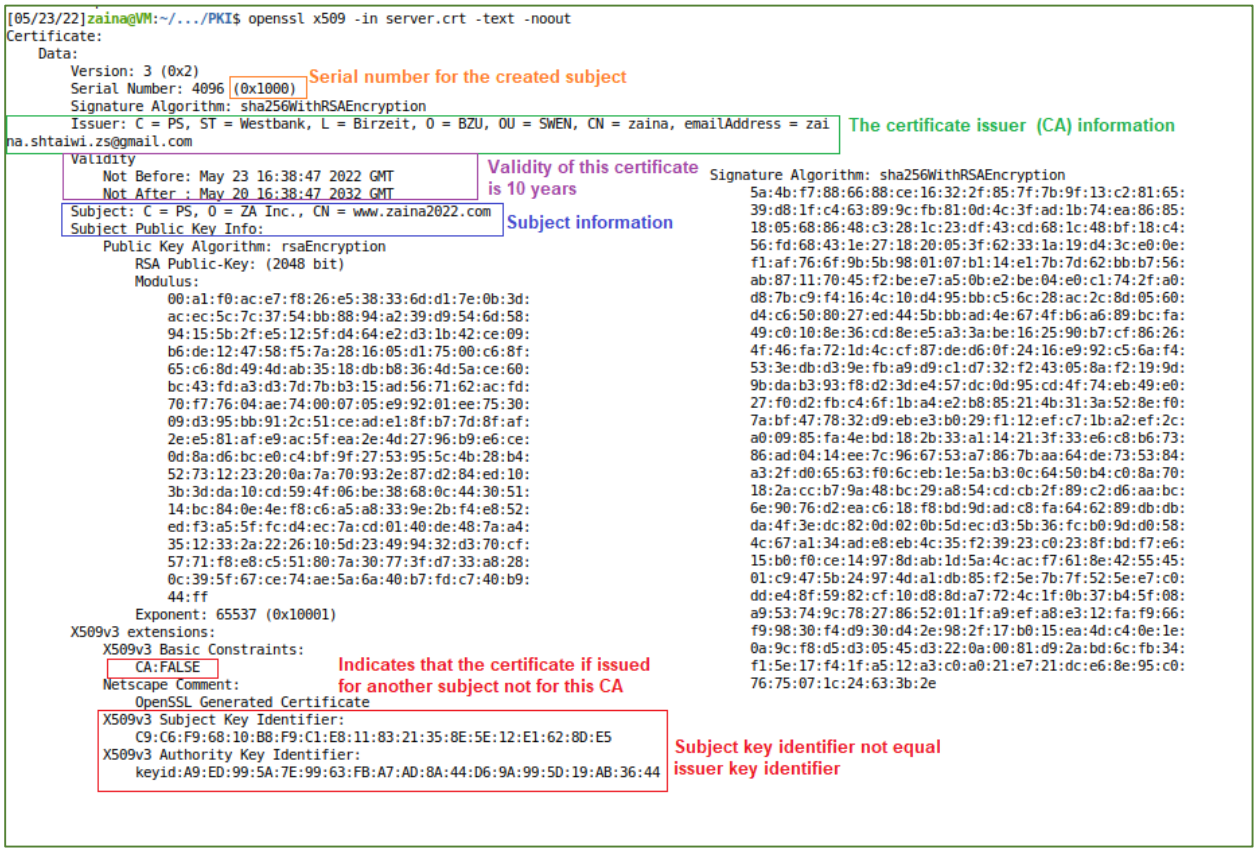


Index.txt file updated by adding the new certificate information



View the certificate:

$ openssl x509 -in server.crt -text -noout



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

Review the CSR creation and CRT creation for our website

```
$openssl req -newkey rsa:2048 -sha256  -keyout zaina2022.key -out zaina2022.csr  -subj "/CN=www.zaina2022.com/O=ZA Inc./C=PS" -addext
"subjectAltName = DNS:www.zaina2022.com, DNS:www.zaina2022A.com,DNS:www.zaina2022B.com" -passout pass:dees
$openssl ca -config openssl.cnf -policy policy_anything -md sha256 -days 3650  -in zaina2022.csr -out zaina2022.crt -batch  -cert ca.crt -keyfile ca.key
```

1) Update apache ssl configuration file

Update the configuration host file of the Bank32 file to match our site names, certificate/ key files for 443 port, and the indices files for both https (443 port) and http (80 port).

zaina2022_apache_ssl.conf

```
<VirtualHost *:443>
    DocumentRoot /var/www/zaina2022
    ServerName www.zaina2022.com
    ServerAlias www.zaina2022A.com
    ServerAlias www.zaina2022B.com
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /certs/zaina2022.crt
    SSLCertificateKeyFile /certs/zaina2022.key
</VirtualHost>

<VirtualHost *:80>
    DocumentRoot /var/www/zaina2022
    ServerName www.zaina2022.com
    DirectoryIndex index_red.html
</VirtualHost>

# Set the following gloal entry to suppress an annoying warning message
ServerName localhost
```

https site configurations =>443 port

public/private keys required

http site configurations => 80 port

public/private keys not required

2) Update the Dockerfile to run the commands while building the docker containers.

Dockerfile

```
FROM handsonsecurity/seed-server:apache-php

ARG WWWDIR=/var/www/zaina2022

COPY ./index.html ./index_red.html $WWWDIR/
COPY ./zaina2022_apache_ssl.conf /etc/apache2/sites-available
COPY ./certs/zaina2022.crt ./certs/zaina2022.key  /certs/

RUN  chmod 400 /certs/zaina2022.key \
     && chmod 644 $WWWDIR/index.html \
     && chmod 644 $WWWDIR/index_red.html \
     && a2ensite zaina2022_apache_ssl

CMD  tail -f /dev/null
```

Our site directory

copy index.html and index_red.html to oir site directory

copy our site ssl configuration file to the apache2 sites-available directory

copy public/private pair to the certs directort

adjust permissions for the copies files

Enable our site in apache2 server

3) Get the container id, run the docker container, enable our site, and restart apache2 server

```
[05/25/22]zaina@VM:~/.../Labsetup$ dockps
f664d3e80437  www-10.9.0.80
[05/25/22]zaina@VM:~/.../Labsetup$ docksh f664d3e80437
root@f664d3e80437:/# 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
root@f664d3e80437:/var/www/html# a2ensite zaina2022_apache_ssl
Site zaina2022_apache_ssl already enabled
root@f664d3e80437:/var/www/html# service apache2 reload
 * Reloading Apache httpd web server apache2
 *
root@f664d3e80437:/var/www/html# service apache2 restart
 * Restarting Apache httpd web server apache2
Enter passphrase for SSL/TLS keys for www.zaina2022.com:443 (RSA):
root@f664d3e80437:
```

Container id

Run the container shell

Enable the SSL module

Already enabled while building the Dockerfile in docker-compose build

Enable the sites described in zaina2022_apache_ssl file

optional

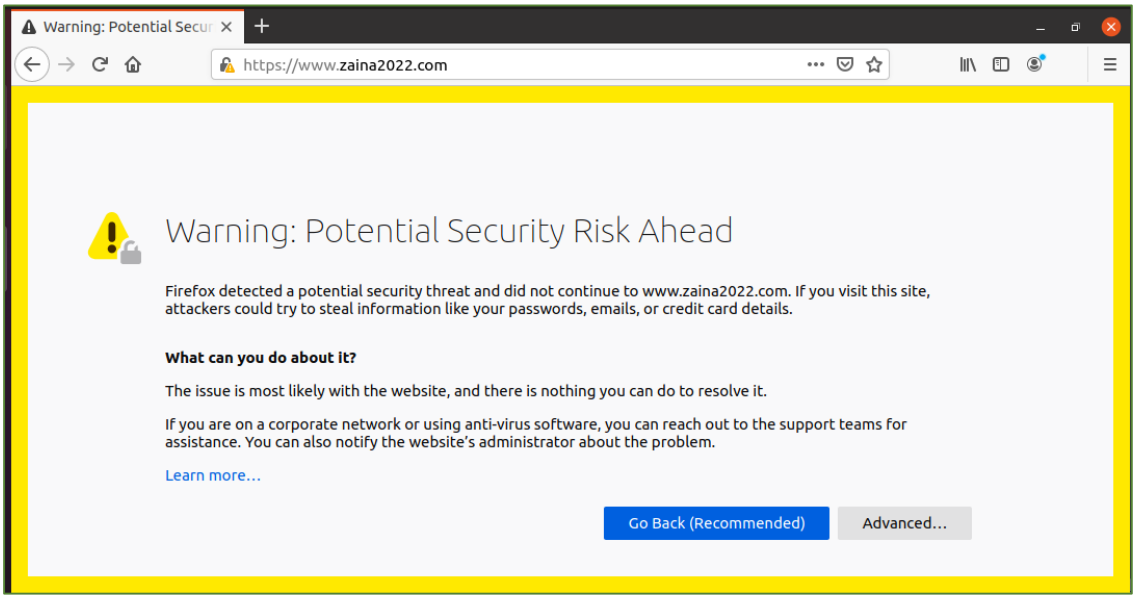
Restart Apache server

password = "dees" According to CA

4) Make sure that necessary files are copied to the required destination after building the container.

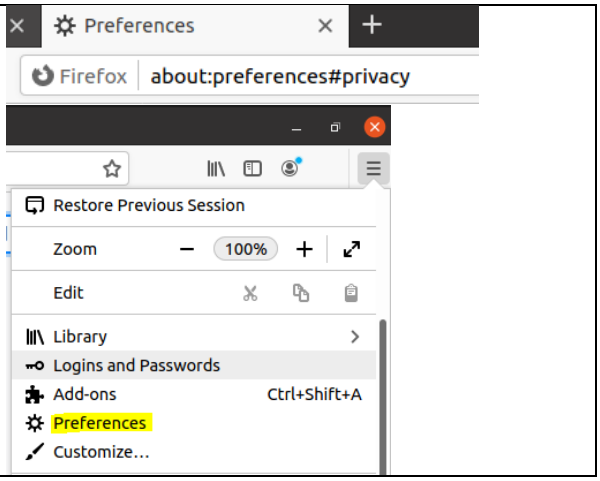
```
root@f664d3e80437:/# cd certs
root@f664d3e80437:/certs# ls
bank32.crt  bank32.key  zaina2022.crt  zaina2022.key
root@f664d3e80437:/var/www# ls
bank32  html  zaina2022
root@f664d3e80437:/var/www# cd zaina2022
root@f664d3e80437:/var/www/zaina2022# ls
index.html  index_red.html
root@f664d3e80437:/# cd etc
root@f664d3e80437:/etc# cd apache2
root@f664d3e80437:/etc/apache2# cd sites-available
root@f664d3e80437:/etc/apache2/sites-available# ls
000-default.conf  bank32_apache_ssl.conf  default-ssl.conf  zaina2022_apache_ssl.conf
```

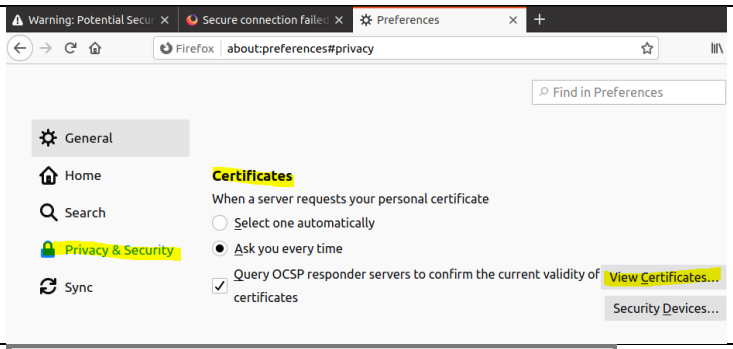
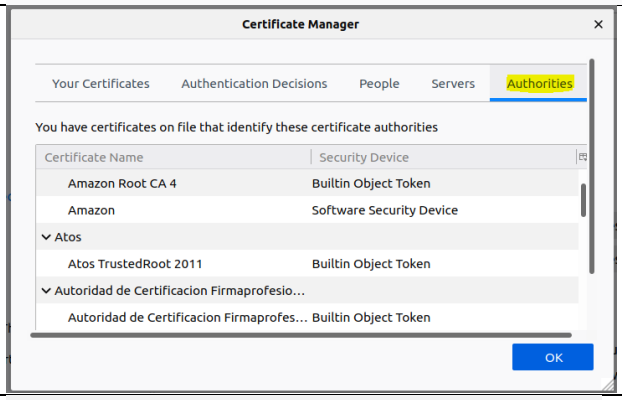
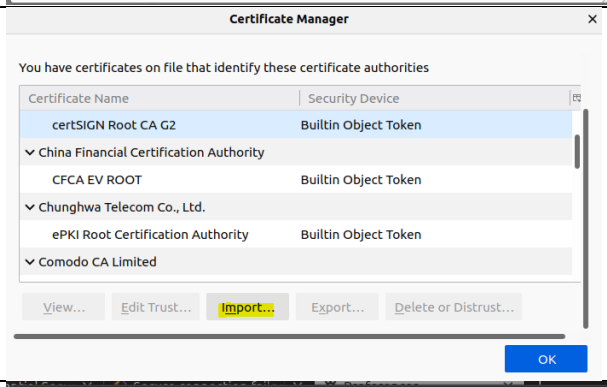
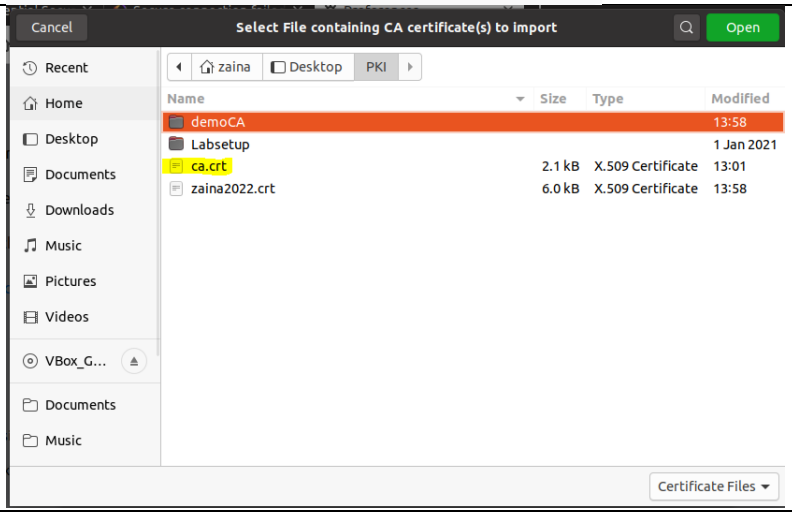
5) Open our site using https protocol, the website did not open because of security issue. This is because root CA Created for ourselves, and the browsers don't trust. So, we need to manually add the certificate file to browser CA.



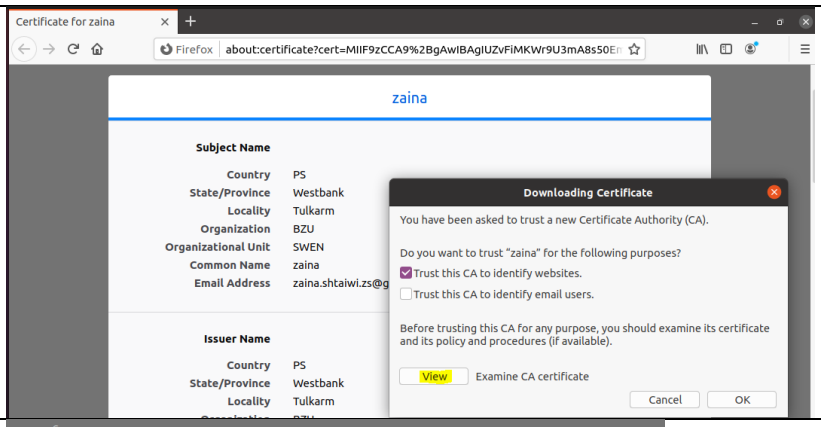
6) Add the certificate authority manually:

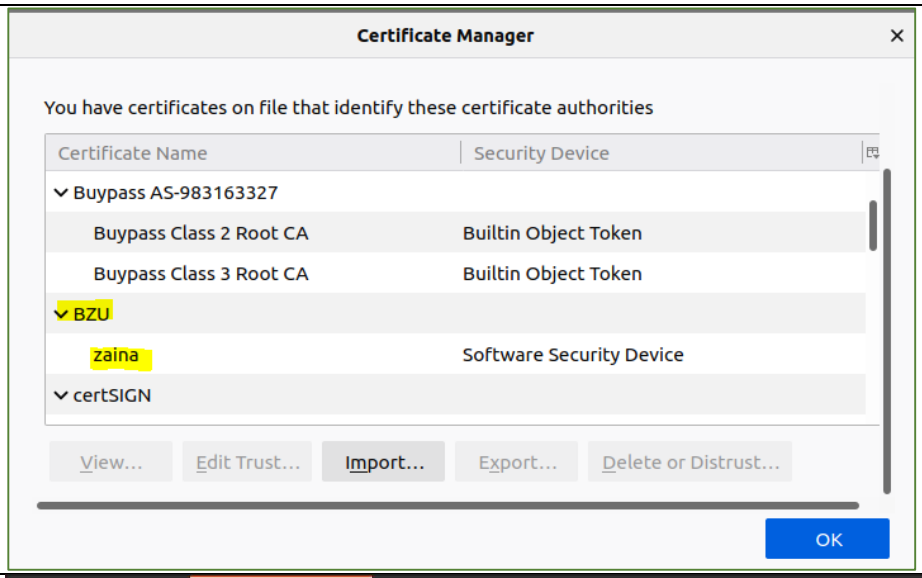
Step 1: go to privacy and security in firefox browser either by typing the path address or by going to edit→ preferences



Step2: go to Certificates→ view Certificates	
Step 3: go to Authorities	
Step 4: click <u>import</u> to add our CA authority	
Step 5: select the CA public key file (ca.cert) that we created in task 1	

Step 6: select “Trust the CA to identify websites” and view the certificate authority information



Step 7: ensure that the CA is added.	
Step 8: close the browser then re-open our website using https. This time the website is trusted by the CA because the website is signed by the CA.	

Task 5 in the next page

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

Step 1: Setting up the malicious website.

1) Create a fake Facebook page that looks like the real Facebook homepage using html and css

Index.html

```
<html>
<head>
    <link href="Style.css" rel="stylesheet" />
    <title>Facebook home page template html</title>
</head>
<body>
    <div id="header_wrapper">
        <div id="header">
            <form action="post">
             <li>Email or Phone<br><input type="text" name="email"></li>
             <li>Password<br><input type="password" name="password"><br><a href="">Forgotten account?</a></li>
                <li><input type="submit" name="login" value="Log In"></li>
            </form>
        </div>
    </div>
    <div id="wrapper">
        <div id="div1">
        </div>
        <div id="div2">
            <h1>Create an account</h1>
            <p>It's free and always will be.</p>
            <li><input type="text" placeholder="First Name" id="Firstname"><input type="text" placeholder="Sur
name" id="surname"></li>
            <li><input type="text" placeholder="Mobile number or email"></li>
            <li><input type="password" placeholder="New password"></li>
            <p>Birthday</p>
            <li>
                <select><option>Day</option></select>
                <select><option>Month</option></select>
                <select><option>Year</option></select>
                <a href="">Why do I need to provide my date of birth?</a>
            </li>
            <li><input type="radio">Female <input type="radio">Male</li>
            <li id="terms">By clicking Create an account, you agree to our <a href="">Terms</a> and that <br>y
ou have read our <a href="">Data Policy</a>, including our <a href="">Cookie Use</a>.</li>
            <li><input type="submit" value="Create an account"></li>
            <li id="create_page"><a href="">Create a Page</a> for a celebrity, band or business.</li>
        </div>
    </div>
    <div id="footer_wrapper">
        <div id="footer1">
            English (UK) <a href="">हिन्दी</a><a href="">ਪੰਜਾਬੀ</a><a href=""> اردو</a><a href="">தமிழ்</a><a href="
">বাংলা</a><a href="">मराठी</a><a href="">తెలుగు</a><a href="">ગુજરાતી</a><a href="">ಕನ್ನಡ</a><a href="">മലയാളം</
a>
        </div>
        <div id="footer2">
            <a href="#">Sign Up</a><a href="#">Log In</a><a href="#">Messenger</a><a href="#">DotNetTec</a><a
href="#">Mobile</a><a href="#">Find Friends</a>
            <a href="#">Badges</a><a href="#">People</a><a href="#">Pages</a><a href="#">Places</a><a href="#"
>Games</a><a href="#">Locations</a>
            <a href="">Celebrities</a><a href="">Groups</a><a href="">Moments</a><a href="">About</a>
            <a href="">Create Advert</a><a href="">Create Page</a><a href="">Developers</a>
            <a href="">Careers</a><a href="">Privacy</a><a href="">Cookies</a><a href="">Ads</a><a href="">Ter
ms</a><a href="">Help</a>
        </div>
    </div>
</body>
</html>
```

Style.css file

```css
* {
    padding: 0;
    margin: 0;
    box-sizing: border-box;
}
body {
    font-family: 'Poppins', sans-serif;
    background-color: #f0f2f5;
    color: #1c1e21;
}
main {
    height: 90vh;
    width: 100vw;
    position: relative;
    margin: 0 auto;
}
footer {
    height: 10vh;
    background-color: #ffffff;
}

.row {
    display: flex;
    justify-content: space-around;
    align-items: center;
    width: 100%;
    max-width: 1000px;
    position: absolute;
    left: 50%;
    top: 50%;
    transform: translate(-50%, -50%);
}
.colm-logo {
    flex: 0 0 50%;
    text-align: left;
}
.colm-form {
    flex: 0 0 40%;
    text-align: center;
}
.colm-logo img {
    max-width: 400px;
}
.colm-logo h2 {
    font: 26px;
    font-weight: 400;
    padding: 0 30px;
    line-height: 32px;
}
.colm-form .form-container {
    background-color: #ffffff;
    border: none;
    border-radius: 10px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1), 0 8px 16px rgba(0, 0, 0, 0.1);
    margin-bottom: 30px;
    padding: 20px;
    max-width: 400px;
}
.colm-form .form-container input, .colm-form .form-container .btn-login {
    width: 100%;
    margin: 5px 0;
    height: 45px;
    vertical-align: middle;
    font-size: 16px;
}
```

```css
.colm-form .form-container input {
    border: 1px solid #dddfe2;
    color: #1d2129;
    padding: 0 8px;
    outline: none;
}
.colm-form .form-container input:focus {
    border-color: #1877f2;
    box-shadow: 0 0 0 2px #e7f3ff;
}
.colm-form .form-container .btn-login {
    background-color: #1877f2;
    border: none;
    border-radius: 6px;
    font-size: 20px;
    padding: 0 16px;
    color: #ffffff;
    font-weight: 700;
}
.colm-form .form-container a {
    display: block;
    color: #1877f2;
    font-size: 14px;
    text-decoration: none;
    padding: 10px 0 20px;
    border-bottom: 1px solid #dadde1;
}
.colm-form .form-container a:hover {
    text-decoration: underline;
}
.colm-form .form-container .btn-new {
    background-color: #42b72a;
    border: none;
    border-radius: 6px;
    font-size: 17px;
    line-height: 48px;
    padding: 0 16px;
    color: #ffffff;
    font-weight: 700;
    margin-top: 20px;
}
.colm-form p {
    font-size: 14px;
}
.colm-form p a {
    text-decoration: none;
    color: #1c1e21;
    font-weight: 600;
}
.colm-form p a:hover {
    text-decoration: underline;
}
.footer-contents {
    position: relative;
    max-width: 1000px;
    margin: 0 auto;
}
footer ol {
    display: flex;
    flex-wrap: wrap;
    list-style-type: none;
    padding: 10px 0;
}
footer ol:first-child {
    border-bottom: 1px solid #dddfe2;
}
footer ol:first-child li:last-child button {
    background-color: #f5f6f7;
    border: 1px solid #ccd0d5;
```

```
    outline: none;
    color: #4b4f56;
    padding: 0 8px;
    font-weight: 700;
    font-size: 16px;
}
footer ol li {
    padding-right: 15px;
    font-size: 12px;
    color: #385898;
}
footer ol li a {
    text-decoration: none;
    color: #385898;
}
footer ol li a:hover {
    text-decoration: underline;
}
footer small {
    font-size: 11px;
}
```

2) Update the Dockerfile to build the container with the new website necessary files : public/private keys, copy the homepage files

Dockerfile
~/Desktop/PKI/Labsetup/image_www

```
1 FROM handsonsecurity/seed-server:apache-php
2
3 ARG WWWDIR=/var/www/facebook
4
5 COPY ./index.html ./Style.css $WWWDIR/
6 COPY ./facebook_apache_ssl.conf /etc/apache2/sites-available
7 COPY ./certs/facebook.crt ./certs/facebook.key  /certs/
8
9 RUN  chmod 400 /certs/facebook.key \
10     && chmod 644 $WWWDIR/index.html \
11     && chmod 644 $WWWDIR/Style.css \
12     && a2ensite facebook_apache_ssl
13
14 CMD  tail -f /dev/null
```

3) Update the “facebook_apache_ssl” config file by adding the necessary https and http configurations that should be copied in sites-available directory. Suppose that facebook.crt and facebook.key are a certificate files that was created by untrusted CA.

facebook_apache_ssl.conf
~/Desktop/PKI/Labsetup/image_www

```
1 <VirtualHost *:443>
2     DocumentRoot /var/www/facebook
3     ServerName www.facebook.com
4     DirectoryIndex index.html
5     SSLEngine On
6     SSLCertificateFile /certs/facebook.crt
7     SSLCertificateKeyFile /certs/facebook.key
8 </VirtualHost>
9
10 <VirtualHost *:80>
11     DocumentRoot /var/www/facebook
12     ServerName www.facebook.com
13     DirectoryIndex index.html
14 </VirtualHost>
15
16 # Set the following gloal entry to suppress an
   annoying warning message
17 ServerName localhost
```

HTTPS

HTTP

4) Build docker-compose yml file that execute the Dockerfile commands then create and start the container.

```
[05/26/22]zaina@VM:~/.../Labsetup$ docker-compose build
Building web-server
Step 1/7 : FROM handsonsecurity/seed-server:apache-php
 ---> 2365d0ed3ad9
Step 2/7 : ARG WWWDIR=/var/www/facebook
 ---> Running in 84ab41cc7145
Removing intermediate container 84ab41cc7145
 ---> cb2f16ba113f
Step 3/7 : COPY ./index.html ./Style.css $WWWDIR/
 ---> ae263f0b89a1
Step 4/7 : COPY ./facebook_apache_ssl.conf /etc/apache2/sites-available
 ---> 0d5f32f09c4d
Step 5/7 : COPY ./certs/facebook.crt ./certs/facebook.key  /certs/
 ---> 2dc003fa51bd
Step 6/7 : RUN  chmod 400 /certs/facebook.key      && chmod 644 $WWWDIR/index.ht
ml      && chmod 644 $WWWDIR/Style.css      && a2ensite facebook_apache_ssl
 ---> Running in 2e0c64326529
Enabling site facebook_apache_ssl.
To activate the new configuration, you need to run:
  service apache2 reload
Removing intermediate container 2e0c64326529
 ---> 5070012e3dfc
Step 7/7 : CMD  tail -f /dev/null
 ---> Running in bebae5cc7939
Removing intermediate container bebae5cc7939
 ---> 8dbde5b43aca
```

Building the container by running the Dockerfile commands

```
Successfully built 8dbde5b43aca
Successfully tagged seed-image-www-pki:latest
[05/26/22]zaina@VM:~/.../Labsetup$ dcup
WARNING: Found orphan containers (oracle-10.9.0.80) for this project. If you rem
oved or renamed this service in your compose file, you can run this command with
 the --remove-orphans flag to clean it up.
Recreating www-10.9.0.80 ... done
Attaching to www-10.9.0.80
```

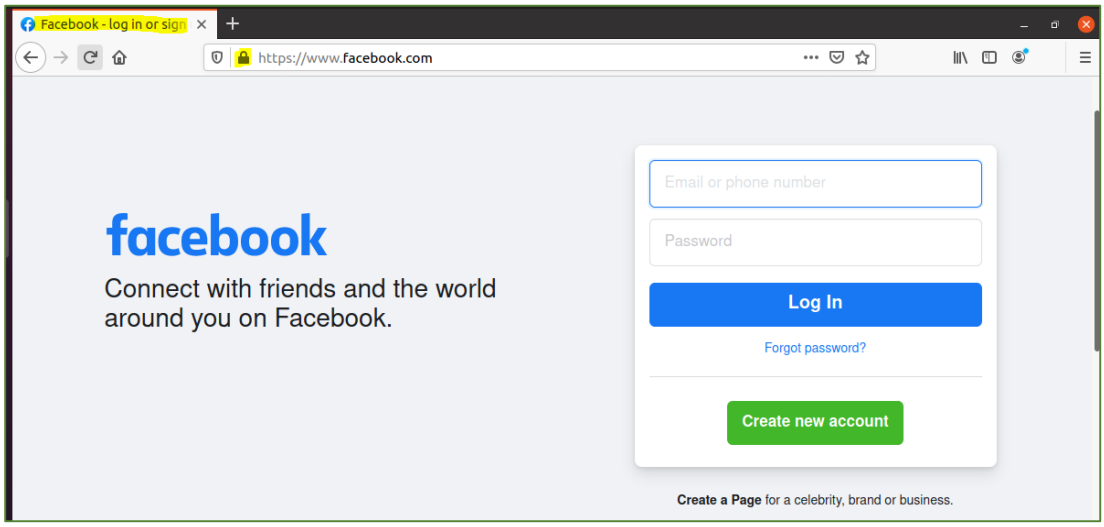
Create and start the container

5) Run the container shell and make sure that all the necessary files is copied correctly, then enable ssl module and the fake facebook site

```
root@79f35d0d6a6f: /etc/...
[05/26/22]zaina@VM:~/.../Labsetup$ dockps
79f35d0d6a6f  www-10.9.0.80
[05/26/22]zaina@VM:~/.../Labsetup$ docksh 79
root@79f35d0d6a6f:/# cd /var/www
root@79f35d0d6a6f:/var/www# ls
facebook  html
root@79f35d0d6a6f:/var/www# cd facebook
root@79f35d0d6a6f:/var/www/facebook# ls
Style.css  index.html
root@79f35d0d6a6f:/var/www/facebook# cd ..
root@79f35d0d6a6f:/var/www# cd /etc/apache2/sites-available
root@79f35d0d6a6f:/etc/apache2/sites-available# ls
000-default.conf  default-ssl.conf  facebook_apache_ssl.conf
root@79f35d0d6a6f:/etc/apache2/sites-available#  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
root@79f35d0d6a6f:/etc/apache2/sites-available#  a2ensite facebook_apache_ssl
Site facebook_apache_ssl already enabled
root@79f35d0d6a6f:/etc/apache2/sites-available# service apache2 restart
 * Restarting Apache httpd web server apache2
Enter passphrase for SSL/TLS keys for www.facebook.com:443 (RSA):
                                                                    [ OK ]
root@79f35d0d6a6f:/etc/apache2/sites-available#
```

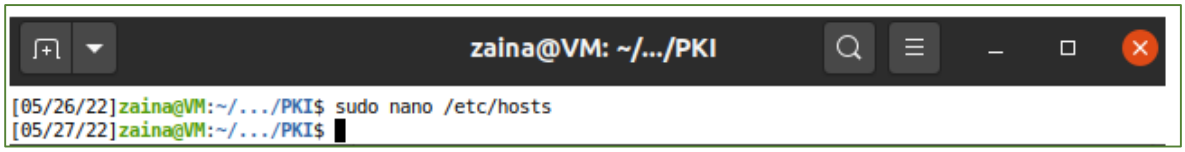
Get the docker container id
Start the container shell
Fake facebook site folder is copied
Fake facebook site files (html, css) is copied
Fake facebook site configuration files is copied
Enable ssl mode
ssl mode already enabled
Enable the fake facebook site
Restart apache2 server

6) Try to open the fake Facebook website by typing : https://wwwfacebook.com in Firefox browser. We observe that the real Facebook page is opened despite that the fake site has a certificate that is signed by a trusted CA.

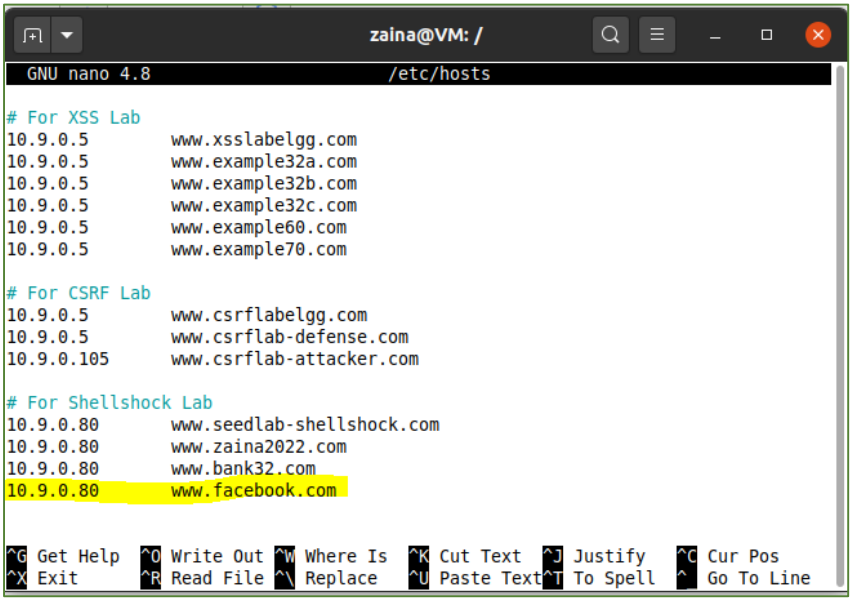


Step 2: Becoming the man in the middle

1) Open the file /etc/hosts using *nano* command

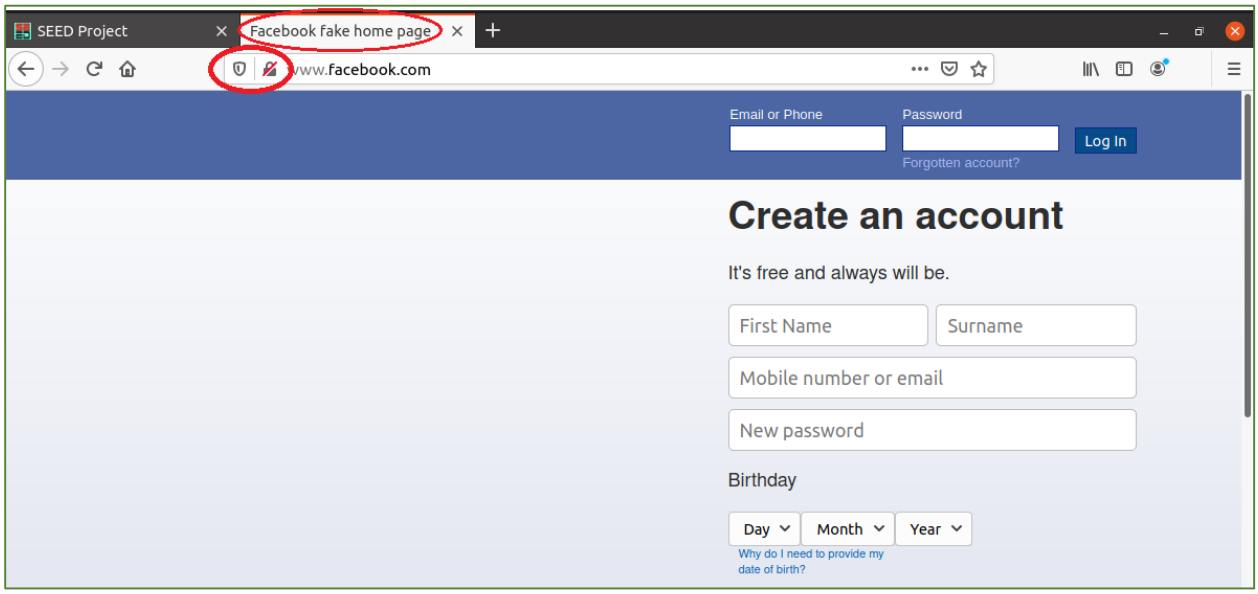


2) Add the following entry in the file. (10.9.0.80 www.facebook.com)

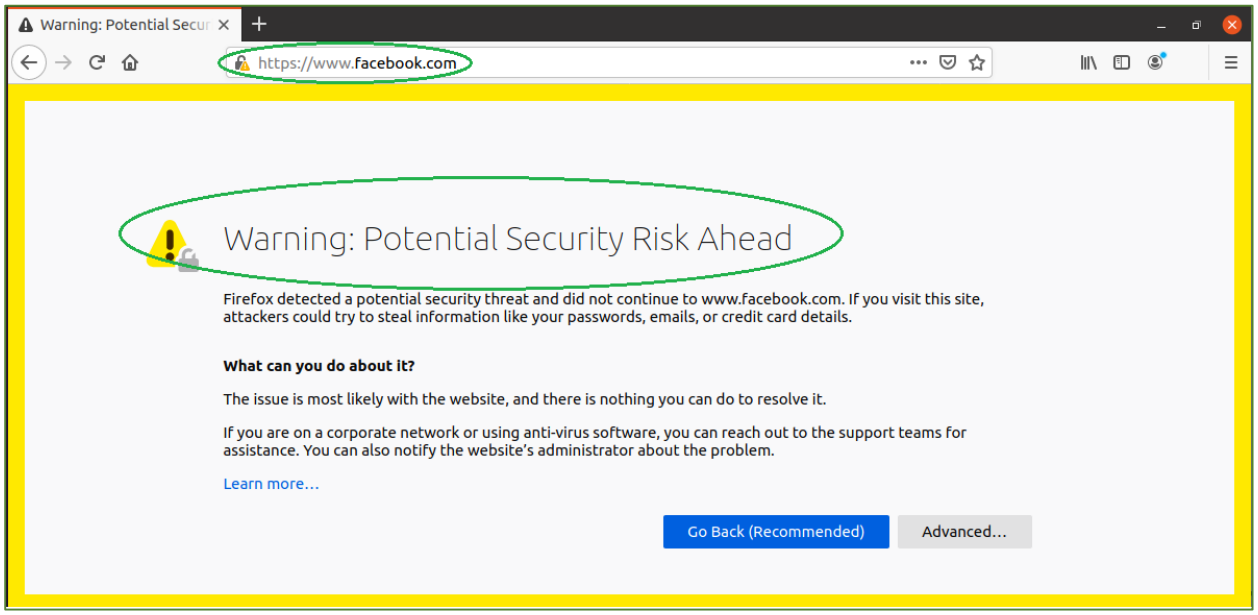


Then Ctrl+X → Y → Enter to save changes

3) Open the fake Facebook using http protocol. The fake page was displayed which is not secured. Note that the page looks like the real Facebook page. When a user enters his credentials to this site it will be stolen by the attacker.



4) Open the fake Facebook using https protocol. The fake page was not displayed because the connection is not secured. Without a valid certificate, a warning is generated by the browser indicating to the user there is a potential issue with the security of the website.



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

Experimental Design:

Step 1: Suppose that we have already compromised a verified root CA.

Step 2: Create new private/public key pair using the compromised and verified root CA. by following the following steps:

Step 2.1 Create the certificate request (.csr) and private key (.key) for the fake facebook site.

Step 2.2 Generate the certificate (facebook.crt) using the CRS file

Step 3: copy the created certificate and key files that is created in step 2 to /certs directory

*Use copy command as follow: cp “source path” “destination path”

*make sure that the certificate files are copied using ls command

Step 4: Create a malicious website similar to the target’s login page using html and css.

Step 5: prepare the VirtualHost file for the malicious website

Step 5.1 Create the SSL configuration file for the malicious website for both http and https and providing the public and private keys for the SSLCertifecate.

Step 5.2 copy the SSL configuration file to /etc/apache2/sites-available

*Use copy command as follow: cp “source path” “destination path”

*make sure that the .conf file is copied using ls command

Step 6: Host the malicious website files in the apache server

Step 6.1 create a folder in /var/www with the website name using mkdir command

Step 6.2 copy the html and css files inside the created directory

Step 7: set permissions options

- ➔ chmod 644 mean that the owner of the file has read and write access, while the group members and other users on the system only have read access.
- ➔ chmod 400 - Gives the user read permission, and removes all other permission.

```
sudo chmod 400 /certs/file.key
sudo chmod 644 /var/www.facebook/indexfile.html
sudo chmod 644 /var/www.facebook /Stylefile.css
```

Step 8: setup apache server

- ➔ Enable SSL module : a2enmod ssl
- ➔ Enable the malicious website : a2ensite milicious_apache_ssl
- ➔ Restart apache server : service apache2 restart

Step 9: Modify the target’s /etc/hosts file by adding the ip and the domain name for the fake website.

* use sudo nano /etc/hosts → add the new entry → ctrl+X → y → Enter

*The ip address would be the real public IP Address of our malicious server.

Step 10: The output of the task performed

Step 10.1 Browse the malicious site using https

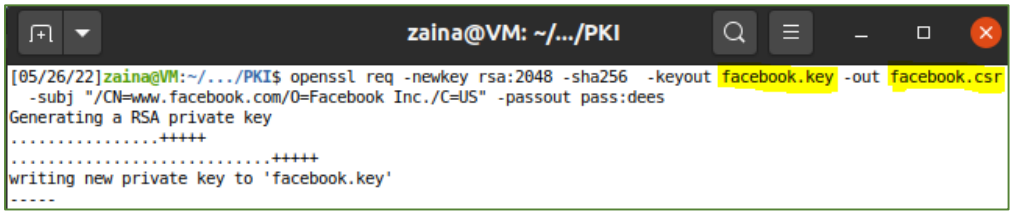
Step 10.2: check the security information and certificate by clicking the lock before the URL

Apply the Experiment:

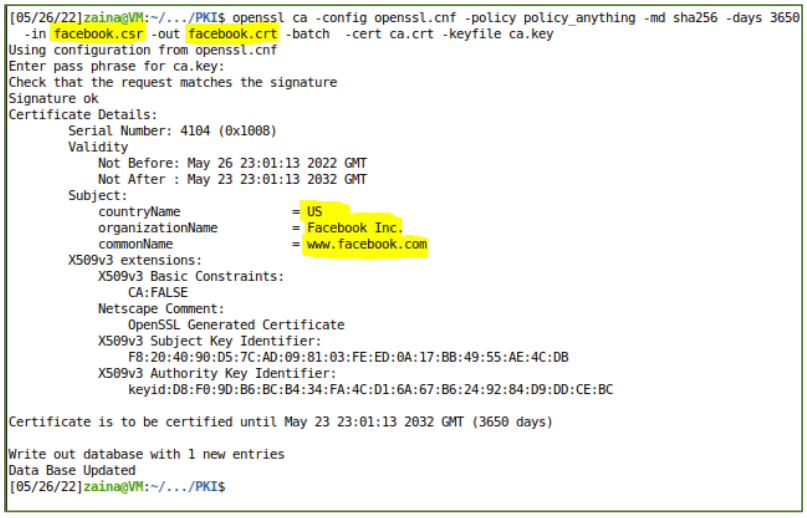
Step 1: Suppose that we have already compromised a verified root CA. Now we can create certificates for any domain.

Step 2: Create new private/public key pair using the compromised and verified root CA. by following the following steps:

Step 2.1 Create the certificate request (facebook.csr) and private key (facebook.key) for the fake facebook site.



Step 2.2 Generate the certificate (facebook.crt) using the CRS file



➔ Here is the created certificate

Step 3: copy the created certificate and key files that is created in step 2 to /certs directory

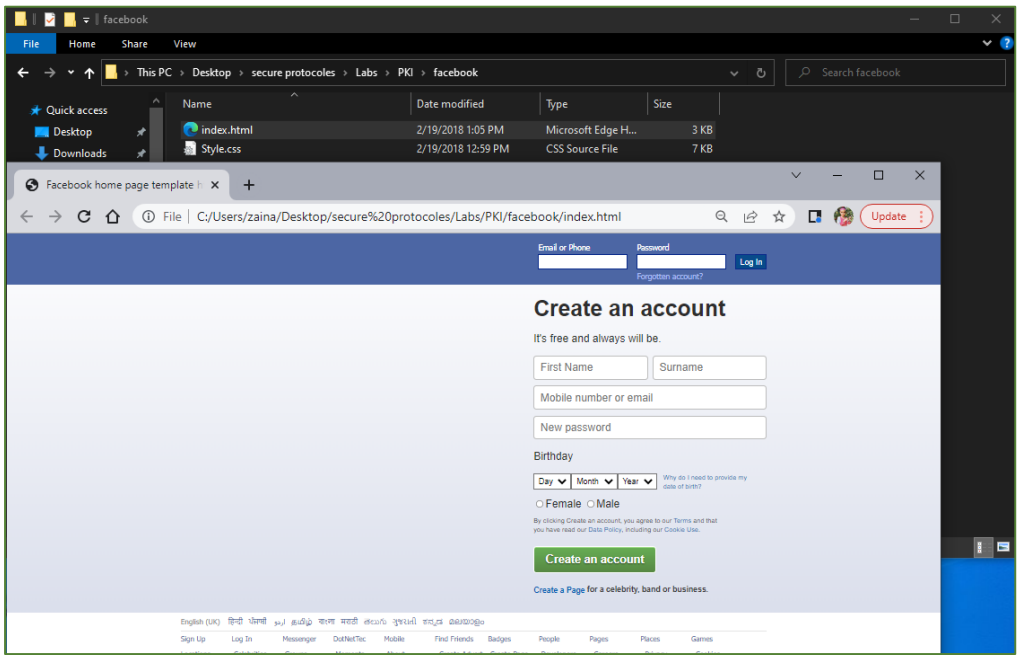
*Use copy command as follow: cp “source path” “destination path”

```
root@79f35d0d6a6f:/certs# cp "/volumes/facebook/facebook.crt" "/certs"
root@79f35d0d6a6f:/certs# cp "/volumes/facebook/facebook.key" "/certs"
```

*make sure that the certificate files are copied using ls command

```
root@79f35d0d6a6f:/# cd certs
root@79f35d0d6a6f:/certs# ls
facebook.crt  facebook.key
```

Step 4: Create a malicious website similar to the target’s login page using html and css.



Step 5: prepare the VirtualHost file for the malicious website

Step 5.1 Create the SSL configuration file for the malicious website for both http and https by providing the public and private keys for the SSLCertifecate (line 6 and 7)

Step 5.2 copy the SSL configuration file to /etc/apache2/sites-available

*Use copy command as follow: cp “source path” “destination path”

```
root@79f35d0d6a6f:/certs# cp "/volumes/facebook/facebook_apache_ssl.conf" "/etc/apache2/sites-available"
```

*make sure that the .conf file is copied using ls command

```
root@79f35d0d6a6f:/certs# cd /etc/apache2/sites-available
root@79f35d0d6a6f:/etc/apache2/sites-available# ls
000-default.conf  default-ssl.conf  facebook_apache_ssl.conf
root@79f35d0d6a6f:/etc/apache2/sites-available#
```

Step 6: Host the malicious website files in the apache server

Step 6.1 create a folder in /var/www with the website name using mkdir command

```
root@79f35d0d6a6f:/var/www# ls
facebook  html
root@79f35d0d6a6f:/var/www#
```

Step 6.2 copy the html and css files inside the created directory

```
root@79f35d0d6a6f:/var/www# cp "/volumes/facebook/index.html" "/var/www.facebook"
root@79f35d0d6a6f:/var/www# cp "/volumes/facebook/Style.css" "/var/www.facebook"
```

```
root@79f35d0d6a6f:/var/www/facebook# ls
Style.css  index.html
```

Step 7: set permissions options

- ➔ chmod 644 mean that the owner of the file has read and write access, while the group members and other users on the system only have read access.
- ➔ chmod 400 - Gives the user read permission, and removes all other permission.

```
sudo chmod 400 /certs/facebook.key
sudo chmod 644 /var/www.facebook/index.html
sudo chmod 644 /var/www.facebook /Style.css
```

Step 8: setup apache server

- ➔ Enable SSL module : a2enmod ssl
- ➔ Enable the malicious website : a2ensite facebook_apache_ssl
- ➔ Restart apache server : service apache2 restart

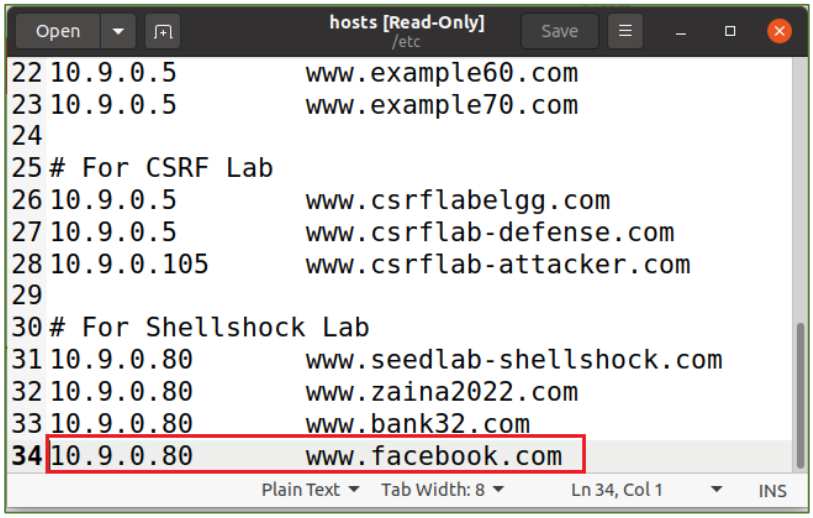
```
root@79f35d0d6a6f:/var/www/facebook# a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
root@79f35d0d6a6f:/var/www/facebook# a2ensite facebook_apache_ssl
Site facebook_apache_ssl already enabled
root@79f35d0d6a6f:/var/www/facebook# service apache2 restart
 * Restarting Apache httpd web server apache2
Enter passphrase for SSL/TLS keys for www.facebook.com:443 (RSA):

root@79f35d0d6a6f:/var/www/facebook#
```

Step 9: Become Man in the Middle by modifying the target's /etc/hosts file by adding the ip and the domain name for the fake website.

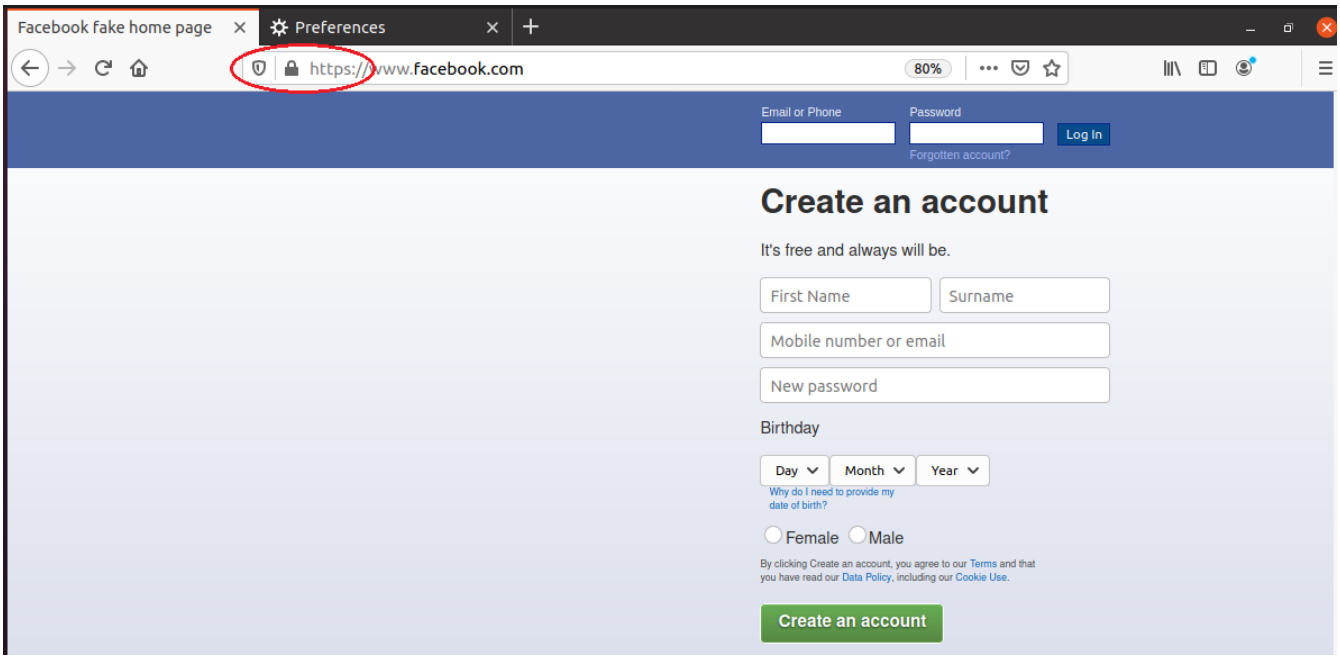
* use sudo nano /etc/hosts → add the new entry → ctrl+X → y → Enter

*The ip address would be the real public IP Address of our malicious server.



Step 10: The output of the task performed

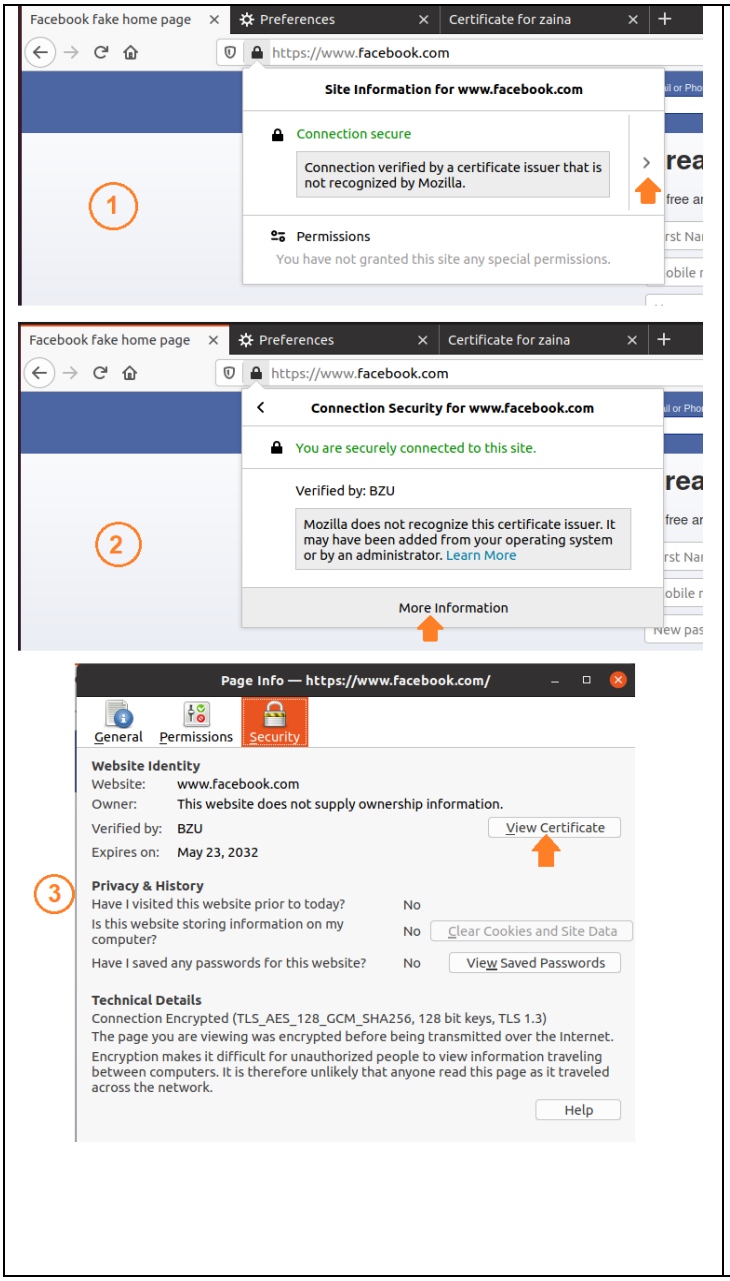
Step 10.1 Browse Https://www.facebook.com

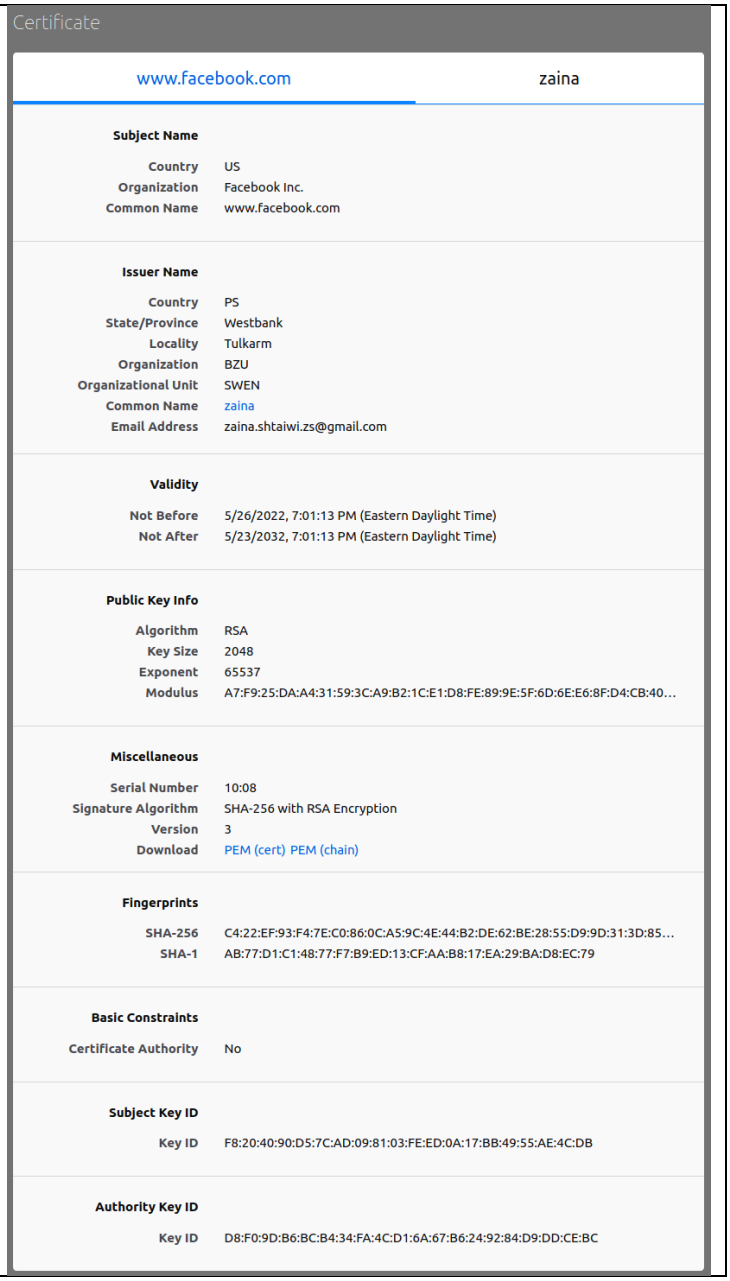


Notice the security lock that indicates that the visited site is secure. This will reduce any suspicion from the end user who is not paying close attention. Once the user's localhost is compromised, every time the user thinks they are going to the real facebook.com, the browser will take them to our malicious website.

Step 10.2: check the security information and certificate by clicking the lock before the URL

It seems that the malicious website certificate is legal and the end user will trust the website and provide his credentials and private information to the attacker.





END