

# **Public Key Encryption**

*Seed Lab*

By Aneesh Nagdev

October 10, 2025

DNS setup ss:

```
seed@VM: ~/.../Labsetup
[10/10/25] seed@VM:~/.../Labsetup$ sudo nano /etc/hosts
```

Screenshot 2:

```
# For Shellshock Lab
10.9.0.80      www.seedlab-shellshock.com
10.9.0.80      www.aneesh2025.com

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify
^X Exit        ^R Read File  ^\ Replace   ^U Paste Text ^T To Spell
```

Task 1:

Uncommenting Unique subject:

```
[ CA_default ]

dir          = ./demoCA                      # 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 certs with same subject
```

Creating Serial file ss:

```
seed@VM: ~/.../Labsetup
[10/10/25] seed@VM:~/.../Labsetup$ echo 1000 > serial
```

Generating self signed certificates for the CA ss:

```
[10/10/25] seed@VM:~/.../Labsetup$ openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -keyout ca.key -out ca.crt -subj "/CN=www.modelCA.com/0=Model CA LTD./C=US" -passout pass:dees
Generating a RSA private key
+++++
writing new private key to 'ca.key'
-----
req: Skipping unknown attribute "0"
```

This command built my root ca pair: a private key (ca.key) and its self-signed certificate (ca.crt). The password was set to dees as it can be seen in the screenshot above.

### Running command:

```
seed@VM: ~/.../Labsetup$ openssl x509 -in ca.crt -text -noout
```

Opening the ca.crt file which contains all the details about the certificate in a readable format. You can see who issued it, who it's for, validity dates. Every single detail can be seen in screenshots below.

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

```
-----  
2C:AF:3D:2B:64  
  
X509v3 Basic Constraints: critical  
CA:TRUE  
-----
```

**Observation:** It can clearly be seen that CA: TRUE which shows that CA's certificate was successfully created.

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

```
-----  
b:b4  
Signature Algorithm: sha256WithRSAEncryption  
Issuer: CN = www.modelCA.com, C = US  
Validity  
    Not Before: Oct 10 08:36:51 2025 GMT  
    Not After : Oct  8 08:36:51 2035 GMT  
Subject: CN = www.modelCA.com, C = US  
Subject Public Key Info:  
-----
```

#### Key Screenshot:

```
X509v3 Subject Key Identifier:  
DC:75:56:7A:B4:18:FE:1C:1A:27:76:2B:C5:95:09:A4:41:  
1A:54:3B  
X509v3 Authority Key Identifier:  
keyid:DC:75:56:7A:B4:18:FE:1C:1A:27:76:2B:C5:95:09:  
A4:41:1A:54:3B  
-----
```

**Explanation:** Because the Issuer and Subject are the same that means it is a self signed certificate. Also, when we confirm the keys for both are identical, that confirms it is a self signed certificate.

Complete output ss for above command:

```
seed@VM: ~/.../Labsetup
[10/12/25] seed@VM:~/.../Labsetup$ openssl x509 -in ca.crt -text -no
out
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            2a:c4:7a:b5:29:cf:6d:9b:d3:fb:5e:3e:6a:71:0b:3e:6f:f8:d
c:d8
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = CA, ST = Some-State, O = SeedLab
        Validity
            Not Before: Oct 12 07:16:36 2025 GMT
            Not After : Oct 10 07:16:36 2035 GMT
        Subject: C = CA, ST = Some-State, O = SeedLab
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public-Key: (4096 bit)
            Modulus:
                00:cf:20:43:69:97:69:46:d1:5f:6f:1f:64:21:9a:
                26:ea:be:6c:c5:16:cc:5d:e2:99:23:bf:c7:3e:01:
                9a:7d:49:8a:e8:49:a4:9a:65:1f:39:2e:17:42:cd:
                70:43:eb:ee:04:f4:54:55:5a:24:02:19:c8:c0:cf:
                b0:4a:98:13:4d:1f:26:ee:66:c5:76:b9:e7:68:3e:
                52:e7:31:a1:9e:33:b6:38:9d:0d:38:10:6b:a3:db:
                29:5e:64:fa:8c:c5:cc:b5:c9:a5:4e:6a:48:71:63:
                a8:f6:7b:d3:0a:4a:a3:bd:45:40:18:02:0e:15:b6:
                c1:18:07:c2:e7:0f:a8:87:f6:50:8e:c3:32:5a:cb:
                7c:ac:ae:00:2c:eb:08:ee:6c:9d:35:86:77:db:f3:
                7a:4a:fd:61:dc:f9:e6:e3:e9:c5:46:5a:58:51:ba:
                a7:13:be:7c:46:85:a1:97:c2:6d:c4:e2:5e:2b:49:
                c1:b1:5d:21:19:31:ac:2f:0a:64:7f:55:4c:a1:9e:
                1b:78:a8:be:c1:30:3b:66:3d:56:e3:25:fc:80:b7:
                8a:40:df:dd:eb:a6:0f:98:9d:71:46:fd:35:31:35:
```

```
seed@VM: ~/.../Labsetup
1f:99:8b:6a:92:46:71:00:c5:51:94:16:bf:1a:0b:
7c:3e:f5:bc:d0:25:82:ad:53:82:0f:eb:07:e3:4f:
65:bb:b3:8d:3b:67:82:6c:f7:20:e1:91:bf:96:d8:
7d:e3:32:41:eb:ed:63:02:98:9b:80:92:68:52:6b:
fc:9c:09:1a:95:8c:8d:96:88:35:e2:45:97:04:19:
a8:00:37:c1:21:06:6f:55:5b:10:51:86:13:9e:bd:
4a:c2:e3:e5:6b:7c:de:fc:69:d4:d9:13:66:e6:77:
77:ea:e2:dc:f3:51:64:1a:b9:03:78:6f:ef:9b:2a:
fc:1d:be:65:75:ec:9d:45:c8:27:d6:23:bc:89:d1:
b5:ad:8a:1e:bd:e9:34:bb:85:b0:02:19:fb:0b:33:
bb:a0:0b:97:4d:f8:4c:16:eb:8d:cc:32:17:2a:5b:
31:2e:26:cb:e6:d0:a1:d5:55:75:92:8d:64:5f:e8:
23:2e:ae:d0:50:5c:63:14:f6:08:3d:83:77:0a:55:
44:a1:a6:28:f3:52:a3:0b:36:e8:b3:35:4a:dd:d1:
0b:5e:f0:0f:5d:a3:cb:2c:2f:1c:ad:e4:18:93:10:
91:f7:4e:3e:db:82:60:89:3c:5c:c6:b4:11:f4:2e:
de:aa:22:a0:1d:92:95:28:54:be:38:f8:13:45:db:
67:3a:45:62:61:4e:c7:5b:b8:66:2b:fd:c0:da:4d:
a7:5d:01:e3:1b:bd:95:32:2b:b3:83:58:61:ed:1d:
59:3d:df
Exponent: 65537 (0x10001)
X509v3 extensions:
    X509v3 Subject Key Identifier:
        DC:75:56:7A:B4:18:FE:1C:1A:27:76:2B:C5:95:09:A4:41:
1A:54:3B
    X509v3 Authority Key Identifier:
        keyid:DC:75:56:7A:B4:18:FE:1C:1A:27:76:2B:C5:95:09:
A4:41:1A:54:3B

    X509v3 Basic Constraints: critical
        CA:TRUE
Signature Algorithm: sha256WithRSAEncryption
4d:13:d5:88:d0:5c:4f:4c:3a:66:a2:13:a4:1e:cb:67:0c:66:
```

```
14:cc:4e:d4:1f:20:a7:2b:8b:83:6e:a2:c0:ef:4a:0f:a7:e5:  
3a:f6:62:f2:98:cd:65:cf:b0:cc:b3:3f:32:35:5f:9b:01:15:  
a9:aa:d4:3b:d5:7e:92:e2:af:60:f9:6d:db:3d:e4:76:3b:76:  
f6:df:ae:e3:75:2a:85:7a:6e:93:9e:69:de:53:ba:de:2a:a7:  
d5:d1:b3:c9:92:84:67:75:4e:ee:82:e0:b5:ed:cd:a2:39:20:  
cc:c0:fe:7c:6f:17:aa:58:a7:f4:7b:47:f3:a3:30:bd:fa:d0:  
c5:1a:d6:b9:8e:21:79:65:7c:3e:88:f9:be:60:38:f7:1b:c3:  
10:15:f4:ad:9d:32:ca:5c:91:3a:e6:a7:56:7f:a0:d5:be:74:  
5f:3e:9f:64:e5:f1:18:4e:7d:4b:87:b6:46:4b:d0:5e:62:da:  
bd:14:d7:86:cc:f1:54:a1:5e:fc:94:79:48:5f:68:75:1e:f4:  
60:a7:3a:23:d9:f9:34:25:4b:71:02:e5:f3:02:b9:be:e1:80:  
b8:db:bb:36:ea:61:47:48:e6:2b:14:0b:c9:6e:ef:f9:b0:06:  
52:29:0d:61:de:04:af:1c:1d:7b:79:2a:bc:5b:99:3c:18:96:  
fe:99:bf:f9:68:42:ff:8f:71:94:67:f9:d8:f7:52:c8:33:e8:  
85:2e:82:ca:f1:ab:60:1e:9f:c5:63:78:fc:c0:33:16:64:74:  
b0:6a:d3:d2:9b:04:a2:1b:21:6f:32:81:9e:d3:66:8a:2d:04:  
2f:81:c3:4b:eb:1e:47:77:1c:b1:4f:a7:33:3d:4c:ef:43:9a:  
5e:b9:14:2c:c1:5d:fc:60:ef:23:f0:9d:64:8f:e5:25:3f:1a:  
ed:fd:89:4b:c8:97:b2:40:39:40:c8:4c:55:59:4e:29:fe:6f:  
2d:f7:76:95:a3:d1:42:b8:9e:24:0f:48:26:45:25:01:0a:2d:  
79:2e:24:d7:94:01:95:7a:ba:80:00:4a:2d:14:f5:92:97:88:  
3d:22:9c:32:75:66:c7:21:c0:39:f6:d5:dc:0c:5a:12:03:0e:  
9f:e1:44:55:38:3d:4b:0a:b3:e2:3b:6c:0f:c6:93:0e:b3:a9:  
3d:ce:a0:31:1f:46:e6:ef:a6:6f:5a:ce:04:a5:4f:0a:6b:d9:  
10:96:9d:a9:35:98:aa:61:7d:0c:30:c2:bc:2c:96:32:ba:8d:  
12:64:4d:29:f0:18:18:3f:a4:8b:6a:04:48:0b:42:e5:f8:54:  
58:a1:10:c0:fe:6c:b1:39:97:be:8c:a3:83:db:32:f9:58:4f:  
c9:8d:3d:9e:fd:12:f8:2a
```

```
[10/12/25]seed@VM:~/.../Labsetup$
```

Running rsa command:

```
seed@VM: ~/.../Labsetup  
[10/10/25]seed@VM:~/.../Labsetup$ openssl rsa -in ca.key -text -noout
```

Public exponent e Screenshot:

```
publicExponent: 65537 (0x10001)
```

### Private exponent d Screenshot:

```
seed@VM: ~/Labsetup
privateExponent:
00:ba:a8:b3:b7:d3:f8:cf:16:5e:af:f1:15:36:e7:
0a:0d:61:00:fa:f8:d9:c1:fe:34:ce:2f:c5:69:ce:
de:1f:ec:87:df:60:86:93:91:6f:fb:02:db:79:9b:
54:82:11:7b:96:41:6e:42:e1:12:34:46:72:51:b8:
99:82:0f:dc:cd:12:b6:ed:ff:f6:12:51:e5:06:5f:
ac:2c:a7:e0:a3:86:ac:c0:06:60:e4:56:7f:fa:d1:
c6:4d:48:29:37:76:06:4f:09:63:0c:f0:e1:4a:36:
6b:0d:7e:77:a8:fd:ac:e4:24:25:67:bd:6a:5e:0d:
2d:a4:ef:51:a9:13:c6:68:89:04:fe:33:51:0c:db:
f7:06:53:64:db:16:11:1d:75:0a:94:33:13:8f:47:
b3:bd:22:ed:ef:cb:d2:55:30:d6:54:d9:94:60:4b:
1a:c8:c3:50:91:e9:59:c0:3f:3a:d7:44:bd:62:bb:
7c:6f:cf:a2:c5:66:ab:6a:55:2e:4e:46:5b:e2:06:
6f:71:64:79:4b:f6:a0:81:39:e4:34:9e:86:18:a5:
cf:55:39:aa:6f:f4:5e:77:c5:e3:50:fb:5d:05:73:
af:7f:9b:51:b6:72:fe:e1:97:f1:35:c8:ed:fd:95:
be:29:ab:21:b6:5f:27:8d:f6:04:ef:3d:7b:05:52:
07:35:ea:b3:fe:69:03:70:48:28:9d:c1:4a:0b:d1:
03:6d:3d:c4:80:92:40:e2:33:c7:29:79:56:73:85:
27:0d:1d:93:d1:d1:54:75:7b:fc:1b:45:8e:fe:8b:
39:fa:04:a3:35:f0:a1:51:3e:d3:6b:22:bb:ca:38:
10:a2:f7:23:bc:e4:44:99:0d:18:81:1f:cf:3c:0e:
4d:bf:10:96:54:d1:e3:b4:69:78:c5:bc:d6:8e:d6:
d4:1a:98:ac:ed:48:15:e9:15:59:bc:75:4e:b3:ad:
43:cc:11:81:7f:13:45:9b:17:dd:94:55:52:19:90:
60:1c:f7:c0:17:8b:22:ca:b5:00:31:4b:37:e7:74:
14:cc:7b:3d:f3:92:6f:15:70:08:c1:f7:87:b5:82:
6e:0e:8e:8b:dc:4a:37:44:6b:15:da:cb:03:5c:28:
b1:53:74:50:8b:c9:72:c5:25:c9:2c:6b:60:56:f5:
3c:e6:fb:c6:a3:c2:9f:91:c6:f7:b5:38:25:bf:83:
cc:25:16:7e:5c:ae:1c:78:4d:81:95:8b:29:1d:87:
90:c3:84:43:15:1b:78:4d:3f:15:f0:aa:62:07:20:
```

### Modulus n ss:

```
seed@VM: ~/Labsetup
$ ./privateKey (1024 bit, 2 primes)
modulus:
00:cf:20:43:69:97:69:46:d1:5f:6f:1f:64:21:9a:
26:ea:be:6c:c5:16:cc:5d:e2:99:23:bf:c7:3e:01:
9a:7d:49:8a:e8:49:a4:9a:65:1f:39:2e:17:42:cd:
70:43:eb:ee:04:f4:54:55:5a:24:02:19:c8:c0:cf:
b0:4a:98:13:4d:1f:26:ee:66:c5:76:b9:e7:68:3e:
52:e7:31:a1:9e:33:b6:38:9d:0d:38:10:6b:a3:db:
29:5e:64:fa:8c:c5:cc:b5:c9:a5:4e:6a:48:71:63:
a8:f6:7b:d3:0a:4a:a3:bd:45:40:18:02:0e:15:b6:
c1:18:07:c2:e7:0f:a8:87:f6:50:8e:c3:32:5a:cb:
7c:ac:ae:00:2c:eb:08:ee:6c:9d:35:86:77:db:f3:
7a:4a:fd:61:dc:f9:e6:e3:e9:c5:46:5a:58:51:ba:
a7:13:be:7c:46:85:a1:97:c2:6d:c4:e2:5e:2b:49:
c1:b1:5d:21:19:31:ac:2f:0a:64:7f:55:4c:a1:9e:
1b:78:a8:be:c1:30:3b:66:3d:56:e3:25:fc:80:b7:
8a:40:df:dd:eb:a6:0f:98:9d:71:46:fd:35:31:35:
1f:99:8b:6a:92:46:71:00:c5:51:94:16:bf:1a:0b:
7c:3e:f5:bc:d0:25:82:ad:53:82:0f:eb:07:e3:4f:
65:bb:b3:8d:3b:67:82:6c:f7:20:e1:91:bf:96:d8:
7d:e3:32:41:eb:ed:63:02:98:9b:80:92:68:52:6b:
fc:9c:09:1a:95:8c:8d:96:88:35:e2:45:97:04:19:
a8:00:37:c1:21:06:6f:55:5b:10:51:86:13:9e:bd:
4a:c2:e3:e5:6b:7c:de:fc:69:d4:d9:13:66:e0:77:
77:ea:e2:dc:f3:51:64:1a:b9:03:78:6f:ef:9b:2a:
fc:1d:be:65:75:ec:9d:45:c8:27:d6:23:bc:89:d1:
b5:ad:8a:1e:bd:e9:34:bb:85:b0:02:19:fb:0b:33:
bb:a0:0b:97:4d:f8:4c:16:eb:8d:cc:32:17:2a:5b:
31:2e:26:cb:e6:d0:a1:d5:55:75:92:8d:64:5f:e8:
23:2e:ae:d0:50:5c:63:14:f6:08:3d:83:77:0a:55:
44:a1:a6:28:f3:52:a3:0b:36:e8:b3:35:4a:dd:d1:
0b:5e:f0:0f:5d:a3:cb:2c:2f:1c:ad:e4:18:93:10:
91:f7:4e:3e:db:82:60:89:3c:5c:c6:b4:11:f4:2e:
de:aa:22:a0:1d:92:95:28:54:be:38:f8:13:45:db:
```

**Secret key p ss:**

```
prime1:  
 00:ea:ae:19:13:cd:40:ad:2c:e5:0c:77:86:ee:2d:  
 26:f6:b3:c1:c4:06:34:68:e0:23:f2:60:2b:f9:61:  
 40:28:8c:50:97:c4:8b:4b:7c:78:ac:7a:00:5d:c0:  
 5d:7a:51:f9:55:88:6d:1d:7f:fa:0f:bc:a3:02:77:  
 4d:e3:4b:96:14:ad:59:ad:e1:1f:db:56:4e:05:74:  
 49:e9:40:34:57:9d:9e:db:b6:ad:3a:6f:ac:dc:b3:  
 64:78:da:77:81:a5:a0:9f:5f:fe:3b:52:7f:aa:1b:  
 5d:4d:bf:92:ac:64:2e:ab:5f:69:2f:1d:5a:be:2a:  
 77:f6:46:27:41:0e:17:1b:53:4c:d9:a9:d1:c7:5d:  
 f9:8c:10:05:99:1d:37:64:4a:72:7d:d1:5a:b6:78:  
 28:a8:2a:46:35:1c:b4:cb:d1:03:2b:17:6a:72:6a:  
 95:77:a0:08:15:b6:50:b8:38:11:62:30:53:42:bf:  
 32:3f:e5:7a:2b:b4:5c:10:77:c4:07:e8:4e:a2:22:  
 a5:e5:d9:4b:28:05:94:1e:53:2e:8b:11:ad:3c:03:  
 77:80:28:09:ce:e8:f0:e8:b4:02:2b:79:fb:c0:e9:  
 41:f5:d3:be:de:5c:d5:be:05:4d:12:15:bf:27:66:  
 63:90:2e:88:49:0e:cd:7f:f6:0c:4e:c0:dd:e3:21:  
 cb:ed
```

**Secret key q ss:**

```
prime2:  
 00:e1:f1:58:e0:6c:72:8c:6e:88:24:dd:9e:8e:08:  
 64:2f:60:61:d9:14:cc:77:95:55:8a:56:85:de:7e:  
 95:d6:f8:f1:ae:5c:8a:03:de:d1:70:f2:dd:8e:04:  
 7d:4a:a3:65:73:1f:07:e6:28:fa:19:cc:69:f1:9c:  
 42:e8:d3:d4:41:06:74:38:d3:9d:ed:78:88:78:6b:  
 e2:8d:fb:06:23:79:ac:d7:f3:4d:81:7b:75:85:f2:  
 af:b4:d4:39:b4:81:74:9f:ef:d6:b2:c7:45:41:7a:  
 4b:02:ed:b1:86:97:f5:16:3b:68:d0:46:b9:33:2b:  
 0e:4f:98:17:be:51:a8:2b:b0:f6:61:ce:5e:f4:1c:  
 4c:d2:34:79:9d:c4:1e:33:0f:8a:5b:e1:59:51:42:  
 81:79:6b:88:84:4d:cc:42:f2:7c:d1:d7:cb:24:ec:  
 be:f9:0d:2a:02:a0:aa:01:08:08:39:8f:30:9c:c8:  
 b8:9e:c8:ae:41:f3:23:66:86:84:e9:02:98:9c:00:  
 26:ca:e4:5a:04:96:7e:91:9e:11:74:20:cd:f9:5e:  
 a6:5d:a0:3a:fd:3b:ae:59:b4:43:10:80:05:7d:38:  
 1a:9c:9a:2e:15:05:d9:5d:e7:31:83:a6:2e:bb:20:  
 3c:d1:f7:a5:9d:1c:6e:9c:b3:f5:1a:5f:46:16:62:  
 ef:7b
```

## Task 2:

Generating CSR Screenshot:

```
[10/12/25] seed@VM:~/.../Labsetup$ openssl req -newkey rsa:2048 -sha256 -keyout server.key -out server.csr -subj "/CN=www.aneesh2025.com/O=Aneesh" -passout pass:dees
Generating a RSA private key
.....................................................................+++++
....+++
writing new private key to 'server.key'
```

**Description:** The above command creates an RSA key which is 2048 bits and uses it to make a certificate signing request called CSR. It saves the private key as server.key and the CSR as server.csr using the name and the org I gave.

Looking at the decoded content CSR:

```
[10/12/25] seed@VM:~/.../Labsetup$ openssl req -in server.csr -text -noout
Certificate Request:
Data:
Version: 1 (0x0)
Subject: CN = www.aneesh2025.com, O = Aneesh
Subject Public Key Info:
    Public Key Algorithm: rsaEncryption
        RSA Public-Key: (2048 bit)
        Modulus:
            00:b6:20:c5:40:f9:91:3b:68:ae:b8:24:67:86:91:
            ba:27:08:9b:8a:dd:01:46:93:ec:a3:fe:78:31:73:
            74:b0:e3:60:b3:f7:0f:c1:69:e6:03:b5:f5:44:1a:
            0a:70:a4:dc:18:f6:61:0a:96:e0:85:7c:2d:cc:8d:
            3b:bf:08:8a:7c:9c:f9:4f:7a:e9:bb:1a:d5:2e:e6:
            d8:06:7f:f1:85:e8:fb:c9:7a:ae:f7:27:7d:99:23:
            57:4d:c0:76:6c:4a:2d:46:2a:85:9f:fd:8c:ba:a9:
            ca:28:6a:07:22:fe:8c:51:3e:30:c1:5d:88:87:4d:
            2e:d7:23:77:fe:86:47:87:03:9d:40:fa:c2:1f:05:
            e6:02:65:4f:e8:56:9a:69:52:dc:81:99:03:98:5b:
            33:37:4f:d1:d1:5f:c7:91:99:2b:72:1a:5d:0f:5b:
            93:d8:70:1f:f7:28:46:cb:b2:6f:ee:4b:7d:e4:0f:
            d8:ac:16:cc:50:f4:5c:bd:45:1d:12:5d:3c:8c:46:
            30:5b:e8:12:e0:e1:f8:4c:ad:af:73:14:94:ad:4a:
            85:e7:61:bf:34:45:00:8d:af:22:d0:ad:da:cb:5d:
            02:58:dd:fe:d2:1f:e8:1b:7b:58:8c:bc:8d:f5:1a:
            8d:dc:98:f7:c9:59:92:c7:cf:ec:7d:3a:8c:b3:2f:
            db:17
        Exponent: 65537 (0x10001)
Attributes:
    a0:00
Signature Algorithm: sha256WithRSAEncryption
    7f:71:45:1d:96:d4:93:0f:7f:3e:b0:a7:94:a4:d4:ff:cc:06:
```

```
.....  
Signature Algorithm: sha256WithRSAEncryption  
7f:71:45:1d:96:d4:93:0f:7f:3e:b0:a7:94:a4:d4:ff:cc:06:  
d2:45:b6:de:00:66:fc:a6:ef:27:4d:e7:49:95:fe:3a:da:cc:  
19:76:ef:5a:be:3f:04:41:7c:d2:93:08:d7:9d:08:98:64:9c:  
a0:4a:45:a5:63:f1:00:bf:80:1d:54:83:65:45:e1:c1:03:3f:  
a8:e9:29:94:6c:3d:fd:32:a1:97:45:b6:0c:0e:c8:97:39:e8:  
c5:06:bb:76:4a:01:19:0b:84:b2:32:2d:65:b9:25:a8:ee:8f:  
88:fb:0c:13:d6:67:3a:b1:f8:7a:52:75:8e:df:1e:0c:5d:42:  
65:63:37:c8:40:58:5c:11:6b:cf:9f:e4:14:3a:b2:24:41:19:  
39:9e:a5:2b:33:ee:34:ea:5c:29:e5:50:de:71:b7:07:2d:bd:  
41:89:ec:38:a3:21:2e:e4:1a:dd:10:b6:80:af:de:04:cc:1f:  
d4:50:a5:d4:a5:de:11:dc:bb:5d:f3:3b:e9:2f:74:e4:e5:22:  
8c:7c:48:ae:72:aa:ae:1a:4f:49:f5:46:a2:46:fa:28:43:dc:  
09:05:0c:b9:81:7f:65:3c:c4:72:7d:d0:4b:bd:3d:42:21:8f:  
fe:7d:00:88:08:2f:10:1e:55:c0:e0:d6:70:00:1d:18:77:48:  
61:3d:d8:22
```

```
[10/12/25]seed@VM:~/.../Labsetup$ █
```

**Running the RSA command:**

```
[10/12/25]seed@VM:~/.../Labsetup$ openssl rsa -in server.key -text  
-noout  
Enter pass phrase for server.key:
```

**Description:** Loading the private key file called server.key which shows the key details in a readable format.

**Modulus Screenshot:**

modulus:

```
00:b6:20:c5:40:f9:91:3b:68:ae:b8:24:67:86:91:  
ba:27:08:9b:8a:dd:01:46:93:ec:a3:fe:78:31:73:  
74:b0:e3:60:b3:f7:0f:c1:69:e6:03:b5:f5:44:1a:  
0a:70:a4:dc:18:f6:61:0a:96:e0:85:7c:2d:cc:8d:  
3b:bf:08:8a:7c:9c:f9:4f:7a:e9:bb:1a:d5:2e:e6:  
d8:06:7f:f1:85:e8:fb:c9:7a:ae:f7:27:7d:99:23:  
57:4d:c0:76:6c:4a:2d:46:2a:85:9f:fd:8c:ba:a9:  
ca:28:6a:07:22:fe:8c:51:3e:30:c1:5d:88:87:4d:  
2e:d7:23:77:fe:86:47:87:03:9d:40:fa:c2:1f:05:  
e6:02:65:4f:e8:56:9a:69:52:dc:81:99:03:98:5b:  
33:37:4f:d1:d1:5f:c7:91:99:2b:72:1a:5d:0f:5b:  
93:d8:70:1f:f7:28:46:cb:b2:6f:ee:4b:7d:e4:0f:  
d8:ac:16:cc:50:f4:5c:bd:45:1d:12:5d:3c:8c:46:  
30:5b:e8:12:e0:e1:f8:4c:ad:af:73:14:94:ad:4a:  
85:e7:61:bf:34:45:00:8d:af:22:d0:ad:da:cb:5d:  
02:58:dd:fe:d2:1f:e8:1b:7b:58:8c:bc:8d:f5:1a:  
8d:dc:98:f7:c9:59:92:c7:cf:ec:7d:3a:8c:b3:2f:  
db:17
```

**Public exponent ss:**

```
publicExponent: 65537 (0x10001)
```

**Private Exponent ss:**

```
privateExponent:  
    00:9b:2e:10:e3:eb:c0:f2:fd:9e:e1:07:23:ae:65:  
    3c:57:45:de:41:d3:4c:e2:f5:e7:0b:54:26:d4:fb:  
    aa:d1:8c:eb:38:6b:a9:e1:02:7a:19:6f:50:2b:56:  
    84:ce:b5:a9:53:1b:9a:53:8b:7a:82:41:ec:43:fd:  
    92:8a:92:9c:6c:d0:d3:1c:05:7e:61:0d:db:d6:ba:  
    9a:15:ef:ad:ba:8b:68:fd:a2:aa:b1:ab:d4:38:47:  
    04:7c:1b:20:cb:e3:e7:ef:81:0b:99:e6:4f:a2:3c:  
    49:23:66:33:ee:7d:34:d1:84:4b:c6:76:0b:7e:3d:  
    25:4b:e9:cc:b3:f1:75:c0:72:c3:18:ed:4a:c7:86:  
    72:9f:f3:40:07:56:96:78:42:25:0b:28:ad:c0:14:  
    81:c9:d3:33:f0:9b:a8:b1:d4:61:1e:44:34:d5:ff:  
    7a:af:9f:47:68:30:98:4d:7e:17:e3:c2:be:2a:7e:  
    a6:2d:5e:e1:25:38:00:8f:1e:8d:29:72:22:66:1c:  
    f4:c3:ab:c1:1f:d2:b9:aa:7f:70:78:81:5f:00:8a:  
    17:65:6e:89:dd:80:73:20:13:d6:92:e1:e1:d0:15:  
    e1:37:a3:52:f5:96:52:54:ae:cb:09:80:ef:c0:e4:  
    11:28:bc:56:52:01:79:0b:d8:ab:af:ee:bf:20:7c:  
    7d:01
```

**Secret key p Screenshot:**

```
prime1:  
    00:e7:28:14:7b:50:ac:6b:d1:aa:2c:39:40:a4:5c:  
    9a:09:38:89:0d:47:bb:53:7a:9e:4e:b4:81:75:9f:  
    5f:32:56:5c:3b:b9:6d:a5:83:79:a1:e4:87:d1:14:  
    fc:7f:14:f2:84:b1:3e:3a:7e:4f:d6:64:cc:0e:ca:  
    f9:53:ef:d3:4b:48:18:88:c1:e7:97:05:06:45:e4:  
    d6:6b:16:4f:98:18:fa:2a:2f:d0:55:34:c5:c8:dc:  
    31:b4:1d:e9:ac:be:da:48:50:f2:97:c2:61:e0:7b:  
    b5:3a:25:c1:82:29:30:3b:3a:bd:2a:5f:8b:79:f3:  
    62:2b:20:e3:5d:56:12:d4:25
```

**Secret key q ss:**

```
prime2:  
    00:c9:b3:be:a6:37:2a:1e:52:e2:e5:81:80:eb:45:  
    ad:65:0c:8e:99:d9:c8:80:ae:12:c9:a9:0b:f9:dc:  
    87:0d:8f:c3:07:d5:e8:dd:d4:ea:cc:15:e4:16:a8:  
    60:ae:3c:8b:6e:5a:e2:44:0c:03:0b:07:79:d8:8c:  
    c7:c4:b2:75:a4:93:41:56:2d:70:1a:af:e7:05:e4:  
    b7:e5:30:16:e3:0b:3f:f6:dc:12:01:0c:81:a2:a3:  
    85:65:b7:ef:82:77:ef:cb:b8:62:7a:75:83:4f:c0:  
    a1:71:9e:86:e3:01:14:0d:09:13:b3:87:09:76:38:  
    40:e7:06:40:15:53:be:8f:8b
```

Adding alternative names to CA certificate:

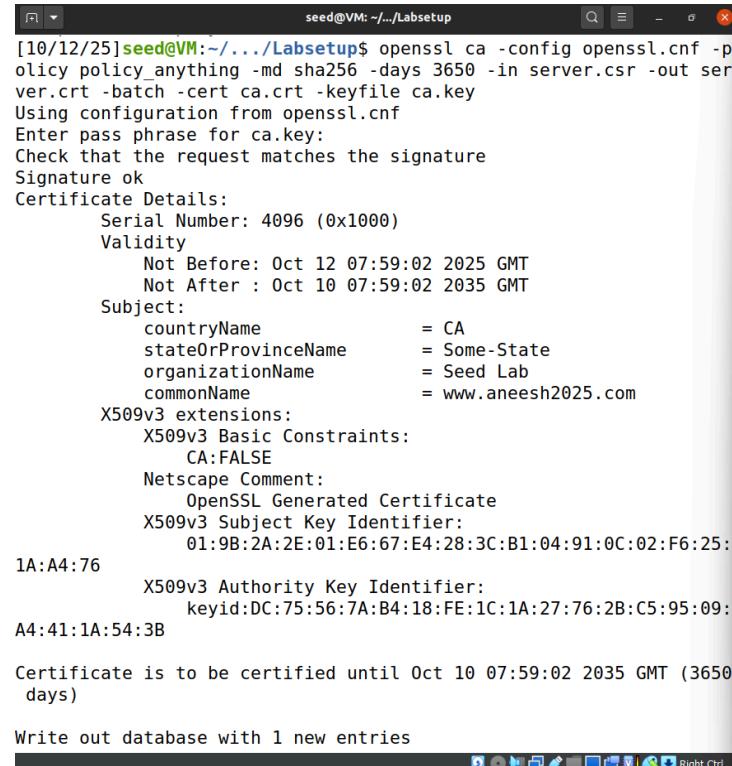
```
[10/12/25]seed@VM:~/.../Labsetup$ openssl req -new -key server.key  
-out server.csr -addext "subjectAltName = DNS:www.aneesh2025.com,  
DNS:www.aneesh2025A.com, DNS:www.aneesh2025B.com"  
Enter pass phrase for server.key:  
You are about to be asked to enter information that will be incorpo-  
rated  
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]:CA  
State or Province Name (full name) [Some-State]:  
Locality Name (eg, city) []:  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Seed Lab  
Organizational Unit Name (eg, section) []:  
Common Name (e.g. server FQDN or YOUR name) []:  
Email Address []:  
  
Please enter the following 'extra' attributes  
to be sent with your certificate request  
A challenge password []:  
An optional company name []:  
[10/12/25]seed@VM:~/.../Labsetup$ █
```

```
[10/12/25] seed@VM:~/.../Labsetup$ openssl req -newkey rsa:2048 -sha  
256 -subj "/CN=www.aneesh2025.com/O=Aneesh" -passout pass:dees -add  
ext "subjectAltName = DNS:www.aneesh2025.com, DNS:www.aneesh2025A.c  
om, DNS:www.aneesh2025B.com"  
Generating a RSA private key  
-----+  
-----++  
writing new private key to 'privkey.pem'  
-----  
-----BEGIN CERTIFICATE REQUEST-----  
MIICzTCCAbUCAQAwLjEbMBkGA1UEAwSd3d3LmFuZWVzaDIwMjUuY29tMQ8wDQYD  
VQQKDAZBbmVlc2gwggiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQC6/Z7m  
kqtYoHMhataoU6j6tyskbMehAfAZB+FV7SuqrSKwlSaYl18ekYDXZC2RDGbLe+Ha  
pq19ffY2hvqEkGLeB93Rh4J82LpipJHQYmg0iLUsL5hFXBazb+98k8r3waIBq7po  
NIxjyfHh5NDZPiFq5Rec0omTGb86gs0W7mPHK04HMoReaNIt03+avLSJ7Ps7uhfX  
Oef7AC/zQVsZCqehN/SA1sIjVjMBeeR0CHafx4AclIJx85FfrY1TsUVE+BNY7Heb  
KiYJNhfeW4ssbC0sVKE10JGBG9h1jD8o+mfuaCxUUkXX65aUBn2w6Q0UbHb8PV0B  
0AdApv3BvIXdaUc5AgMBAAGgWjBYBgkqhkiG9w0BCQ4xSzBJMEcGA1UdEQRAMD6C  
End3dy5hbmVlc2gyMDI1LmNvbYITd3d3LmFuZWVzaDIwMjVBLmNvbYITd3d3LmFu  
ZWVzaDIwMjVCLmNvbTANBgkqhkiG9w0BAQsFAA0CAQEAn1z3WqdrfTDaLNsRbqX  
DWK56AVDxjAe/n01yur8p46YSJHikzX8UsNPxcimSc2ahWnobUKAmvwN4gb8X2GN  
JXrHqVwj9UHuqpst1wzj1WGWBgC7eNxp2YW2JQm3Jp0eqqAedBLBu7g3opeW2Nr  
RjIity7fJaV7wo04AoprMfS3q7FqISQcJ93YvILAjLwKhBEch8fVIKlkHVvRuQhM  
RpI3q9T6labnKbybayx21mX1LpV5d8kAenimdruHGpfNS6kcyi1LX8v1/x/ggWK1  
dFP1Xo+Qa/URgu9hMeaCZcHCy8XLQB6XRUpKYFwsPbBY5QttdtwMrSAh4g8sKjCsR  
RQ==  
-----END CERTIFICATE REQUEST-----  
[10/12/25] seed@VM:~/.../Labsetup$
```

**Description and Explanation:** Other than omitting the `-x509` option, the CSR generation procedure is pretty much the same as that of generating the CA certificate in Task 1. When the `-x509` is omitted, OpenSSL will generate a signing request instead of a self-signed certificate. Then, the program adds three hostnames to the `subjectAltName` extension, which is necessary for the current browsers to prevent “hostname mismatch” issues. The private key analysis reveals that the key is a 2048-bit RSA key with the usual public exponent 65537, and the decoded CSR validates the right CN and SAN values.

### Task 3:

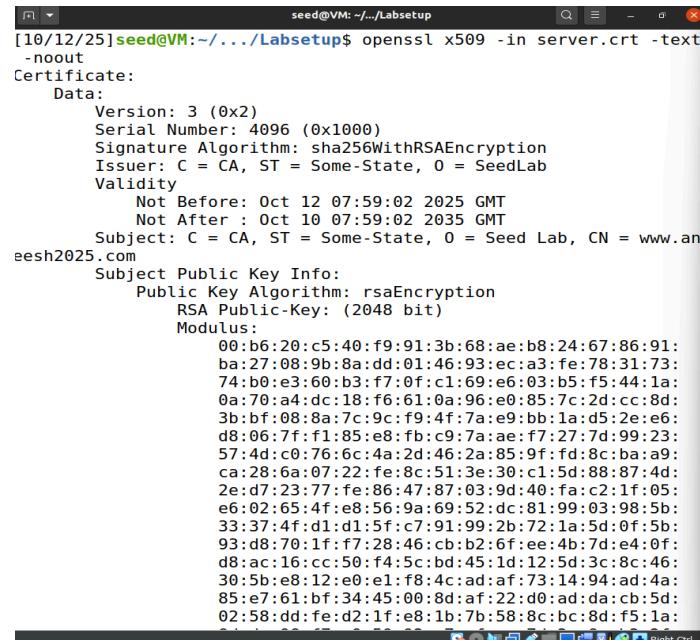
#### Generating certificate for my server:



```
[10/12/25]seed@VM:~/.../Labsetup$ 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: Oct 12 07:59:02 2025 GMT
        Not After : Oct 10 07:59:02 2035 GMT
    Subject:
        countryName      = CA
        stateOrProvinceName = Some-State
        organizationName   = Seed Lab
        commonName         = www.anneesh2025.com
X509v3 extensions:
    X509v3 Basic Constraints:
        CA:FALSE
    Netscape Comment:
        OpenSSL Generated Certificate
    X509v3 Subject Key Identifier:
        01:9B:2A:2E:01:E6:67:E4:28:3C:B1:04:91:0C:02:F6:25:
1A:A4:76
    X509v3 Authority Key Identifier:
        keyid:DC:75:56:7A:B4:18:FE:1C:1A:27:76:2B:C5:95:09:
A4:41:1A:54:3B
Certificate is to be certified until Oct 10 07:59:02 2035 GMT (3650
days)
Write out database with 1 new entries
```

**Description and Explanation:** This command basically tells OpenSSL to act like a certificate authority and sign the server's request file. It uses the CA's private key which we created before and certificate to create a new server certificate. The config file defines the rules, and the -days3650 part makes it valid for 10 years.

#### Printing the decoded content of the certificate:



```
[10/12/25]seed@VM:~/.../Labsetup$ openssl x509 -in server.crt -text
-nouout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 4096 (0x1000)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = CA, ST = Some-State, O = SeedLab
        Validity
            Not Before: Oct 12 07:59:02 2025 GMT
            Not After : Oct 10 07:59:02 2035 GMT
        Subject: C = CA, ST = Some-State, O = Seed Lab, CN = www.an
        eesh2025.com
    Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
        RSA Public-Key: (2048 bit)
        Modulus:
            00:b6:20:c5:40:f9:91:3b:68:ae:b8:24:67:86:91:
            ba:27:08:9b:8a:dd:01:46:93:ec:a3:fe:78:31:73:
            74:b0:e3:60:b3:f7:0f:c1:69:e6:03:b5:f5:44:1a:
            0a:70:a4:dc:18:f6:61:0a:96:e0:85:7c:2d:cc:8d:
            3b:bf:08:8a:7c:9c:f9:4f:7a:e9:bb:1a:d5:2e:e6:
            d8:06:7f:f1:85:e8:fb:c9:7a:ae:f7:27:7d:99:23:
            57:4d:c0:76:6c:4a:2d:46:2a:85:9f:fd:8c:ba:a9:
            ca:28:6a:07:22:fe:8c:51:3e:30:c1:5d:88:87:4d:
            2e:d7:23:77:fe:86:47:87:03:9d:40:fa:c2:1f:05:
            e6:02:65:4f:e8:56:9a:69:52:dc:81:99:03:98:5b:
            33:37:4f:d1:d1:5f:c7:91:99:2b:72:1a:5d:0f:5b:
            93:d8:70:1f:f7:28:46:cb:b2:6f:ee:4b:7d:e4:0f:
            d8:ac:16:cc:50:f4:5c:bd:45:1d:12:5d:3c:8c:46:
            30:5b:e8:12:e0:e1:f8:4c:ad:af:73:14:94:ad:4a:
            85:e7:61:bf:34:45:00:8d:af:22:d0:ad:da:cb:5d:
            02:58:dd:fe:d2:1f:e8:1b:7b:58:8c:bc:8d:f5:1a:
            01:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:
```

1A:A4:76

X509v3 Authority Key Identifier:  
keyid:DC:75:56:7A:B4:18:FE:1C:1A:27:76:2B:C5:95:09:

A4:41:1A:54:3B

Signature Algorithm: sha256WithRSAEncryption

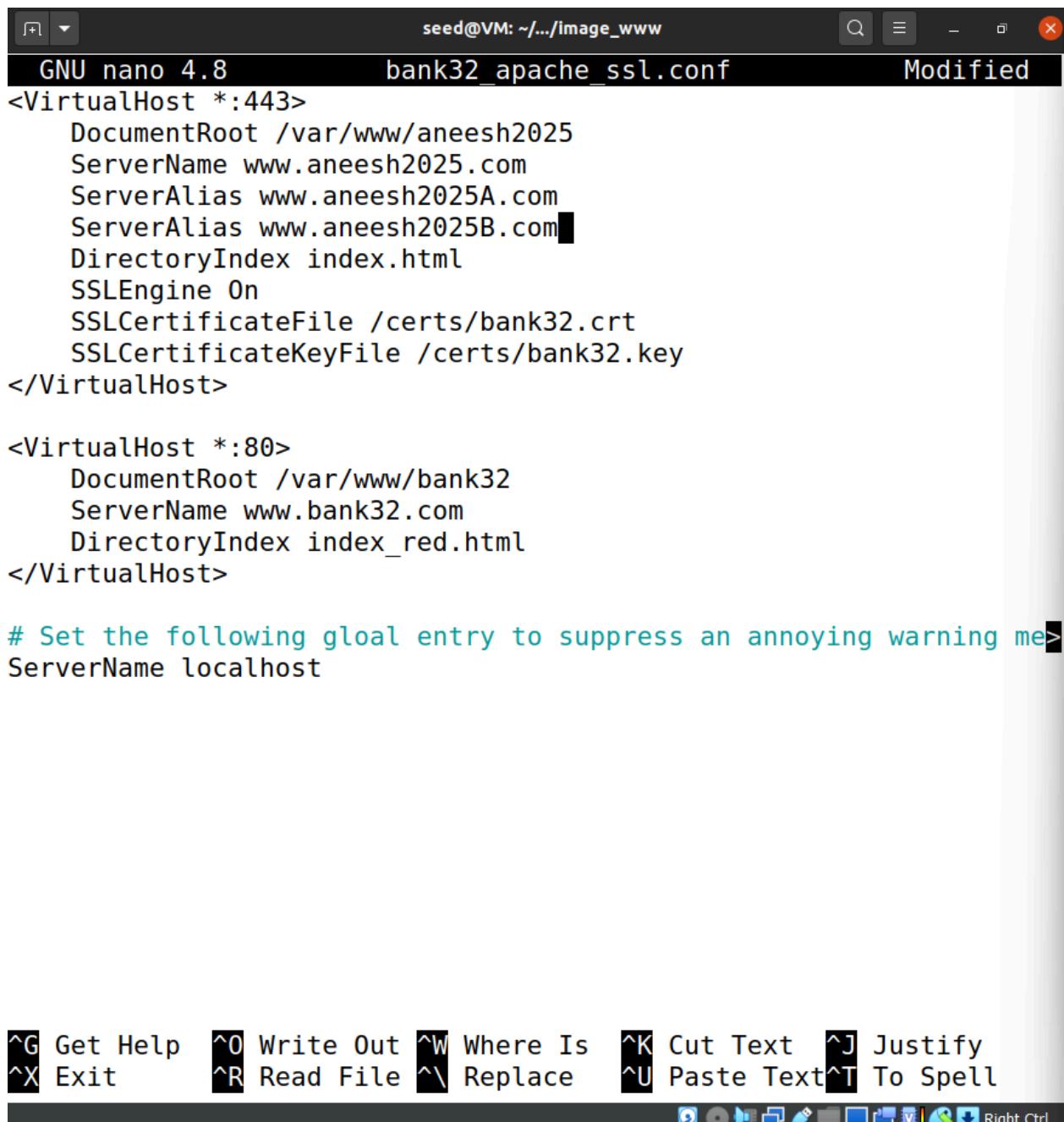
bb:bf:5f:67:ed:62:2d:c8:7b:02:24:76:5e:cc:84:3b:c3:7d:  
6c:47:27:31:be:95:74:ea:5d:71:fd:3a:48:62:f6:f0:2b:0d:  
98:a1:ba:a2:86:77:dd:06:1a:4c:11:cb:b7:20:76:58:33:33:  
37:05:20:9d:d2:6e:8f:15:59:c2:8a:22:6e:af:e4:07:22:16:  
bf:af:19:5a:f3:a3:80:5a:46:f8:40:71:b2:6e:13:93:1f:09:  
60:4e:a1:bc:58:e9:cc:7e:e6:21:9f:7a:06:80:81:9b:8a:9e:  
65:1b:bc:8e:fc:dc:0d:d2:e9:0d:7c:e7:35:6c:d0:92:14:3c:  
f8:07:24:a4:b8:2c:28:a3:2c:8b:a9:bd:b8:96:39:af:7b:6a:  
d8:c8:a8:96:a3:d1:cb:56:1d:e6:d6:db:f8:55:d0:9b:9a:6f:  
97:2d:4c:09:3a:8c:93:83:1e:3f:ec:86:3d:5a:1a:b3:42:1c:  
5e:cc:13:e7:34:47:e7:1f:ab:59:24:ff:91:5a:9c:f3:a9:0c:  
75:98:df:d9:b6:99:d2:36:36:00:89:de:69:c4:55:e6:19:69:  
3b:47:d4:92:83:fd:eb:59:ab:76:85:0b:9d:c3:71:e0:f1:24:  
a8:70:b7:1e:70:37:e7:2b:ea:e9:c5:95:db:8e:d6:a2:9d:86:  
08:0d:1f:8d:b8:a8:75:aa:6c:f1:a3:3f:f4:e9:57:74:17:1e:  
7f:1a:1e:e8:29:0f:bc:ba:21:b7:aa:2f:4c:10:0c:24:dc:0a:  
89:6c:1f:bd:36:06:aa:fc:31:c9:3f:78:e6:e0:3d:0b:61:db:  
61:46:56:17:b0:74:ae:eb:5b:5c:53:b7:1c:61:72:4f:1a:28:  
f9:2e:16:4f:88:b0:23:c1:93:fa:5e:d6:95:11:60:75:46:65:  
64:67:b1:4d:13:e9:db:a9:71:ec:39:ec:23:40:58:ba:72:4b:  
3c:9d:c6:62:0c:88:cf:8f:d7:79:95:08:4a:01:43:b8:f7:9b:  
69:2a:5e:31:21:71:8c:16:e4:ee:24:56:c2:64:3f:e7:d3:03:  
8f:42:e4:5a:8a:f1:05:9e:a4:2f:19:19:94:e5:3a:93:fa:1f:  
33:c8:7d:6b:b4:1e:15:f0:7d:c6:12:b0:6b:33:72:76:97:d1:  
84:bd:9a:fc:74:1c:61:02:eb:20:90:a3:21:69:a7:ef:0f:0c:  
8f:25:7f:62:f3:eb:1d:ab:82:cf:7d:a6:00:9f:2a:5d:50:44:  
82:a3:d8:31:06:aa:5e:fb:cd:2b:3b:1e:a6:c7:36:f6:92:a7:



**Description:** We can see the server's certificate which we generated and it shows all the details in proper readable format.

#### Task 4:

Changing the files in the container to correct files Screenshot:



```
seed@VM: ~/.../image_www
GNU nano 4.8          bank32_apache_ssl.conf      Modified
<VirtualHost *:443>
    DocumentRoot /var/www/aneesh2025
    ServerName www.aneesh2025.com
    ServerAlias www.aneesh2025A.com
    ServerAlias www.aneesh2025B.com
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /certs/bank32.crt
    SSLCertificateKeyFile /certs/bank32.key
</VirtualHost>

<VirtualHost *:80>
    DocumentRoot /var/www/bank32
    ServerName www.bank32.com
    DirectoryIndex index_red.html
</VirtualHost>

# Set the following gloal entry to suppress an annoying warning me>
ServerName localhost

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File  ^\ Replace   ^U Paste Text^T To Spell

```

**Explanation:** Our files are called server.crt and not bank32.crt (original files) so changed that for both server.crt and server.key and also changed the DocumentRoot to my folder rather than the original folder.

**Modifying the DockerFile with correct information:**



```
seed@VM: ~/.../image_www
GNU nano 4.8          Dockerfile
FROM handsonsecurity/seed-server:apache-php

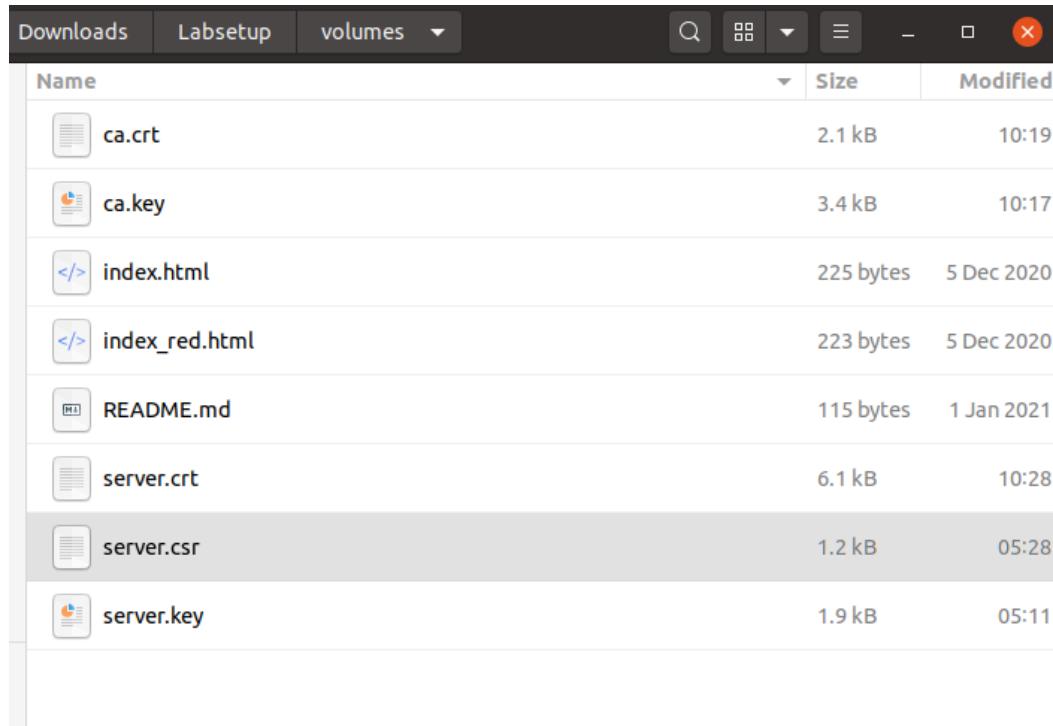
ARG WWWDIR=/var/www/aneesh2025

COPY ./index.html ./index_red.html $WWWDIR/
COPY ./bank32_apache_ssl.conf /etc/apache2/sites-available
COPY ./certs/bank32.crt ./certs/bank32.key /certs/

RUN chmod 400 /certs/bank32.key \
&& chmod 644 $WWWDIR/index.html \
&& chmod 644 $WWWDIR/index_red.html \
&& a2ensite bank32_apache_ssl

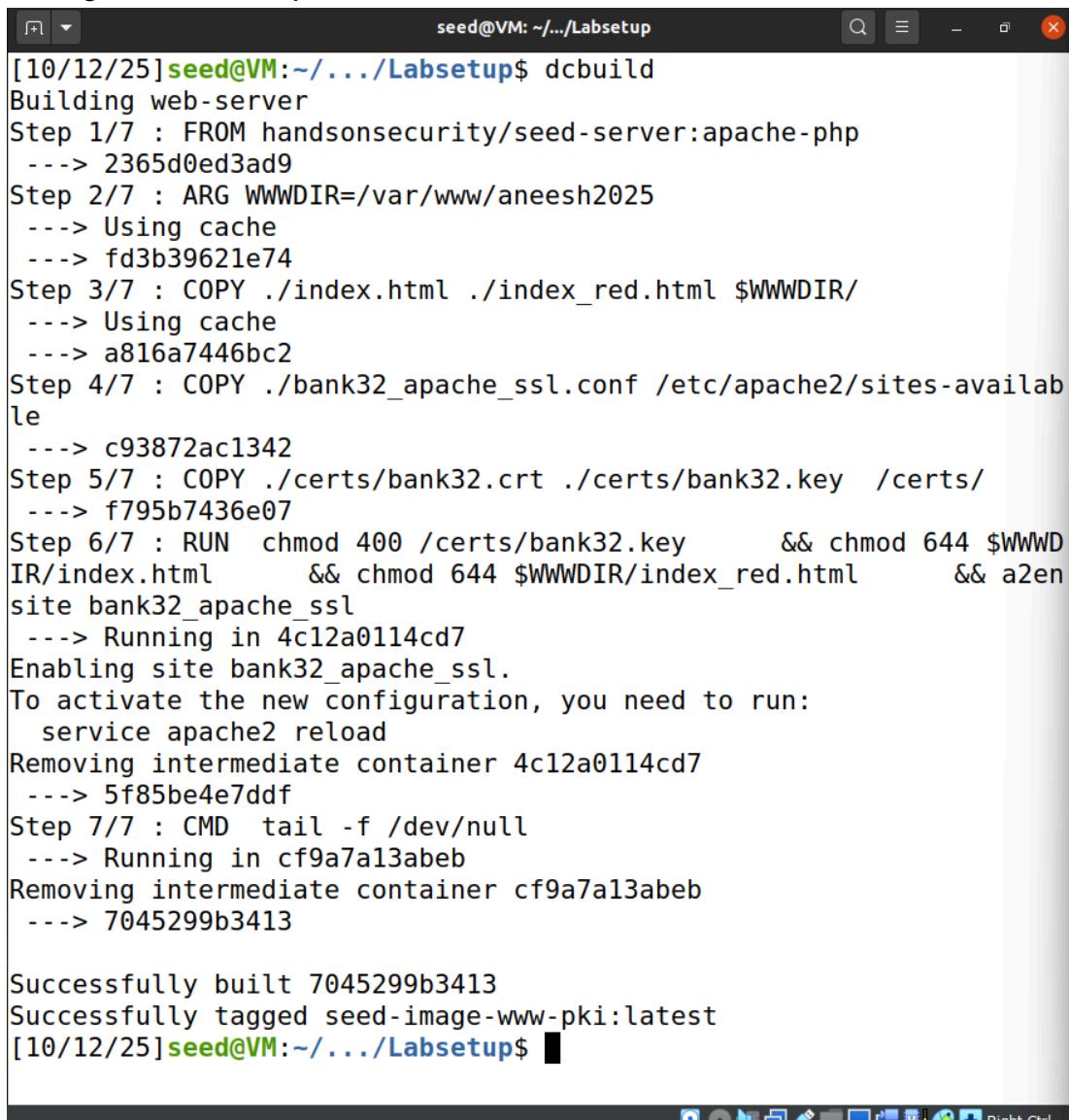
CMD tail -f /dev/null
```

**Moving all the essential files in the volumes folder:**



Name	Size	Modified
ca.crt	2.1 kB	10:19
ca.key	3.4 kB	10:17
index.html	225 bytes	5 Dec 2020
index_red.html	223 bytes	5 Dec 2020
README.md	115 bytes	1 Jan 2021
server.crt	6.1 kB	10:28
server.csr	1.2 kB	05:28
server.key	1.9 kB	05:11

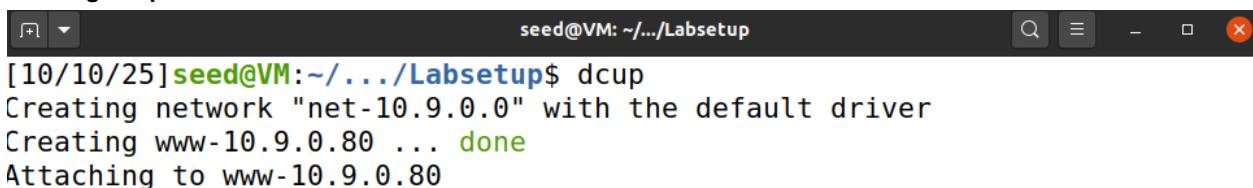
Running “dcbuild” to compose the docker files:



```
[10/12/25] seed@VM:~/.../Labsetup$ dcbuild
Building web-server
Step 1/7 : FROM handsonsecurity/seed-server:apache-php
--> 2365d0ed3ad9
Step 2/7 : ARG WWWDIR=/var/www/aneesh2025
--> Using cache
--> fd3b39621e74
Step 3/7 : COPY ./index.html ./index_red.html $WWWDIR/
--> Using cache
--> a816a7446bc2
Step 4/7 : COPY ./bank32_apache_ssl.conf /etc/apache2/sites-available
--> c93872ac1342
Step 5/7 : COPY ./certs/bank32.crt ./certs/bank32.key /certs/
--> f795b7436e07
Step 6/7 : RUN chmod 400 /certs/bank32.key && chmod 644 $WWWDIR/index.html && chmod 644 $WWWDIR/index_red.html && a2ensite bank32_apache_ssl
--> Running in 4c12a0114cd7
Enabling site bank32_apache_ssl.
To activate the new configuration, you need to run:
  service apache2 reload
Removing intermediate container 4c12a0114cd7
--> 5f85be4e7ddf
Step 7/7 : CMD tail -f /dev/null
--> Running in cf9a7a13abeb
Removing intermediate container cf9a7a13abeb
--> 7045299b3413

Successfully built 7045299b3413
Successfully tagged seed-image-www-pki:latest
[10/12/25] seed@VM:~/.../Labsetup$
```

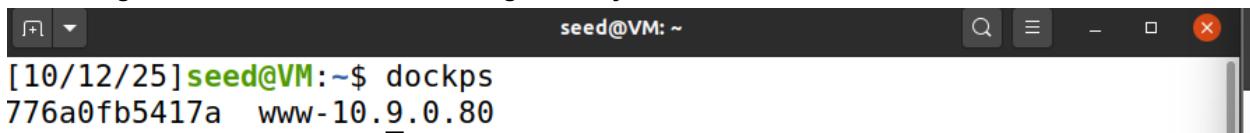
Running dcup Screenshot:



```
[10/10/25] seed@VM:~/.../Labsetup$ dcup
Creating network "net-10.9.0.0" with the default driver
Creating www-10.9.0.80 ... done
Attaching to www-10.9.0.80
```

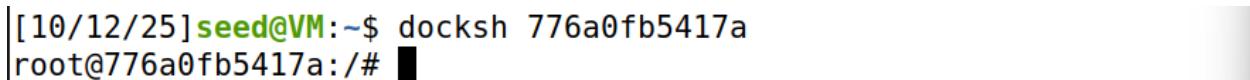
**Explanation:** This confirms that the Docker container was built and the Apache HTTPS server has started successfully.

**Confirming if the docker container is running correctly or not Screenshot:**



```
[10/12/25] seed@VM:~$ dockps  
776a0fb5417a  www - 10.9.0.80
```

**Starting the Docker container terminal:**



```
[10/12/25] seed@VM:~$ docksh 776a0fb5417a  
root@776a0fb5417a:/#
```

**Explanation:** I have used the same container ID to start the docker terminal so we can start Apache server.

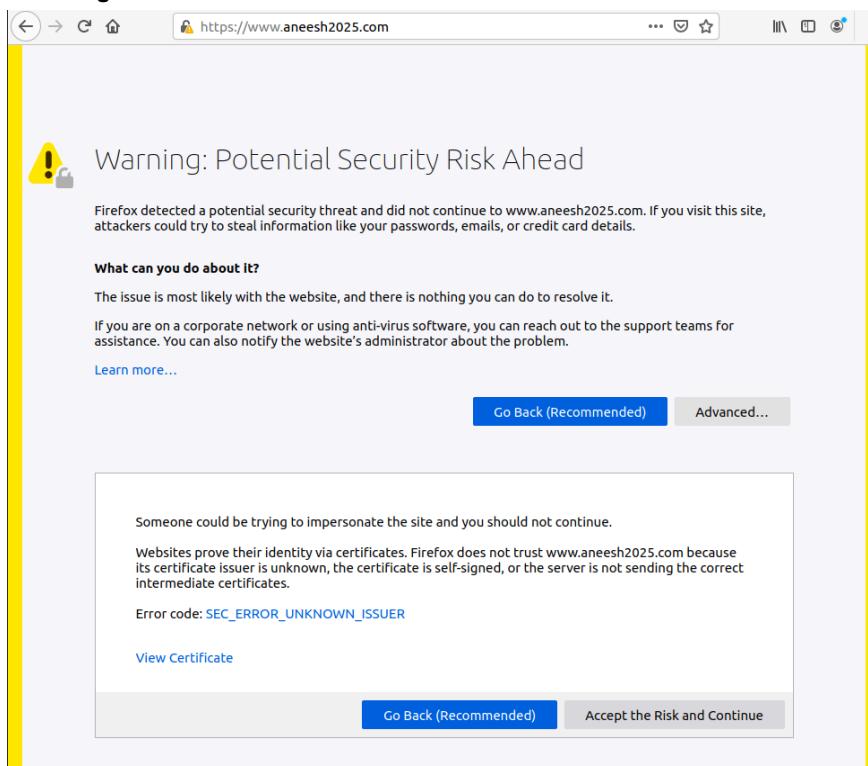
**Starting the server successfully Screenshot:**



```
[10/12/25] seed@VM:~$ docksh 776a0fb5417a  
root@776a0fb5417a:/# service apache2 start  
 * Starting Apache httpd web server apache2  
AH00112: Warning: DocumentRoot [/var/www/bank32] does not exist  
Enter passphrase for SSL/TLS keys for www.aneesh2025.com:443 (RSA):  
*  
root@776a0fb5417a:/#
```

**Description:** Here “dockps” shows all the running containers with their IDs and names. We can use the Id to get access to that container. “Docksh 776a0fb5417a” opens the shell session inside that specific container that can be seen running. Now we can run our further commands in the container's environment.

**Checking if the server is live ss:**



## Importing certificate in Authorities:

The screenshot shows the Firefox Certificate Manager interface. A modal dialog box titled "Downloading Certificate" is displayed over the main window. The dialog contains the following text and options:

You have been asked to trust a new Certificate Authority (CA).

Do you want to trust "Certificate Authority (unnamed)" for the following purposes?

Trust this CA to identify websites.

Trust this CA to identify email users.

Before trusting this CA for any purpose, you should examine its certificate and its policy and procedures (if available).

View    Examine CA certificate

Cancel    OK

Below the dialog, the main Certificate Manager window is visible. It has tabs for "Your Certificates", "Authentication Decisions", "People", "Servers", and "Authorities". The "Authorities" tab is selected. It lists certificates for three authorities:

Certificate Name	Security Device
IZENPE S.A. Izenpe.com	Builtin Object Token
Japan Certification Services, Inc. SecureSign RootCA11	Builtin Object Token
KI Aneesh	Software Security Device

Buttons at the bottom of the main window include: View..., Edit Trust..., Import..., Export..., Delete or Distrust..., and OK.

At the bottom of the Firefox window, there are links for "Extensions & Themes", "Firefox Support", and two radio button options for "Enable HTTPS-Only Mode":

- Enable HTTPS-Only Mode in all windows
- Enable HTTPS-Only Mode in private windows only



**Description and explanation:** Initially, when I tried to access the HTTPS website using the server name, the browser displayed a warning that the site may be unsafe or harmful. This happened because the browser did not yet trust the CA certificate that signed our server certificate. I then clicked "Advanced" and then "Accept the Risk and Continue" to bypass the warning. After doing this, the page loaded successfully as seen in the screenshot above. Browsers enforce certificate trust to ensure secure communication. Since our certificate was self-signed, the browser did not recognize it initially. Then, after importing the certificate in the Authorities and restarting the browser I could open the site without that warning. It can be seen from the screenshot above that there is no warning sign on lock

**Task 5:****Adding another virtual host to redirect them to another website Screenshot:**

```
root@776a0fb5417a: /etc/apache2/sites-available/bank32_apache_ssl.conf
<VirtualHost *:443>
    DocumentRoot /var/www/aneesh2025
    ServerName www.aneesh2025.com
    ServerAlias www.aneesh2025A.com
    ServerAlias www.aneesh2025B.com
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /root/Labsetup1/server.crt
    SSLCertificateKeyFile /root/Labsetup1/server.key
</VirtualHost>

<VirtualHost *:443>
    DocumentRoot /var/www/bank32
    ServerName www.example.com
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /root/Labsetup1/server.crt
    SSLCertificateKeyFile /root/Labsetup1/server.key
</VirtualHost>
```

**Description:** Here we are changing configuration to add our alternate url that the victim will add and looks legit. Everything stays the same except for the serverName.

**Restarting apache server again:**

```
root@776a0fb5417a:/# 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@776a0fb5417a:/# a2ensite bank32_apache_ssl.conf
Site bank32_apache_ssl already enabled
root@776a0fb5417a:/# service apache2 reload
 * Reloading Apache httpd web server apache2
 *
root@776a0fb5417a:/# service apache2 restart
 * Restarting Apache httpd web server apache2
AH00112: Warning: DocumentRoot [/var/www/bank32] does not exist
AH00112: Warning: DocumentRoot [/var/www/bank32] does not exist
Enter passphrase for SSL/TLS keys for localhost:443 (RSA):
 [ OK ]
```

**Description:** We restarted the server so that the new url can be loaded in and the victim can run it on their browser to check if they are reverted to the actual url that the victim shouldn't run.

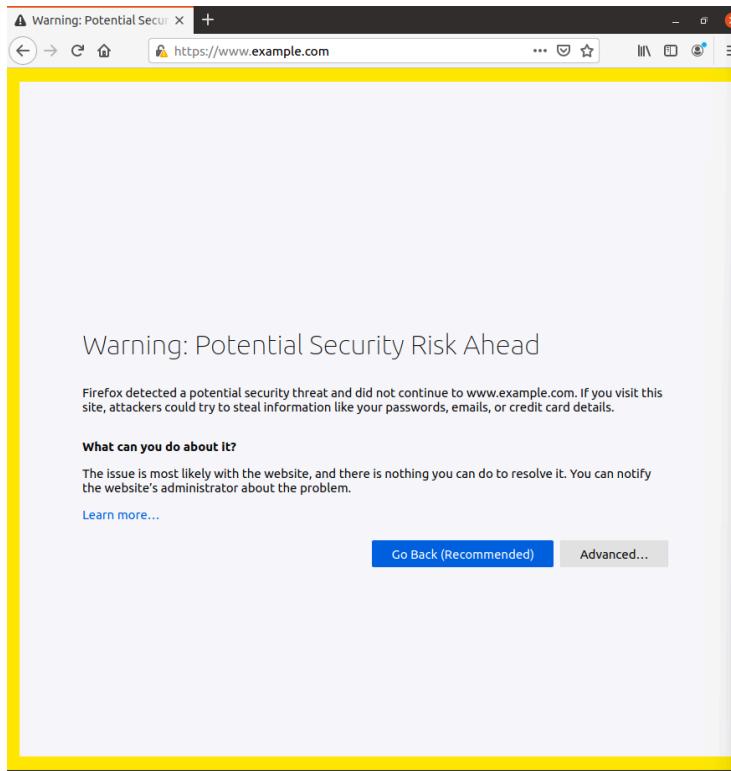
**Adding url to hosts file:**

```
# For Shellshock Lab
10.9.0.80      www.seedlab-shellshock.com
10.9.0.80      www.aneesh2025.com
10.9.0.80      www.example.com
```

**Restarting the apache servers after adding new url:**

```
root@776a0fb5417a:/# service apache2 restart
 * Restarting Apache httpd web server apache2
AH00112: Warning: DocumentRoot [/var/www/bank32] does not exist
AH00112: Warning: DocumentRoot [/var/www/bank32] does not exist
Enter passphrase for SSL/TLS keys for localhost:443 (RSA):
                                              [ OK ]
root@776a0fb5417a:/# service apache2 restart
 * Restarting Apache httpd web server apache2
AH00112: Warning: DocumentRoot [/var/www/bank32] does not exist
AH00112: Warning: DocumentRoot [/var/www/bank32] does not exist
Enter passphrase for SSL/TLS keys for localhost:443 (RSA):
                                              [ OK ]
root@776a0fb5417a:/# █
```

**Testing url on browser:**



**Observations and Explanation:** When we access [www.example.com](https://www.example.com), as we are using the same certificate as in [www.aneesh2025.com](https://www.aneesh2025.com), the browser warns us that this certificate is not for [example.com](https://example.com) by comparing the visited URL with the Common Name present in the certificate. We can see that warning in the following image. We successfully rerouted traffic intended for [www.example.com](https://www.example.com) to our malicious web server at 10.9.0.80. The only reason it failed is because when our server presented its SSL certificate to the browser, Firefox performed a PKI validation check and saw that there is a difference between the domain name in the certificate and the requested website URL. Hence, PKI defeats MITM attacks through certificate validation.

**Task 6:**

Generating certificate request for example website:

```
[10/12/25] seed@VM:~$ openssl req -newkey rsa:2048 -sha256 -keyout fake.key -out fake.csr -subj "/CN=www.example.com/0@example" -passout pass:dees
Generating a RSA private key
...+++++
.....+++++
writing new private key to 'fake.key'
-----
[10/12/25] seed@VM:~$
```

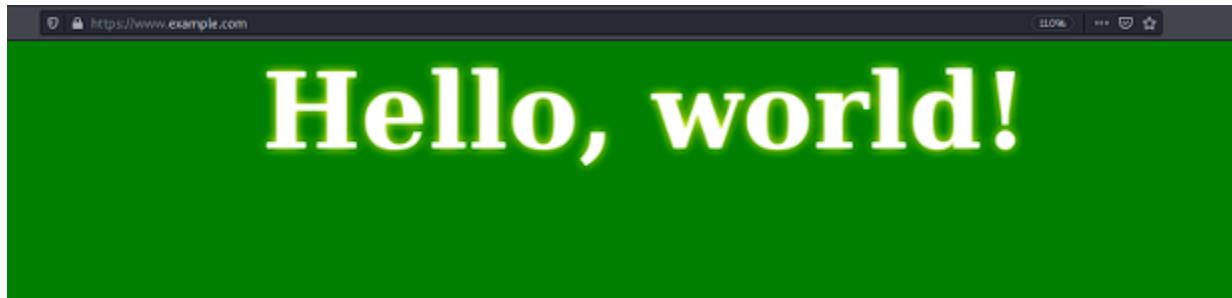
Generating the certificate for the server:

```
[10/12/25] seed@VM:~/.../Labsetup$ openssl ca -config openssl.cnf -policy policy_anything -md sha256 -days 3650 -in fake.csr -out fake.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: 4097 (0x1001)
    Validity
        Not Before: Oct 12 09:50:51 2025 GMT
        Not After : Oct 10 09:50:51 2035 GMT
    Subject:
        organizationName      = example
        commonName            = www.example.com
    X509v3 extensions:
        X509v3 Basic Constraints:
            CA:FALSE
        Netscape Comment:
            OpenSSL Generated Certificate
        X509v3 Subject Key Identifier:
            24:3B:EF:94:97:B7:9E:8A:C0:14:8E:BC:99:ED:DB:80:88:
07:4B:53
        X509v3 Authority Key Identifier:
            keyid:DC:75:56:7A:B4:18:FE:1C:1A:27:76:2B:C5:95:09:
A4:41:1A:54:3B
    Certificate is to be certified until Oct 10 09:50:51 2035 GMT (3650
    days)
    Write out database with 1 new entries
Data Base Updated
```

**Moving new files to the docker container:**

```
[10/12/25] seed@VM:~$ sudo docker cp /home/seed/Downloads/Labsetup/i  
mage_www/certs/fake.crt www-10.9.0.80:/certs/fake.crt  
[10/12/25] seed@VM:~$ sudo docker cp /home/seed/Downloads/Labsetup/i  
mage_www/certs/fake.key www-10.9.0.80:/certs/fake.key
```

After restarting the apache2 server and running the example url on browser it successfully loaded without any warnings:



**Observations and explanation:** once everything was done correctly, when I visited new url [www.example.com](https://www.example.com), the page loaded without any warnings and browser also showed secure lock. When I looked at the certificate it was also issued to the [example.com](https://example.com), but signed by my old seed lab which was connected to [aneesh2025.com](https://aneesh2025.com), shows thats browser trusted the forged certificate and attackers server could impersonate the site completely.