LAB 9: Public – Key Infrastructure (PKI) Lab

TASK 1: Becoming a Certificate Authority (CA)

```
[04/14/19]\Shenava@VM:~$ cd /home/seed/Desktop/PKI/
[04/14/19]\Shenava@VM:~/.../PKI$ gedit openssl.cnf
```

```
####################################################################
[ ca ]
default_ca      = CA_default            # The default ca section

####################################################################
[ CA_default ]

dir             = /home/seed/Desktop/PKI # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
#unique_subject = no                    # Set to 'no' to allow creation of
                                        # several ctificates with same subject.
new_certs_dir   = $dir/newcerts         # default place for new certs.

certificate     = $dir/cacert.pem       # The CA certificate
serial          = $dir/serial           # The current serial number
crlnumber       = $dir/crlnumber        # the current crl number
                                        # must be commented out to leave a V1 CRL
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/cakey.pem# The private key
RANDFILE        = $dir/private/.rand    # private random number file

x509_extensions = usr_cert              # The extentions to add to the cert

# Comment out the following two lines for the "traditional"
# (and highly broken) format.
name_opt        = ca_default            # Subject Name options
cert_opt        = ca_default            # Certificate field options

# Extension copying option: use with caution.
# copy_extensions = copy
```

```
[04/14/19]\Shenava@VM:~/.../PKI$ mkdir certs crl newcerts
[04/14/19]\Shenava@VM:~/.../PKI$ touch index.txt serial
[04/14/19]\Shenava@VM:~/.../PKI$ echo 1000 > serial
[04/14/19]\Shenava@VM:~/.../PKI$ ls
certs  crl  index.txt  newcerts  openssl.cnf  serial
```

We copy and configure the openssl.cnf folder and make directories in our current folder.

```
[04/14/19]\Shenava@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 incorp
orated
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) []:SY
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SU
Organizational Unit Name (eg, section) []:EECS
Common Name (e.g. server FQDN or YOUR name) []:Mrudhu
Email Address []:mas@gmail.com
[04/14/19]\Shenava@VM:~/.../PKI$
```

We generate the public key certificate *ca.crt* and the private key *ca.key* for the CA using openssl and RSA command.

```
[04/14/19]\Shenava@VM:~/.../PKI$ gedit ca.key
```


```
-----BEGIN ENCRYPTED PRIVATE KEY-----
MIIFDjBABgkqhkiG9w0BBQ0wMzAbBgkqhkiG9w0BBQwwDgQI3j9E4rAJtZMCAggA
MBQGCCqGSIb3DQMHBAg6/n49ANvg5ASCBMjp9qzW2hZRsVGpmHk+HafzxEL3bwYA
UfHz4KW1PuNP8FVtl5nkQZKQIDT3P9D6ONFhGKmeq9bFzAxv9P+55hhGd1wLlXX9
GTVEW9EsN8ANoBJ1aJOy//khZzcc5Rr89wFd7Q6y8QH5KU9FJ1d2PqriZBttBRfx
qLLVwnRri7f4lRirLCsMvu1OiJjMO/TWk3lMIX+R99Tjf3InmvlqHg6mJbgCkJuz
E4rsrWmwvFiZrzrBY8r5v77VO8yr+EdXK9rvMz3N5ct2hkMVZlhkTSWuHGhCjkY1
Fw8oeHYV/xGb/D/p2mWM8XZPcoKr1bn57s3MPTNekBvxaCsMX7TZxCkeFoT/6Htk
67P8aKMIMowqFyB/NLgpdIF0OcT8fAE1OoUvu72qgFeg0el5yGFu34CJ7Wzv8ln5
RjSUFwX0n7g+Bi5+I748mul9JpSWyV1Fyh/YWsW0fo9cZtkzr6Fkc3dzjwG2dMmN
IrFoaiKMfDqAH/m8GZwHpHAMbopVJvc3JETtbgSywZMqYqX9fK+OXRnM0PhCsOec
iMO5MNWr03a+cDvEAxTRzh4VfGNdESiOcSVDVPgKxiv7n0Rb++t+/slqBsfBxlyz
W+KzkxKucba/MiNCxmGv2dgZUYcjTE1soH1yZ02L1u7vrHk4H7DXhi+8m1wvZgt+
n29zjoqpRxeFC0rcLRQ9z/DMG8Y82eTJaeXveg29ZjyzPRvZlf60QZEcG31un9dX
gLtaKnCNiHPWNupjjl93EE8Hhel992QqJyU86Tqny716V3qqTrLOasJfHw0M5h1n
3FxbrD/TFuxY+cvuXyPYDOqJBPEq1her23ztMprh1rMDtBQtxFEy2F9NUx7O4rdq
fARNfqw3ootmojT7J91HtP1NWxUS0f31fiTolAkJoUy7p2oREFwY1L8qh7ncmN7P
YoJPiid/iPaJUebdAQuMBjKtJLj+J3a5yGUeccHbDFMrqnCItbL3I0+XRncAoANh
8cDf8idOBjN+qRFiX7UMXAATNHxymn31a+DRBu+xe8S5O35OqSP1JTIYmealePZZ
341IH1bWX4jTIQ6kUCLmNG0zIXa6ab2rbpErHU/ulq3/rhFnOW/jlCyrmeXxJXD2
SGpzW4LAV1pDjqWVuN4G2mr8kvUFPnlKTl83BUXgHWgnfvqIj+L0rMSsEE+Pz7o/
JNtthf1XXI+zmjDyLYnwTNZqZfTN+o+oSplGKfREajNTEapjB3EB55wv8ZXEkfF4
JlB9qDnNNzIES3SMiOZC/KVCpO3lRpIc7HwTaCLpEhKl3FYjDWlZTiV7W7gmyZ2G
OQxPZj52fYPjkr3DdXiOZU1FCwsSoBpn6WExyMYPM1wnWE/C0ztdMORvdeW4pmAx
N+svo9BH16ZMlSqPU9XREeEUdCvx0dfMI7gmsRWBpw4LuSoEX9xWwfFtyp/Z7wC5
DpU5bcD6dIA8fVBd5jv9K2gkONsdhtzU3SnwsqOFevC8uPp2jduzLZIGnK7DlCtR
gIkaXuiPVG9GCRfKO5v3lpp+ZxGyKZRdzbyV5Nm2WOaQV6Y3D2o5be/qQ5d0HczN
9Gu8vZXUA9tGD7hnBw+3DUA+opm7bW/Kno3d5N4f8YwGmi7NLOx6O52RdnDe9S2w
8EE=
-----END ENCRYPTED PRIVATE KEY-----
```


CA’s private key

```
[04/14/19]\Shenava@VM:~/.../PKI$ gedit ca.crt
```

Mrudhu
Identity: Mrudhu
Verified by: Mrudhu
Expires: 05/15/2019

Details

Subject Name

C (Country):	US
ST (State):	NY
L (Locality):	SY
O (Organization):	SU
OU (Organizational Unit):	EECS
CN (Common Name):	Mrudhu
EMAIL (Email Address):	mas@gmail.com

Issuer Name

C (Country):	US
ST (State):	NY
L (Locality):	SY
O (Organization):	SU
OU (Organizational Unit):	EECS
CN (Common Name):	Mrudhu
EMAIL (Email Address):	mas@gmail.com

Issued Certificate

Version:	3
Serial Number:	00 D1 35 4B 90 6D 97 CC B6
Not Valid Before:	2019-04-15
Not Valid After:	2019-05-15

Certificate Fingerprints

SHA1:	1F 72 9F AA D6 AF 40 59 60 58 33 CB EF 84 0A 11 BF EA 1F 3A
MD5:	4B 91 D4 2E C5 8D 05 92 82 73 1B 8B 36 E8 BB 5B

Public Key Info

Key Algorithm:	RSA
Key Parameters:	05 00
Key Size:	2048
Key SHA1 Fingerprint:	51 3C 56 FA CB 3E F9 3E B1 B7 05 46 5C AC FA 92 27 A9 2D FF
Public Key:	30 82 01 0A 02 82 01 01 00 B4 9E A6 FE 9A 70 8A 77 96 B1 FD DF 70 26 63 19 7E 69 86 55 76 65 3E 89 51 A3 E8 4E B9 B7 28 63 59 A2 F3 EE 45 8E 89 4D E7 76 6C 39 78 32 38 8E E5 D1 53 70 9A 85 58 FB 33 81 9E 63 A5 12 59 A2 05 BD FE B0 59 95 23 41 A4 51 80 89 B2 B3 2E 9C B5 9F 00 55 45 0B A0 E5 1E D2 EB 88 B4 8B CE 70 8D 4D 9B C0 C5 CB A3 60 75 54 1E EA EF 99 43 8D 1D E5 7B 6D 5F A1 71 E8 25 A0 46 76 42 24 C8 6E 16 DB DC C6 F5 FE 50 78 29 AC 59 10 96 AD A4 9A 24 67 67 0F 9C 94 26 83 E9 6F F4 8E 28 D3 BB 0A DD 37 65 0C 49 16 33 57 1B 78 F7 34 F7 F3 D5 6C AA 38 D4 54 42 CD 12 CA EC D8 62 EB 3E 91 58 B9 F3 B8 0C 0E C8 5D 10 3A 8A 1E C1 57 41 63 CA 31 4C 42 30 7E 86 28 59 CE B6 71 04 7F 53 E9 4D AD E1 64 85 0A 45 8D 9E 30 62 41 65 C8 22 42 29 48 19 F2 15 EB 56 9C AD 3B 30 F4 61 0F EE CA 9F 75 02 03 01 00 01

Subject Key Identifier

Key Identifier:	3C 8A 5C C9 BB 11 05 62 8B 56 6C 19 8E E3 52 0C B8 F4 C5 52
Critical:	No

Extension

Identifier:	2.5.29.35
Value:	30 16 80 14 3C 8A 5C C9 BB 11 05 62 8B 56 6C 19 8E E3 52 0C B8 F4 C5 52
Critical:	No

Basic Constraints

Certificate Authority:	Yes
Max Path Length:	Unlimited
Critical:	No

Signature

Signature Algorithm:	1.2.840.113549.1.1.11
Signature Parameters:	05 00
Signature:	4A EE 74 4D 12 4A 20 3F 44 BA 46 B2 F0 69 97 45 DC E6 00 B0 5F B5 B9 9F BD 28 A6 C0 56 37 40 F2 95 35 08 8D 48 D2 B7 3D DB BE 15 FA 51 9B EF 8D F4 2C BA F6 3A C1 B0 73 14 BF 94 CA 8B E3 08 1A D4 62 1F 04 33 82 5C 0D 31 02 D8 8C E1 FC 6F EA AD F3 B7 28 BF FF 20 A6 5D FB 83 D3 B9 21 E4 EF 19 C2 1B 63 0F 12 C9 89 FD 17 E4 A2 7C 3A 6A 76 01 31 85 2B C7 B8 AD 19 37 74 ED 07 EE E4 2D 3D 9A 1C 85 C0 E1 A6 D8 2D 77 45 30 57 43 7E C8 EE 25 92 26 91 66 99 7E BB 58 9C A8 54 BE 4A 00 68 AF 67 51 9C 71 4D 48 57 0D F0 2F FB 53 EE 24 00 0D A2 E9 53 19 3C 78 13 5C 77 EA 8C 1B C8 1D BD 1A C8 9B 75 C8 6F 27 53 4C 29 EC 73 73 78 B2 C3 56 5A 09 3E DC C2 D7 25 B6 61 AF 3F 6D 2A 7A 3C D8 B9 DA 03 15 06 F3 D3 E8 84 34 2D 16 F7 BD 67 F8 40 90 DA 63 9E BA 9C BC D2 E9 88 29 6C 0B 92

Close Import

CA’s public key certificate.

TASK 2: Creating a Certificate for SEEDPKILab2018.com

Step 1: Generate public/private key pair.

```
[04/14/19]\Shenava@VM:~/.../PKI$ openssl genrsa -aes128 -out serve
r.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:
[04/14/19]\Shenava@VM:~/.../PKI$
```

```
[04/14/19]\Shenava@VM:~/.../PKI$ openssl rsa -in server.key -text
Enter pass phrase for server.key:
Private-Key: (1024 bit)
modulus:
    00:c0:9a:a5:8a:1d:2f:10:64:b2:21:85:9f:e7:72:
    90:90:d7:73:73:69:e9:56:80:1c:b0:15:bd:1a:95:
    b5:ab:1f:ee:17:37:5d:aa:78:20:20:82:e2:7b:f6:
    50:83:49:16:c9:47:03:23:95:3e:2d:60:ad:94:f5:
    da:90:ce:7c:75:e7:9e:df:57:95:7a:e5:3a:e4:91:
    b9:77:a8:07:b5:75:82:8d:94:dd:96:0f:fa:d4:c0:
    f0:45:14:56:98:b2:9e:f0:c1:7d:13:b2:ac:46:c8:
    3a:7c:be:85:58:91:58:c9:ef:fc:07:33:de:a4:46:
    74:66:65:6b:4d:d9:af:71:0f
publicExponent: 65537 (0x10001)
privateExponent:
    7a:2c:9e:fd:71:6d:bd:54:62:ba:68:ec:f9:ff:ee:
    3c:32:af:05:73:15:94:cd:47:65:5b:24:cd:8c:51:
    e4:ef:28:34:89:e6:5c:90:3e:b0:e0:d5:92:14:b9:
    15:ce:73:29:d2:65:e7:58:70:4c:90:9b:0b:76:b8:
    e2:4a:59:1f:c7:a3:50:f2:b0:ab:b5:b7:51:9e:93:
    ea:45:c0:fd:a8:ea:2e:17:8a:13:79:67:fb:32:1d:
    1a:a1:70:90:b1:94:80:3a:86:1e:dd:28:66:1b:5f:
    a9:03:aa:bf:f2:56:9c:7d:56:8b:a0:3c:a2:f9:88:
    47:ea:e8:9b:32:b8:c7:e1
prime1:
    00:e5:90:12:6d:77:93:85:b3:56:29:bb:dc:6f:70:
    99:e4:27:60:46:7c:96:ec:90:af:64:aa:f6:94:0f:
    44:d4:b4:4a:35:bf:b5:d4:d0:f5:cb:7e:72:d0:e6:
    15:ce:40:1b:15:5d:38:00:bd:20:cc:ea:87:0f:d5:
    3d:64:dd:ba:eb
```

```
prime2:
    00:d6:c8:f7:18:c2:81:22:c4:b8:b6:55:e1:ce:55:
    1f:4c:d4:bd:e8:eb:36:70:b6:8c:f3:17:24:4a:76:
    75:c5:98:f5:b8:10:12:7f:4c:94:22:a8:28:8c:c5:
    4b:7f:ef:16:3c:12:01:d3:6f:22:3d:31:e0:ca:e7:
    ae:7f:c7:d1:6d
exponent1:
    00:bd:3b:48:e5:71:a7:6a:94:0d:57:b3:a3:1f:3d:
    bb:ad:86:03:e7:b6:b6:16:65:a2:1f:22:8a:c6:3b:
    db:ea:56:7f:c1:d4:b9:47:a7:c4:a5:ad:23:6a:2d:
    f0:15:a7:64:6b:f4:fe:c0:25:27:0f:a3:d8:17:e6:
    18:88:b7:fa:67
exponent2:
    00:94:29:f3:d2:bf:b2:47:62:bf:2d:00:d7:90:f7:
    45:6d:41:c0:73:e1:76:1f:90:be:70:9e:c1:29:20:
    c8:a3:73:9d:e5:c8:28:8a:6a:30:32:2c:20:34:e3:
    90:ee:82:e6:aa:af:9d:54:d6:70:1f:18:be:42:1a:
    4a:54:d9:69:8d
coefficient:
    36:56:97:3d:58:f9:ec:1e:81:73:8a:fa:86:c6:35:
    f1:c7:ca:9c:53:59:54:f3:de:55:69:13:c8:bb:35:
    70:ca:1d:36:dd:80:47:ed:5f:0e:4f:cf:ae:77:95:
    3d:ab:1d:73:ea:c2:8b:69:60:16:9f:76:8e:7e:7b:
    01:05:af:4a
writing RSA key
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDAmqWKHS8QZLIhhZ/ncpCQ13NzaelWgBywFb0albWrH+4XN12q
eCAgguJ79lCDSRbJRwMjlT4tYK2U9dqQznx1557fV5V65Trkkbl3qAe1dYKNlN2W
D/rUwPBFFFaYsp7wwX0TsqxGyDp8voVYkVjJ7/wHM96kRnRmZWtN2a9xDwIDAQAB
AoGAeiye/XFtvVRiumjs+f/uPDKvBXMVlM1HZVskzYxR5O8oNInmXJA+sODVkhS5
Fc5zKdJl51hwTJCbC3a44kpZH8ejUPKwq7W3UZ6T6kXA/ajqLheKE3ln+zIdGqFw
kLGUgDqGHt0oZhtfqQOqv/JWnH1Wi6A8ovmIR+romzK4x+ECQQDlkBJtd5OFs1Yp
u9xvcJnkJ2BGfJbskK9kqvaUD0TUtEo1v7XU0PXLfnLQ5hXOQBsVXTgAvSDM6ocP
1T1k3brrAkEA1sj3GMKBIsS4tlXhzlUfTNS96Os2cLaM8xckSnZ1xZj1uBASf0yU
IqgojMVLf+8WPBIB028iPTHgyueuf8fRbQJBAL07SOVxp2qUDVezox89u62GA+e2
thZloh8iisY72+pWf8HUuUenxKWtI2ot8BWnZGv0/sAlJw+j2BfmGIi3+mcCQQCU
KfPSv7JHYr8tANeQ90VtQcBz4XYfkL5wnsEpIMijc53lyCiKajAyLCA045Duguaq
r51U1nAfGL5CGkpU2WmNAkA2Vpc9WPnsHoFzivqGxjXxx8qcU1lU895VaRPIuzVw
yh023YBH7V8OT8+ud5U9qx1z6sKLaWAWn3aOfnsBBa9K
-----END RSA PRIVATE KEY-----
[04/14/19]\Shenava@VM:~/.../PKI$
```

We generate the public/private RSA key pair using openssl and it is stored in *server.key*.

Step 2: Generate a Certificate Signing Request (CSR).

```
[04/17/19]\Shenava@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 incorp
orated
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) []:SY
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SU
Organizational Unit Name (eg, section) []:EECS
Common Name (e.g. server FQDN or YOUR name) []:SEEDPKILab2018.com
Email Address []:seed@labs.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
[04/17/19]\Shenava@VM:~/.../PKI$
```

We generate the Certificate Signing Request (CSR) using *server.key* which is used to generate the certificate for the key.

Step 3: Generating Certificates.

```
[04/17/19]\Shenava@VM:~/.../PKI$ openssl ca -in server.csr -out se
rver.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: 4096 (0x1000)
        Validity
            Not Before: Apr 17 17:35:11 2019 GMT
            Not After : Apr 16 17:35:11 2020 GMT
        Subject:
            countryName               = US
            stateOrProvinceName       = NY
            organizationName          = SU
            organizationalUnitName    = EECS
            commonName                = SEEDPKILab2018.com
            emailAddress              = seed@labs.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                32:D4:47:F2:02:AC:06:93:A1:18:21:8D:AE:5D:60:BE:21
:AA:16:74
            X509v3 Authority Key Identifier:
                keyid:C9:FC:D9:5F:F7:72:6A:53:FA:8F:D9:6B:A8:E5:E8
:CF:CC:B9:78:11

Certificate is to be certified until Apr 16 17:35:11 2020 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
[04/17/19]\Shenava@VM:~/.../PKI$
```

The CSR file is sent to the CA and it is signed using the *ca.crt* and *ca.key* to generate *server.crt* certificate for SEEDPKILab2018.com

TASK 3: Deploying Certificate in an HTTPS Web Server

Step 1: Configuring DNS.

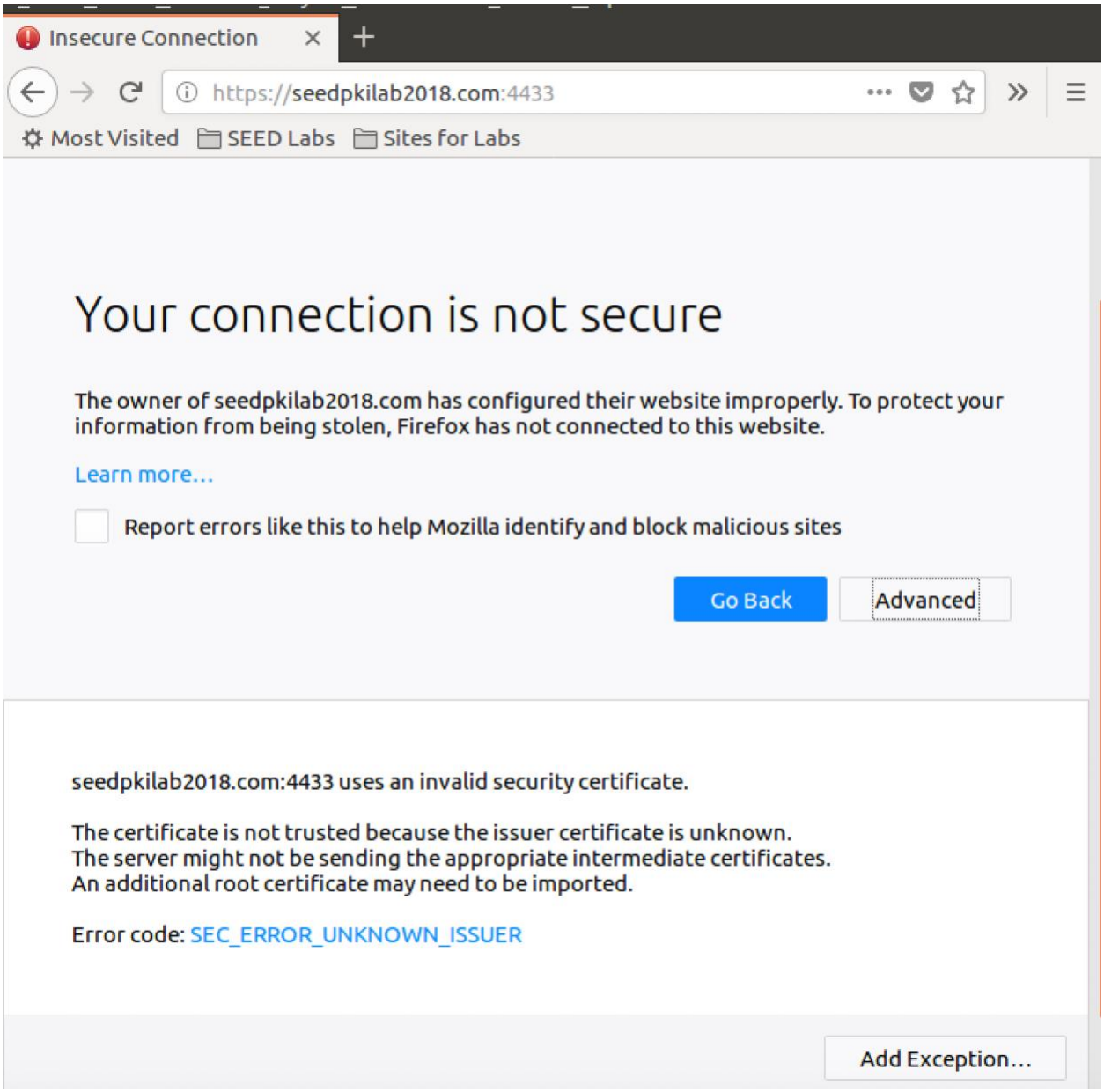
```
[04/17/19]\Shenava@VM:~$ cd /etc/
[04/17/19]\Shenava@VM:/etc$ sudo gedit hosts
[sudo] password for seed:
```

```
Open ▾                    hosts
                          /etc
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       User
127.0.0.1       Attacker
127.0.0.1       Server
127.0.0.1       www.SeedLabSQLInjection.com
127.0.0.1       www.xsslabelgg.com
127.0.0.1       www.csrflabelgg.com
127.0.0.1       www.csrflabattacker.com
127.0.0.1       www.repackagingattacklab.com
127.0.0.1       www.seedlabclickjacking.com
127.0.0.1       SEEDPKILab2018.com
```

We configure the hosts file such that our computer recognizes the name of our website. This entry maps the hostname SEEDPKILab2018.com to our localhost.

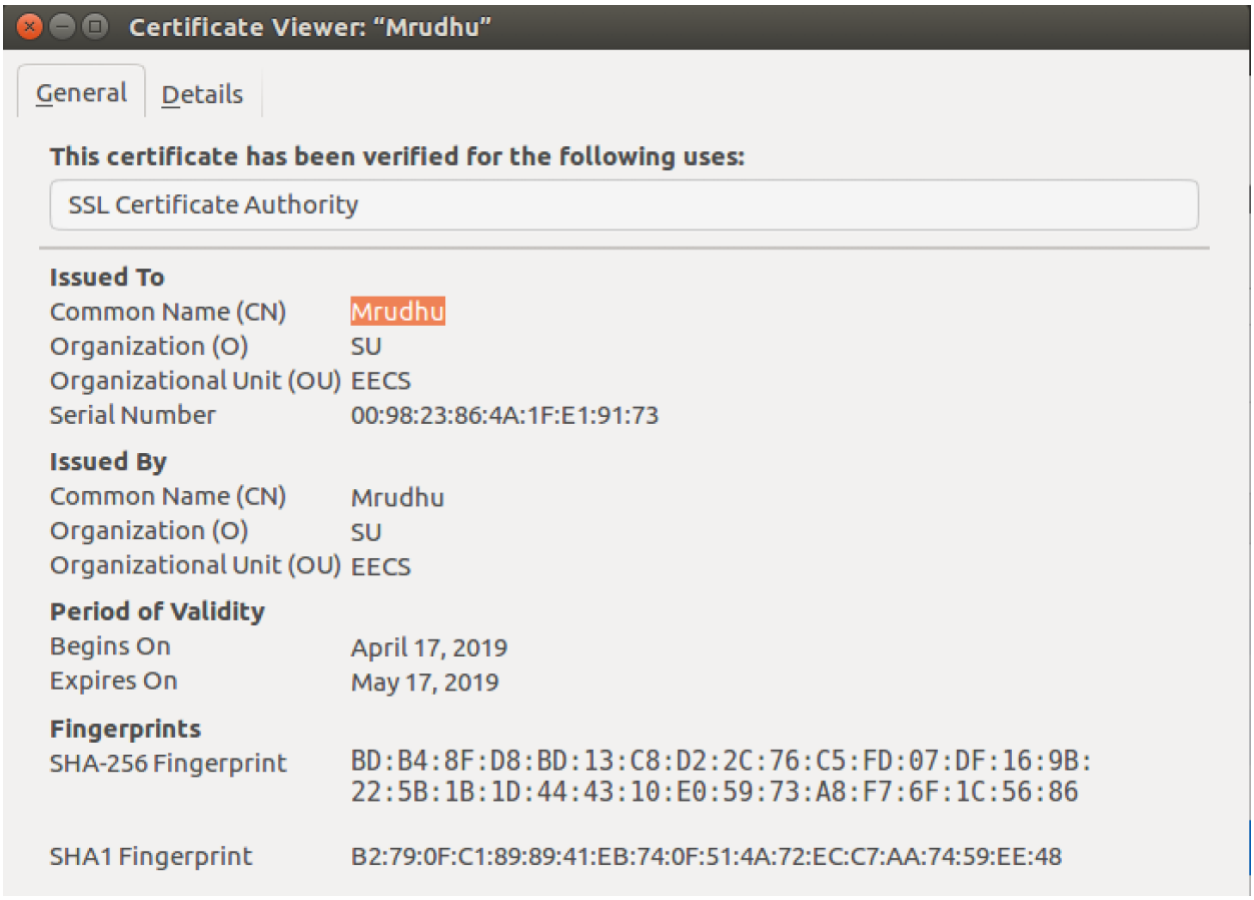
Step 2: Configuring the web server.

```
[04/17/19]\Shenava@VM:/$ cd /home/seed/Desktop/PKI/
[04/17/19]\Shenava@VM:~/.../PKI$ cp server.key server.pem
[04/17/19]\Shenava@VM:~/.../PKI$ cat server.crt >> server.pem
[04/17/19]\Shenava@VM:~/.../PKI$ openssl s_server -cert server.pem
 -www
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
```



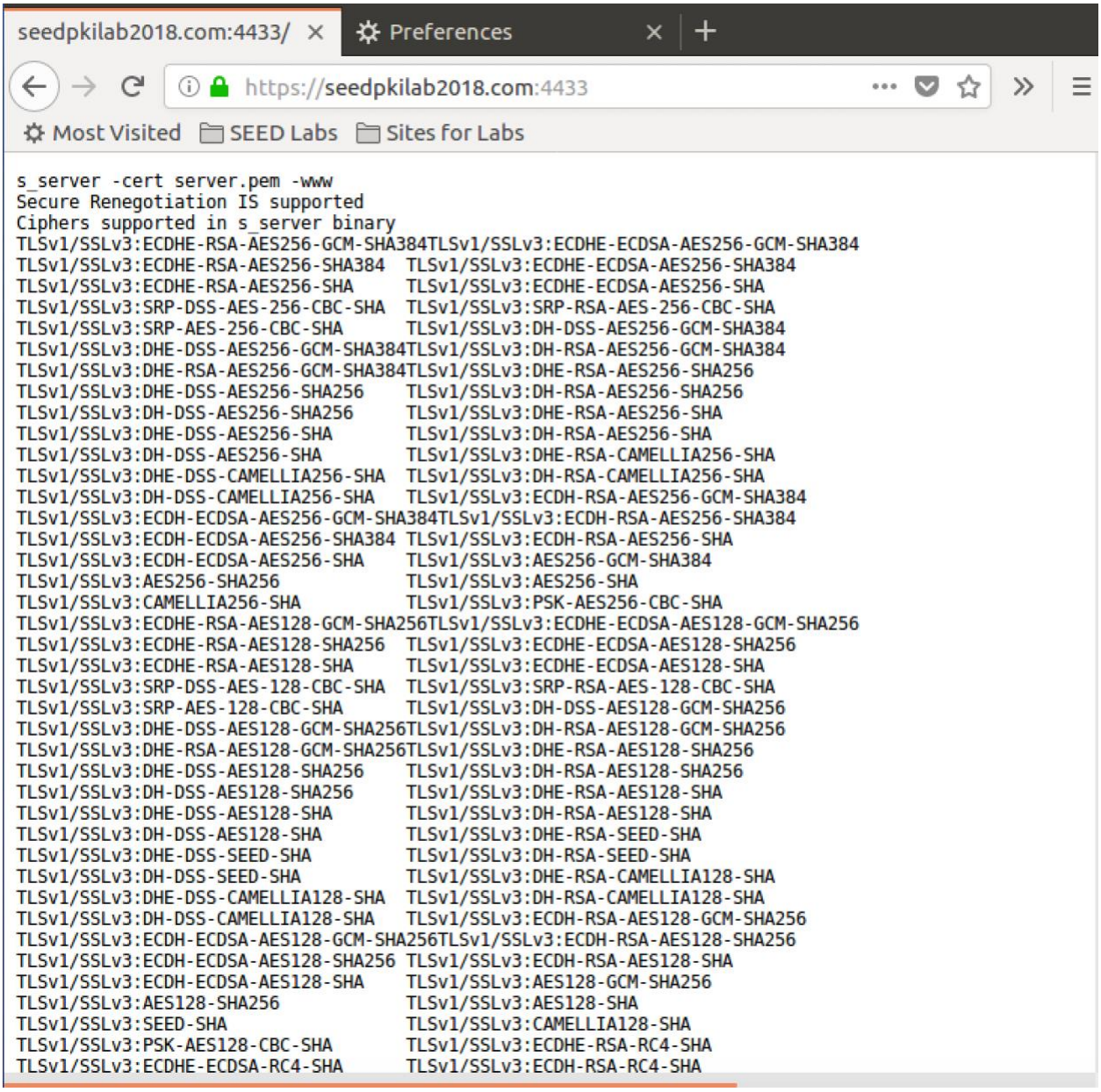
The server listens on port 4433 by default. When we try to access *https://SEEDPKILab2018.com:4433/* we notice that the site throws a message stating the site is not secure. Hence, the access to the server using the URL is not trusted by the web browser since the certificate is not assigned by any trusted company.

Step 3: Getting the browser to accept our CA certificate.



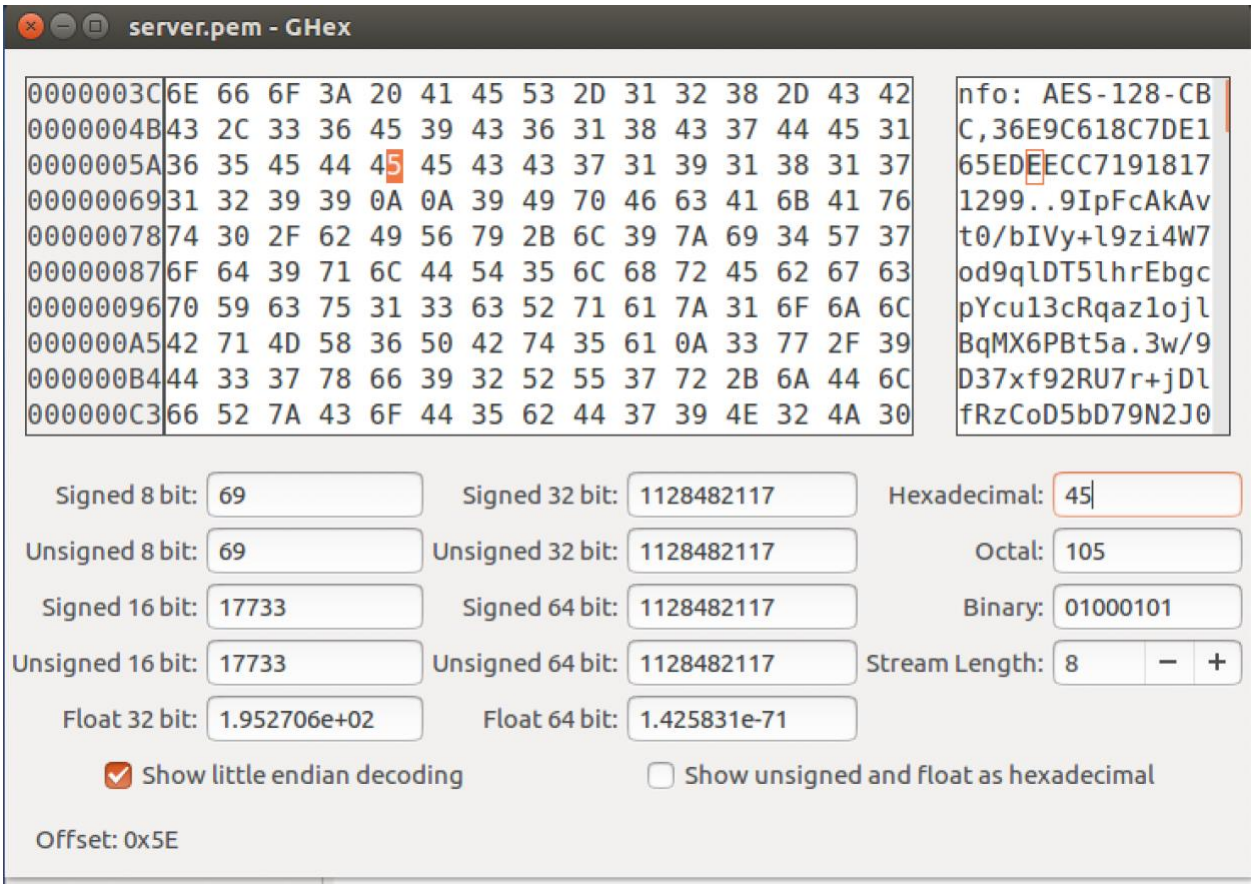
We manually add the ca.crt into the web browser.

Step 4: Testing our HTTPS website.

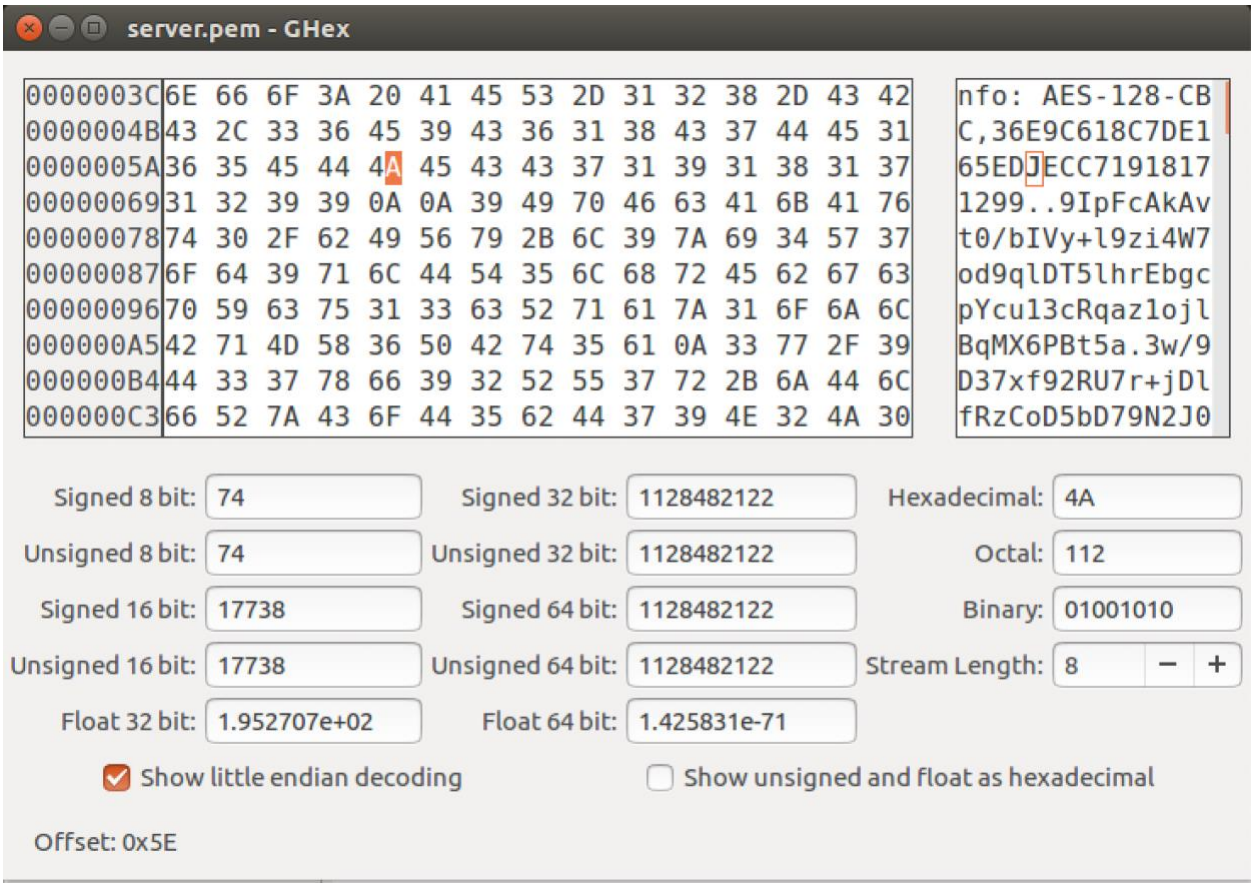


When we point the browser to https://SEEDPKILab2018.com:4433/ we notice that the URL is agreed as trusted since it is certified by *ca.crt*.

Q1. Modify a single byte of server.pem, and restart the server, and reload the URL. What do you observe? Make sure you restore the original server.pem afterward. Note: the server may not be able to restart if certain places of server.pem is corrupted; in that case, choose another place to modify.



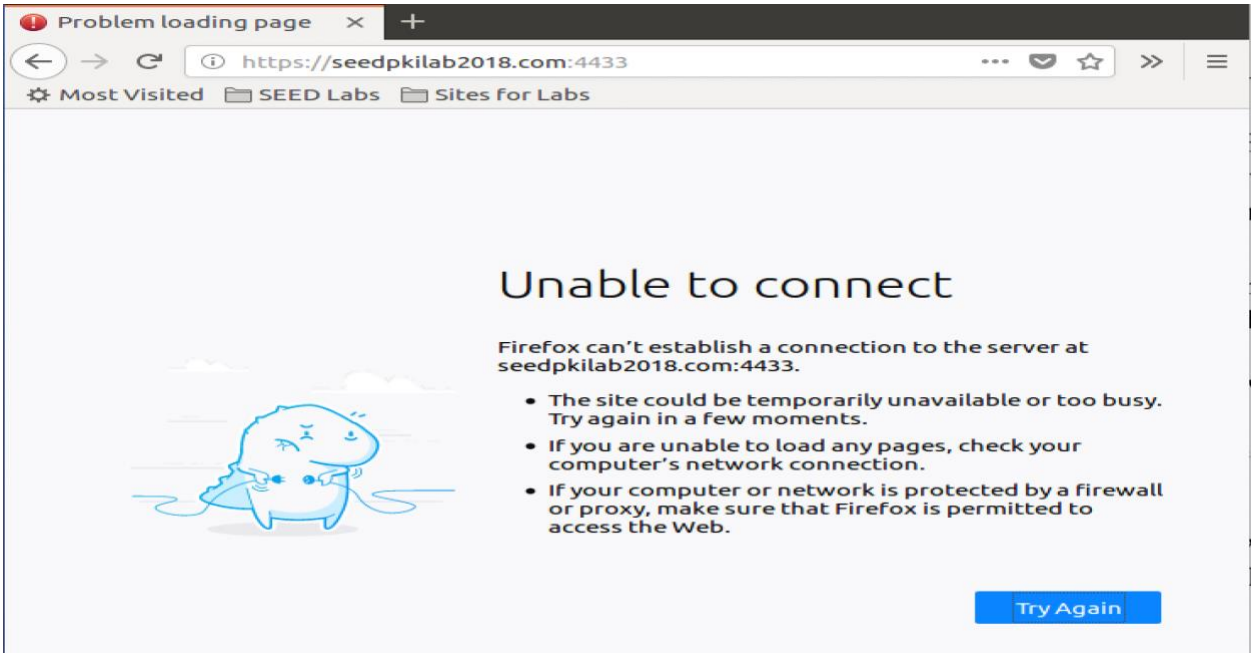
Original server.pem file



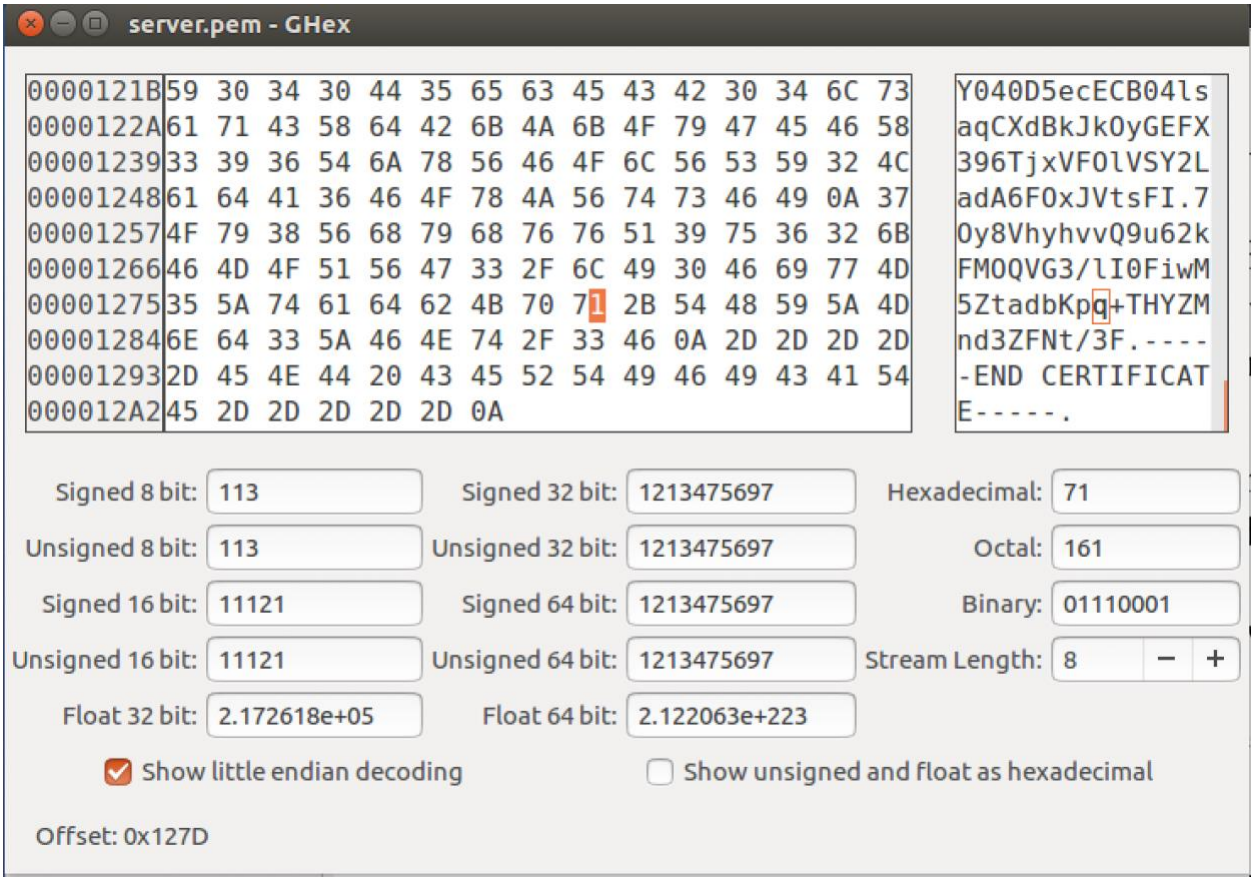
The corrupted file of server.pem after modifying the single byte.

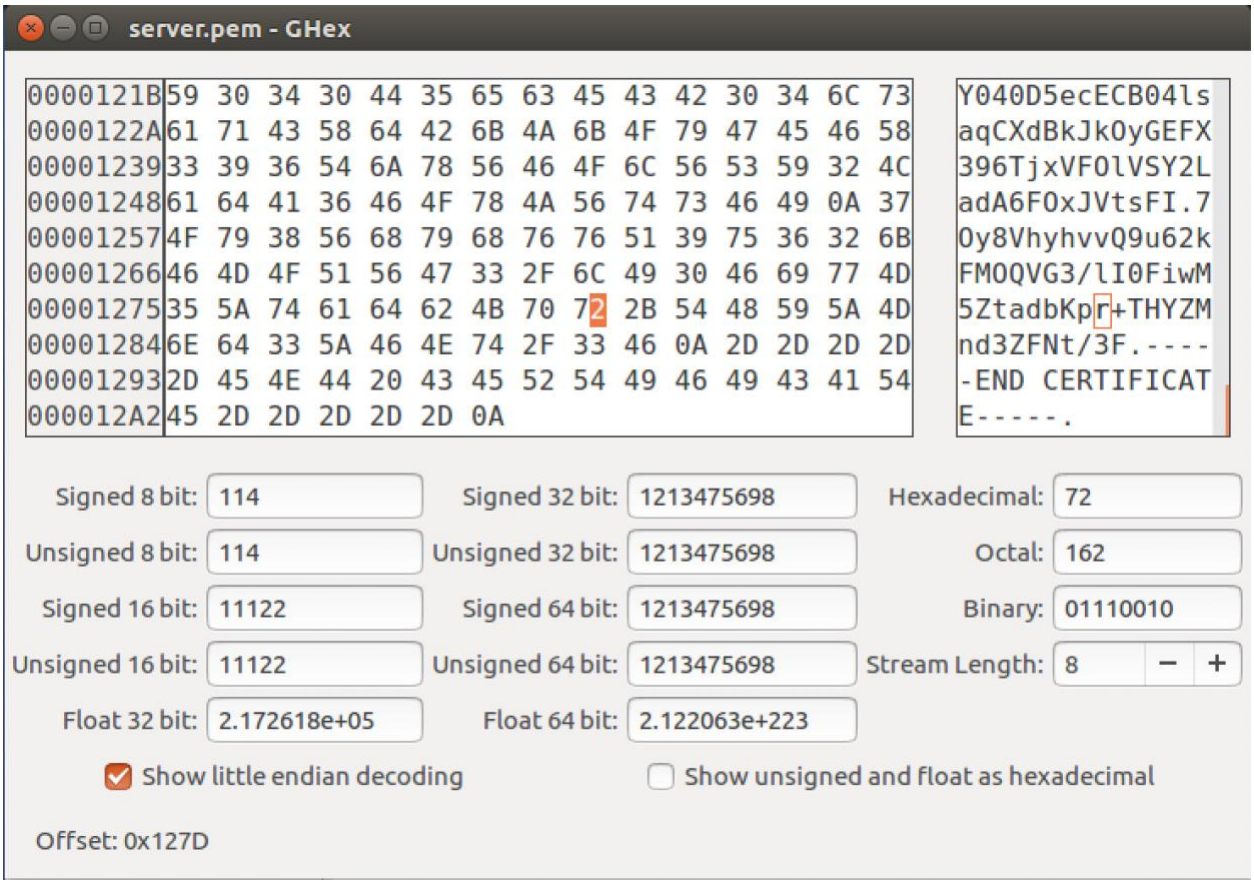
```
[04/17/19]\Shenava@VM:~/.../PKI$ openssl s_server -cert server.pem
 -www
unable to load server certificate private key file
3070654144:error:09065067:PEM routines:LOAD_IV:bad iv chars:pem_li
b.c:570:
[04/17/19]\Shenava@VM:~/.../PKI$
```

When we restart the server it throws an error.



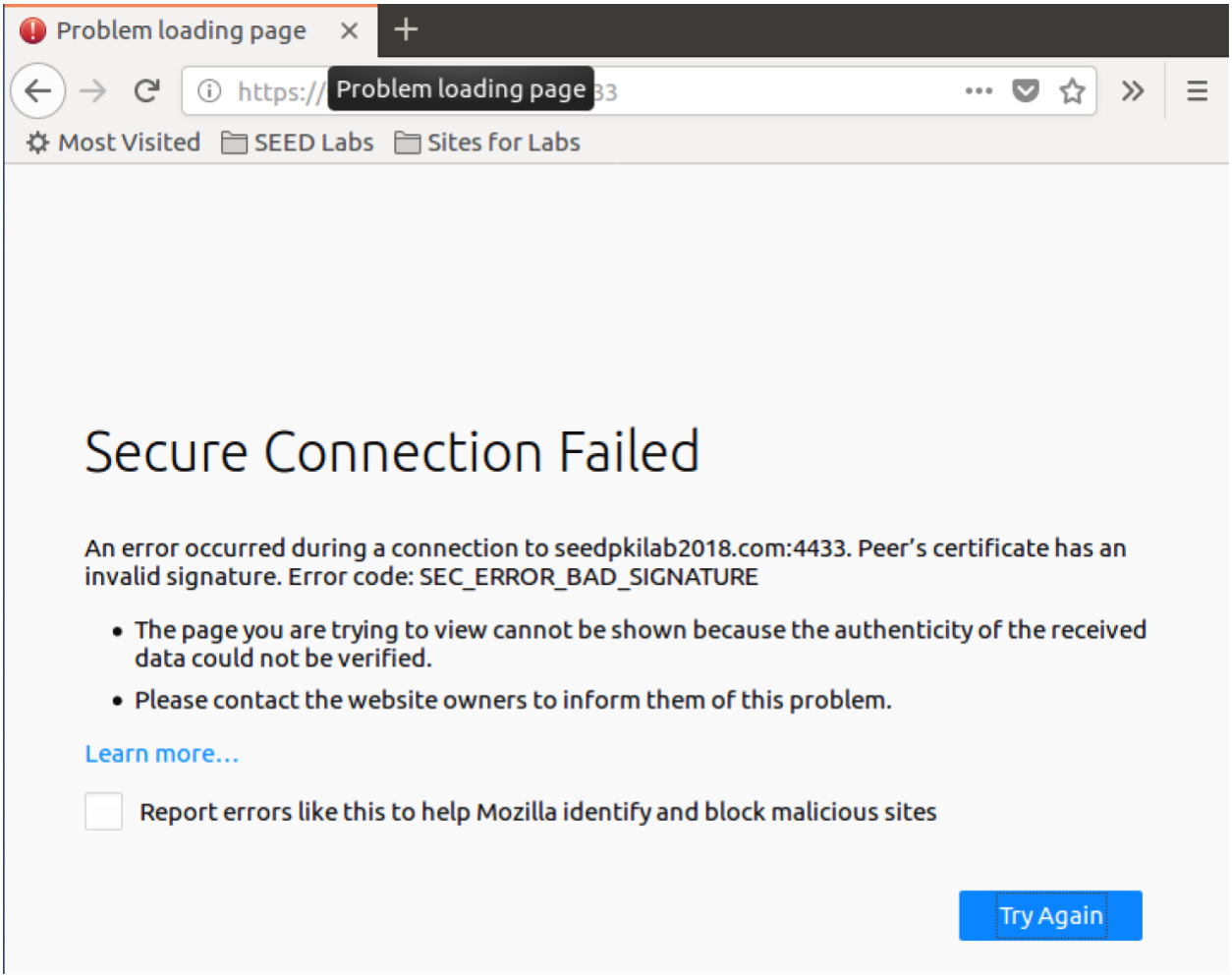
The connection cannot be established to SEEDPKILab2018.com since the server certificate private key is corrupted.





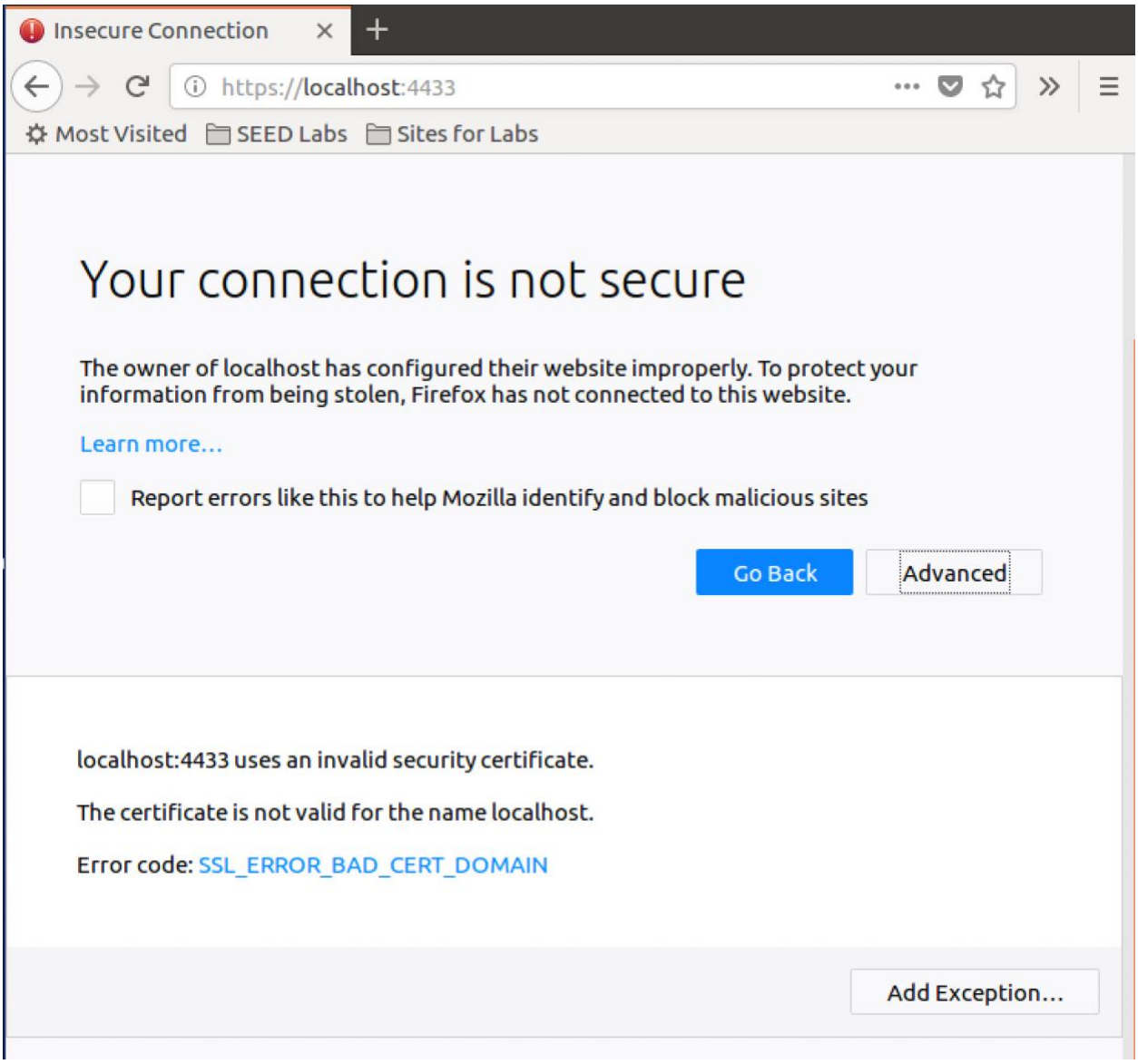
We change the single byte belonging to *server.crt* in *server.pem*.

```
[04/17/19]\Shenava@VM:~/.../PKI$ openssl s_server -cert server.pem
 -www
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
```

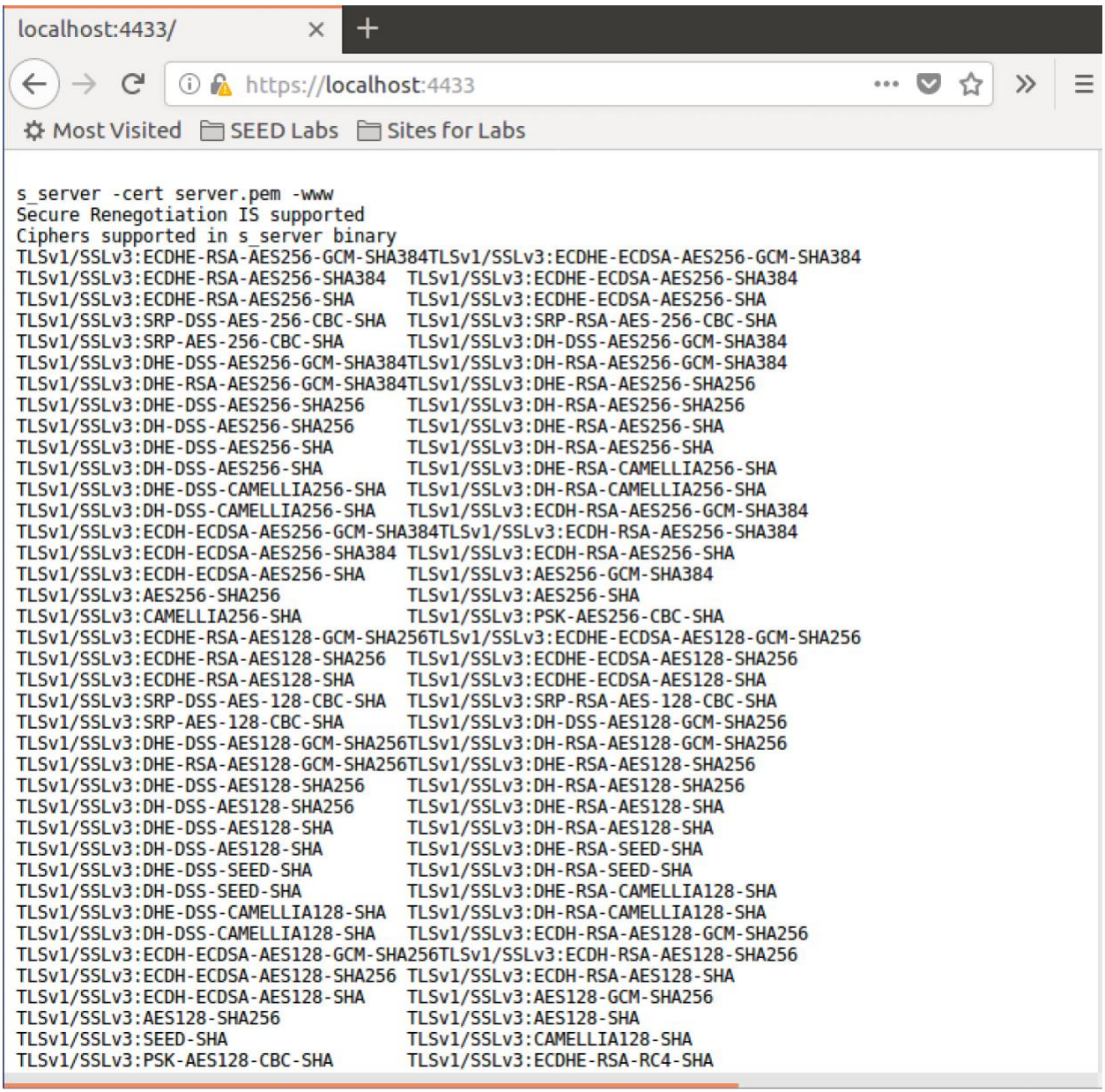


The server shows no error and launches as the server private key is not corrupted but still the web browser will not authenticate the server's certificate.

Q2. Since SEEDPKILab2018.com points to the localhost, if we use https://localhost:4433 instead, we will be connecting to the same web server. Please do so, describe and explain your observations.



The SEEDPKILab2018.com is replaced with the localhost in the URL and the server is access again. Even though both SEEDPKILab2018.com and localhost are same i.e., 127.0.0.1, the certificate is associated with the name SEEDPKILab2018.com. Hence, it fails to trust the name localhost.



After the exception is added to the name localhost the web browser trusts its certificate.

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

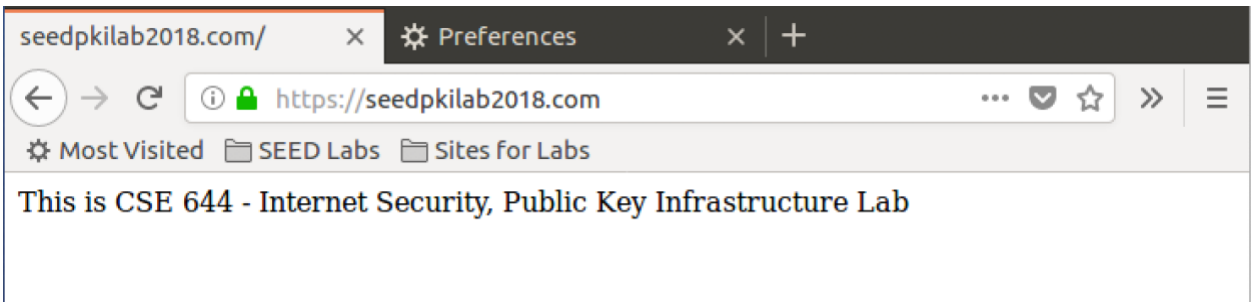
```
[04/17/19]\Shenava@VM:~$ cd /etc/apache2/sites-available/
[04/17/19]\Shenava@VM:.../sites-available$ sudo gedit default-ssl.
conf
[sudo] password for seed:
```

```
Open ▾   default-ssl.conf
         /etc/apache2/sites-available   Save
violates
        #    the SSL/TLS standard but is needed for some brain-dead browsers. Use
        #    this when you receive I/O errors because of the standard approach
where
        #    mod_ssl sends the close notify alert.
        #   o ssl-accurate-shutdown:
        #    This forces an accurate shutdown when the connection is closed,
i.e. a
        #    SSL close notify alert is send and mod_ssl waits for the close
notify
        #    alert of the client. This is 100% SSL/TLS standard compliant, but in
        #    practice often causes hanging connections with brain-dead browsers.
Use
        #    this only for browsers where you know that their SSL implementation
        #    works correctly.
        #   Notice: Most problems of broken clients are also related to the HTTP
        #   keep-alive facility, so you usually additionally want to disable
        #   keep-alive for those clients, too. Use variable "nokeepalive" for
this.
        #   Similarly, one has to force some clients to use HTTP/1.0 to
workaround
        #   their broken HTTP/1.1 implementation. Use variables "downgrade-1.0"
and
        #   "force-response-1.0" for this.
        # BrowserMatch "MSIE [2-6]" \
        #       nokeepalive ssl-unclean-shutdown \
        #       downgrade-1.0 force-response-1.0

    </VirtualHost>

    <VirtualHost *:443>
        ServerName SEEDPKILab2018.com
        DocumentRoot /var/www/SEEDPKI
        DirectoryIndex index.html
        SSLEngine On
        SSLCertificateFile /home/seed/Desktop/PKI/server.crt
        SSLCertificateKeyFile /home/seed/Desktop/PKI/server.key
    </VirtualHost>

</IfModule>
Plain Text ▾   Tab Width: 4 ▾   Ln 131, Col 1   INS
```

Since we are working with HTTPS website, we add a VirtualHost entry to the default-ssl.conf file located in /etc/apache2/sites-available. We provide the Server Name as SEEDPKILab2018.com and the we give the path of the server certificate and the private key.

```
[04/17/19]\Shenava@VM:~$ cd /etc/apache2/sites-available/
[04/17/19]\Shenava@VM:.../sites-available$ sudo apachectl configte
st
[sudo] password for seed:
AH00112: Warning: DocumentRoot [/var/www/seedlabclickjacking] does
 not exist
AH00558: apache2: Could not reliably determine the server's fully
qualified domain name, using 127.0.1.1. Set the 'ServerName' direc
tive globally to suppress this message
Syntax OK
[04/17/19]\Shenava@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
[04/17/19]\Shenava@VM:.../sites-available$ sudo a2ensite default-s
sl
Site default-ssl already enabled
[04/17/19]\Shenava@VM:.../sites-available$ sudo service apache2 re
start
Enter passphrase for SSL/TLS keys for SEEDPKILab2018.com:443 (RSA)
: ********
[04/17/19]\Shenava@VM:.../sites-available$
```

We run a series of command to enable SSL.



When we search for *https://SEEDPKILab2018.com.* we get the contents of the html we provided in the apache configuration. This shows that we were able to successfully setup our website.

```
Open ▾   index.html
         /var/www/SEEDPKI                                   Save
<!DOCTYPE html>
<html>
        <body>
                <p> This is CSE 644 - Internet Security, Public Key
Infrastructure Lab </p>
        </body>
</html>
```

The HTML file content that we stored in our apache configuration file

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

Step 1: Setting up the malicious website.

```
Open ▾   default-ssl.conf
         /etc/apache2/sites-available                       Save
where
        #    mod_ssl sends the close notify alert.
        #   o ssl-accurate-shutdown:
        #    This forces an accurate shutdown when the connection is closed,
i.e. a
        #    SSL close notify alert is send and mod_ssl waits for the close
notify
        #    alert of the client. This is 100% SSL/TLS standard compliant, but in
        #    practice often causes hanging connections with brain-dead browsers.
Use
        #    this only for browsers where you know that their SSL implementation
        #    works correctly.
        #   Notice: Most problems of broken clients are also related to the HTTP
        #   keep-alive facility, so you usually additionally want to disable
        #   keep-alive for those clients, too. Use variable "nokeepalive" for
this.
        #   Similarly, one has to force some clients to use HTTP/1.0 to
workaround
        #   their broken HTTP/1.1 implementation. Use variables "downgrade-1.0"
and
        #   "force-response-1.0" for this.
        # BrowserMatch "MSIE [2-6]" \
        #       nokeepalive ssl-unclean-shutdown \
        #       downgrade-1.0 force-response-1.0

    </VirtualHost>

    <VirtualHost *:443>
        #ServerName SEEDPKILab2018.com
        ServerName facebook.com
        DocumentRoot /var/www/SEEDPKI
        DirectoryIndex index.html
        SSLEngine On
        SSLCertificateFile /home/seed/Desktop/PKI/server.crt
        SSLCertificateKeyFile /home/seed/Desktop/PKI/server.key
    </VirtualHost>

</IfModule>

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

As we had already set up the Apache server in task 4, we only change the server name to the website we select. In this case the website I have selected is facebook.com and rest of the configuration remain the same.

Step 2: Becoming the man in the middle.

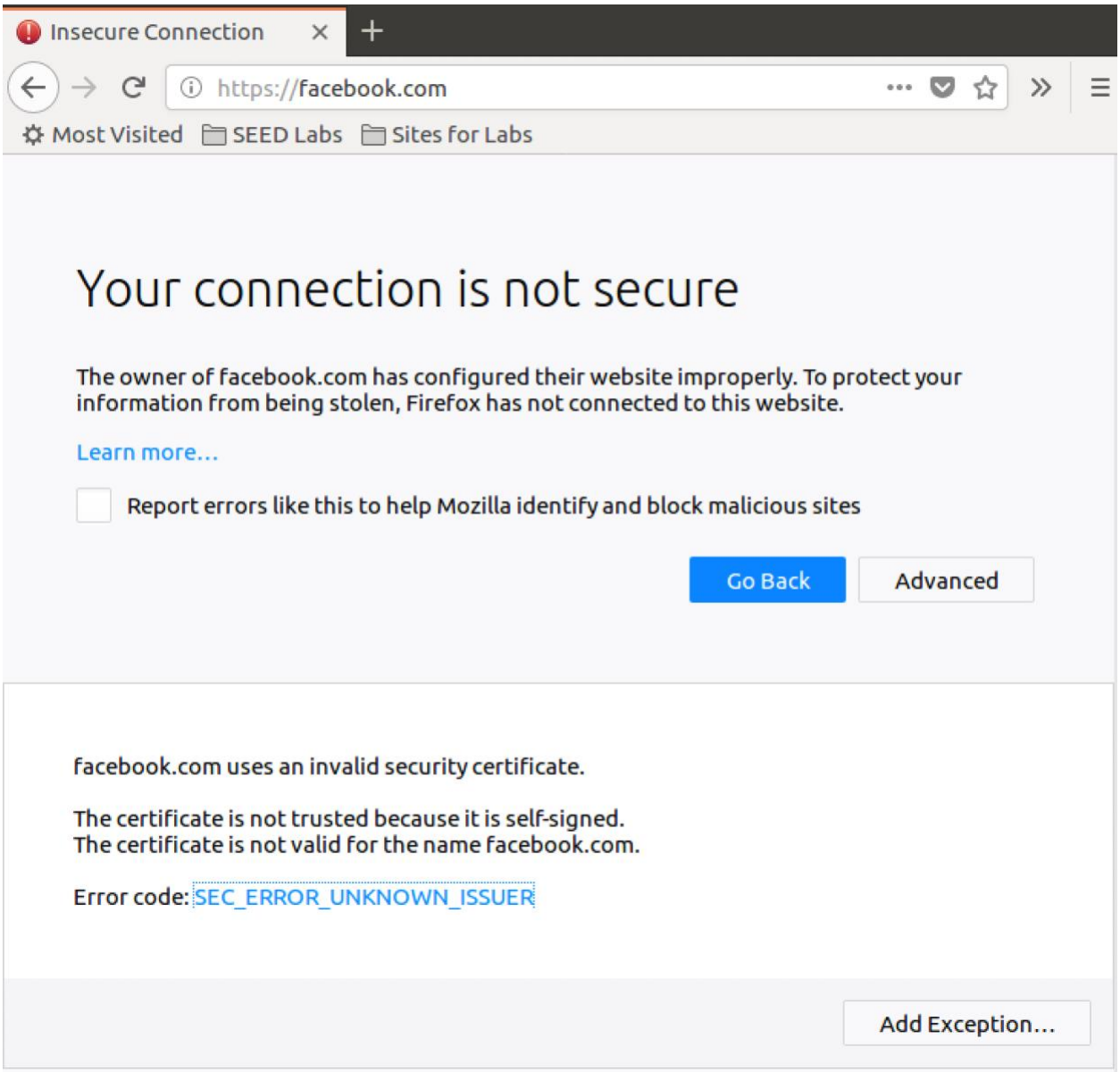
```
[04/17/19]\Shenava@VM:~$ cd /etc/
[04/17/19]\Shenava@VM:/etc$ sudo gedit hosts
[sudo] password for seed:
```

Open hosts /etc Save

```
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       User
127.0.0.1       Attacker
127.0.0.1       Server
127.0.0.1       www.SeedLabSQLInjection.com
127.0.0.1       www.xsslabelgg.com
127.0.0.1       www.csrflabelgg.com
127.0.0.1       www.csrflabattacker.com
127.0.0.1       www.repackagingattacklab.com
127.0.0.1       www.seedlabclickjacking.com
#127.0.0.1      SEEDPKILab2018.com
127.0.0.1       facebook.com
```

In the victim machine we set up the IP address of the attacker machine from where the apache of the website we want to redirect will be provide in the hosts file

Step 3: Browse the target website.

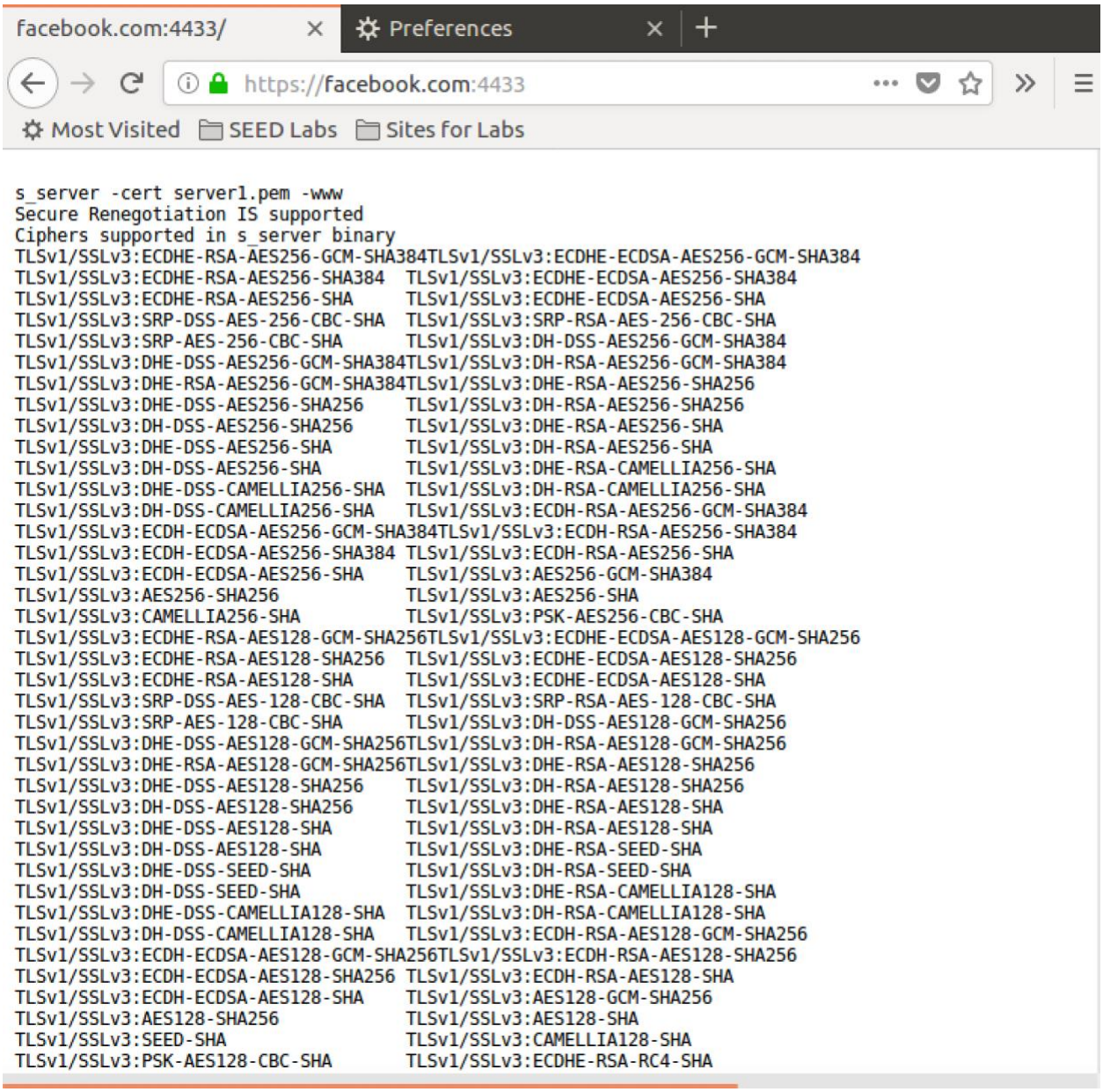


Here we notice that PKI defeats Man-In-The-Middle attack as it states that the certificate is not recognized and not authentic and throws the exception.

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

In this task, we do all the steps from task 2 and create a self-signed certificate for our chosen website that is the facebook.com.

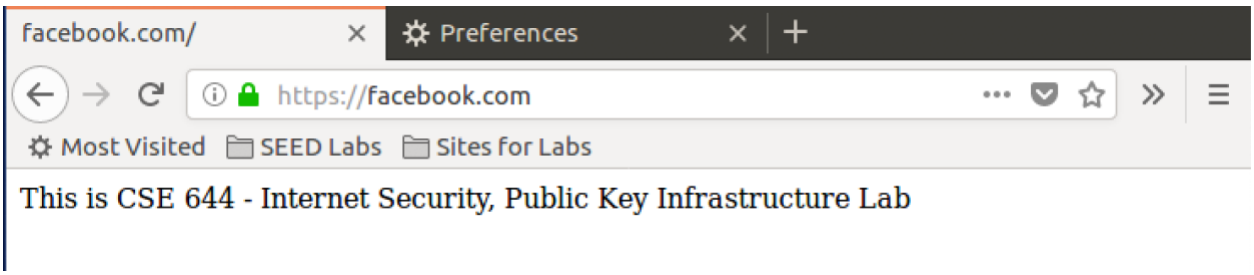
The access to the server using the URL is not trusted by the web browser since the certificate is not assigned by any trusted company.



When we point the browser to https://facebook.com/ we notice that the URL is agreed as trusted since it is certified by *ca.crt*.

```
[04/19/19]\Shenava@VM:~/.../PKI$ sudo apachectl configtest
AH00112: Warning: DocumentRoot [/var/www/seedlabclickjacking] does
 not exist
AH00558: apache2: Could not reliably determine the server's fully
qualified domain name, using 127.0.1.1. Set the 'ServerName' direc
tive globally to suppress this message
Syntax OK
[04/19/19]\Shenava@VM:~/.../PKI$ 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
[04/19/19]\Shenava@VM:~/.../PKI$ sudo a2ensite default-ssl
Site default-ssl already enabled
[04/19/19]\Shenava@VM:~/.../PKI$ sudo service apache2 restart
Enter passphrase for SSL/TLS keys for facebook.com:443 (RSA): ****
****
[04/19/19]\Shenava@VM:~/.../PKI$
```

We restart the apache server.



We notice that MITM attack is successful and the victim see what the attacker wants to display. And the browser does not raise any suspicions hen victim tries to visit the website.