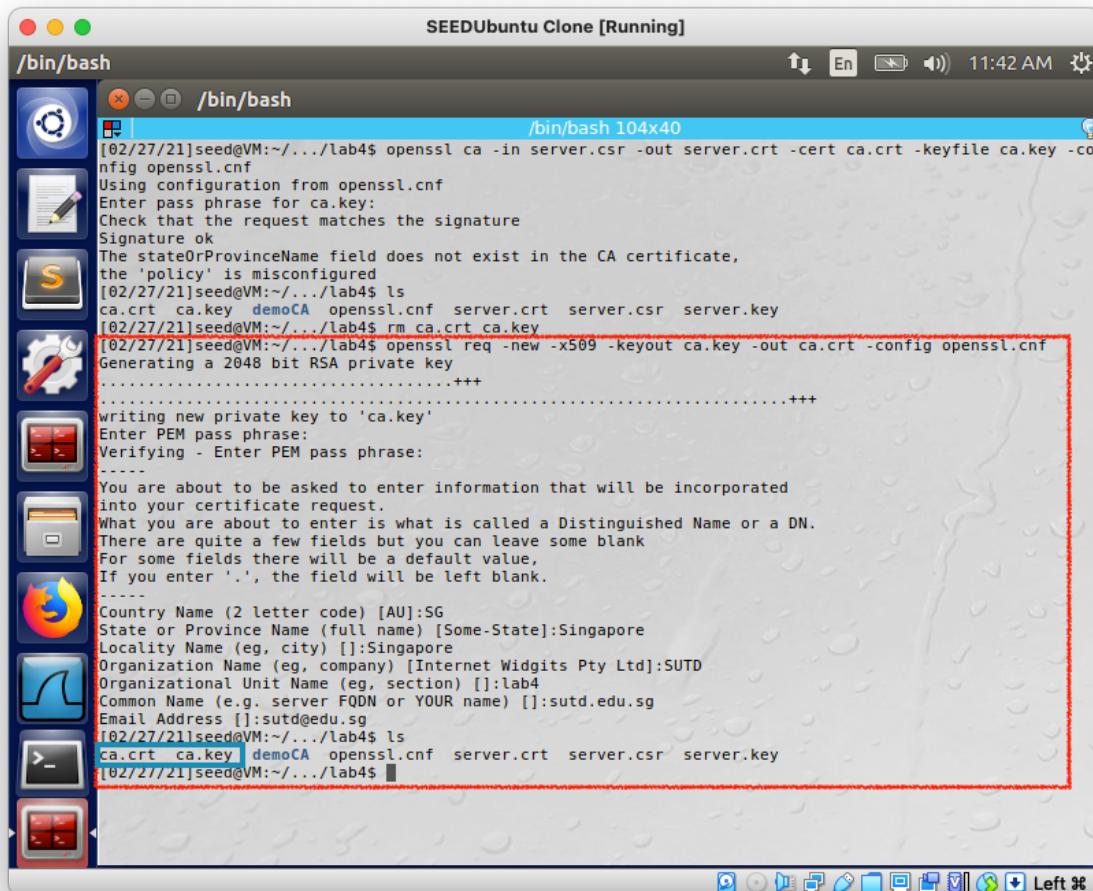


Author: Wong Tin Kit  
Student ID: 1003331

This task was done on SEED Ubuntu Clone of IP 10.0.2.10

## Task 1: Becoming a Certificate Authority (CA)



```
[02/27/21]seed@VM:~/.../lab4$ openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key -co
nfig openssl.cnf
Using configuration from openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
The stateOrProvinceName field does not exist in the CA certificate,
the 'policy' is misconfigured
[02/27/21]seed@VM:~/.../lab4$ ls
ca.crt ca.key demoCA openssl.cnf server.crt server.csr server.key
[02/27/21]seed@VM:~/.../lab4$ rm ca.crt ca.key
[02/27/21]seed@VM:~/.../lab4$ openssl req -new -x509 -keyout ca.key -out ca.crt -config openssl.cnf
Generating a 2048 bit RSA private key
.....+
writing new private key to 'ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:SG
State or Province Name (full name) [Some-State]:Singapore
Locality Name (eg, city) []:Singapore
Organization Name (eg, company) [Internet Widgits Pty Ltd]:SUTD
Organizational Unit Name (eg, section) []:lab4
Common Name (e.g. server FQDN or YOUR name) []:sutd.edu.sg
Email Address []:sutd@edu.sg
[02/27/21]seed@VM:~/.../lab4$ ls
ca.crt ca.key demoCA openssl.cnf server.crt server.csr server.key
[02/27/21]seed@VM:~/.../lab4$
```

Fig. 1 Getting CA certificate and key

The PEM pass phrase used is **tinkit**.

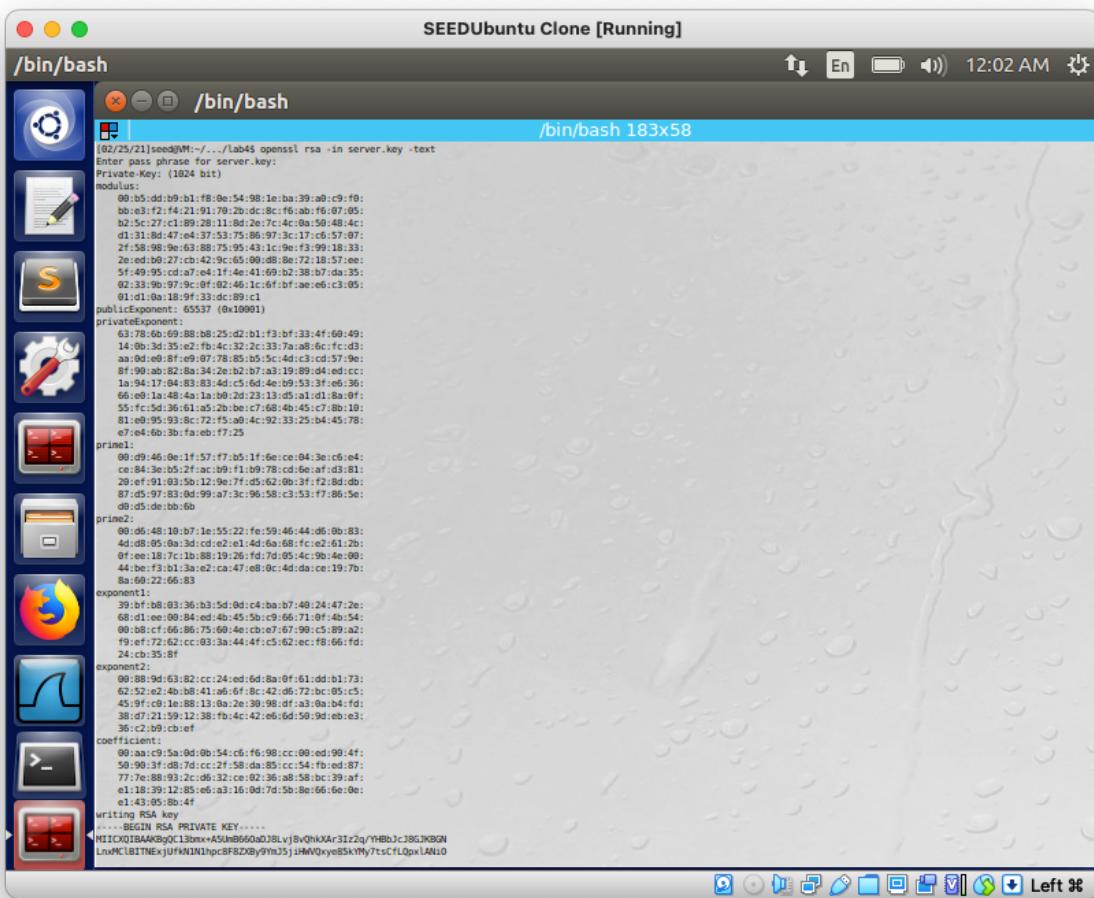
Fig. 1 shows the process in creating **ca.crt** and **ca.key** in the red highlighted box. We confirm the creation with **ls** and see that the **ca.crt** and **ca.key** are created.

# Task 2: Creating a Certificate for SEEDPKILab2020.com

## Step 1: Generate public/private key pair.

Passphrase for server.key: seedpkilab2020

The image below depicts the text output of **server.key**.



```
[02/25/21] seed@M-:~/.../lab45 openssl rsa -in server.key -text
Enter pass phrase for server.key:
-----
Private Key: (1024 bit)
modulus:
00:b5:dd:b9:1f:8e:54:98:1e:ba:39:a0:c9:f9:
bb:e3:f2:f4:21:91:70:2b:dc:8c:f6:ab:f6:07:05:
b2:5c:27:c1:89:28:11:8d:2e:c7:4c:8a:59:48:4c:
d1:31:8d:47:e4:37:53:75:86:97:3c:17:c8:57:07:
2f:58:98:9e:09:88:75:95:43:1c:9e:f3:99:18:33:
2e:0d:98:90:2e:8a:30:22:05:40:03:8e:72:18:57:ee:
5f:49:95:cd:a7:e4:1f:4e:41:69:h2:38:b7:da:35:
02:33:9b:97:9c:0f:02:46:1c:6f:bf:ae:ed:c3:85:
01:d1:8a:18:9f:33:dc:89:c1
publicExponent: 65537 (0x8901)
privateExponent:
63:78:6b:69:88:bb:25:d2:b1:f3:bf:33:4f:60:49:
14:0b:3d:35:09:fb:4c:32:2a:33:1a:8b:6c:f0:43:
3d:0b:69:09:3d:10:11:05:05:4d:01:89:04:3e:36:
8f:90:ab:82:8a:34:2e:hb:27:a3:19:89:d4:ad:cc:
1a:94:17:04:83:83:44:c5:8d:4e:b9:53:3f:e6:36:
66:ee:91:1a:48:4a:1a:bb:2d:23:13:d5:a1:d1:8a:36:
55:fc:c5:d3:61:a5:2b:be:c7:68:4b:45:c7:bb:10:
81:eb:95:93:8c:72:f5:ab:4c:92:33:25:b4:45:78:
e7:e4:6b:3b:fa:eb:f7:25
prime1:
00:d9:46:10:b7:1e:55:22:fe:59:46:44:d6:6b:83:
4d:db:05:0a:3d:cde2:el:4d:6a:68:fc:e2:61:2b:
07:0b:87:7c:1b:88:19:26:fd:7d:7d:05:4c:9b:4e:00:
44:be:7a:01:3a:e2:ca:47:e8:0c:4d:da:ce:19:7b:
8a:69:22:66:83
prime2:
39:bf:bb:03:36:b3:5d:0d:c4:ba:bf:7:40:24:47:2e:
68:di:ee:00:84:ed:4b:45:4b:c9:66:71:0f:4b:54:
00:bb:cf:66:86:75:69:4e:cb:e7:67:90:c5:89:a2:
19:ef:72:62:cc:03:3a:44:4f:c5:62:ec:f8:66:fd:
24:cb:35:8f
exponent:
00:89:9d:63:82:cc:24:ed:6d:8a:0f:61:dd:bb:1:73:
62:52:02:4b:4b:41:a6:67:8c:42:d6:72:bc:05:c5:
45:9f:c9:1e:88:13:0a:2e:39:98:df:a3:0a:ba:4:fd:
38:df:21:59:12:38:fb:4c:42:e6:6d:50:9d:eb:e3:
36:c2:b9:cb:ef
coefficient:
00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:
00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:
writing RSA key
-----BEGIN RSA PRIVATE KEY-----
MIIECQIBAAQByQ130nx+A5UhB660aJ8LyjBvQhkXAr31zq/YHbJcJBGJKBN
LnMC(BITNE)xUkN1N1hpcF8ZQBy9Ym75j1HMVQxy85kYH7tsCLOpXLAN10
```

Fig. 2.1 Output of **server.key**

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

```
SEEDUbuntu Clone [Running]
/bin/bash
/bin/bash 104x40
JuefolloVg375AuVdWoucTaP0JFABZtt5rtnpg7x/VsyXqxlytDLE/wqaLRW5aGG
n98T3lPmvb/Ve9VeSb/3FcZGfy7IWIf0z1ffgG4sQKb1VMyosySuG6I0D0LY5tK0
bpZktADKkms4ExZ324Lg0XLVcdeQmE5QVJ044l5MoYmxPQ==
----END CERTIFICATE-----
[02/27/21]seed@VM:~/.../lab4$ openssl s_server -cert server.pem -www
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
^C
[02/27/21]seed@VM:~/.../lab4$ ^C
[02/27/21]seed@VM:~/.../lab4$ ^C
[02/27/21]seed@VM:~/.../lab4$ ls
ca.crt ca.key demoCA openssl.cnf server.crt server.csr server.key server.pem
[02/27/21]seed@VM:~/.../lab4$ rm server.crt server.csr server.pem
[02/27/21]seed@VM:~/.../lab4$ ls
ca.crt ca.key demoCA openssl.cnf server.key
[02/27/21]seed@VM:~/.../lab4$ openssl req -new -key server.key -out server.csr -config openssl.cnf
Enter pass phrase for server.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:SG
State or Province Name (full name) [Some-State]::
Locality Name (eg, city) []:Singapore
Organization Name (eg, company) [Internet Widgits Pty Ltd]:seedpki
Organizational Unit Name (eg, section) []:lab4
Common Name (e.g. server FQDN or YOUR name) []:SEEDPKILab2020.com
Email Address []:SEEDPKILab2020@edu.sg
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
[02/27/21]seed@VM:~/.../lab4$ ls
ca.crt ca.key demoCA openssl.cnf server.csr server.key
[02/27/21]seed@VM:~/.../lab4$
```

Fig. 2.2 Configuration in generating the certificate signing request.

As requested, I used SEEDPKILab2020.com as the Common Name in this CSR.

## Step 3: Generating Certificates.

The screenshot shows a terminal window titled "SEEDUbuntu Clone [Running]" with the command "/bin/bash". The terminal content is as follows:

```
[02/27/21]seed@VM:~/.../lab4$ ls  
ca.crt ca.key demoCA openssl.cnf server.csr server.key  
[02/27/21]seed@VM:~/.../lab4$ openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key \  
> -config openssl.cnf  
Using configuration from openssl.cnf  
Enter pass phrase for ca.key:  
Check that the request matches the signature  
Signature ok  
Certificate Details:  
    Serial Number: 4098 (0x1002)  
    Validity  
        Not Before: Feb 28 03:21:54 2021 GMT  
        Not After : Feb 28 03:21:54 2022 GMT  
    Subject:  
        countryName = SG  
        localityName = Singapore  
        organizationName = seedpki  
        organizationalUnitName = lab4  
        commonName = SEEDPKILab2020.com  
        emailAddress = SEEDPKILab2020@edu.sg  
X509v3 extensions:  
    X509v3 Basic Constraints:  
        CA:FALSE  
    Netscape Comment:  
        OpenSSL Generated Certificate  
X509v3 Subject Key Identifier:  
        F6:C5:D7:A8:DB:F4:B5:D8:0A:B4:B6:B4:EE:4E:DC:EA:02:AE:22:B0  
X509v3 Authority Key Identifier:  
        keyid:56:08:3B:C5:F1:71:1C:5C:91:6F:01:CA:F4:0E:4D:7F:60:0A:5F:97  
Certificate is to be certified until Feb 28 03:21:54 2022 GMT (365 days)  
Sign the certificate? [y/n]:y  
  
1 out of 1 certificate requests certified, commit? [y/n]  
Write out database with 1 new entries  
Data Base Updated  
[02/27/21]seed@VM:~/.../lab4$ ls  
ca.crt ca.key demoCA openssl.cnf server.crt server.csr server.key  
[02/27/21]seed@VM:~/.../lab4$
```

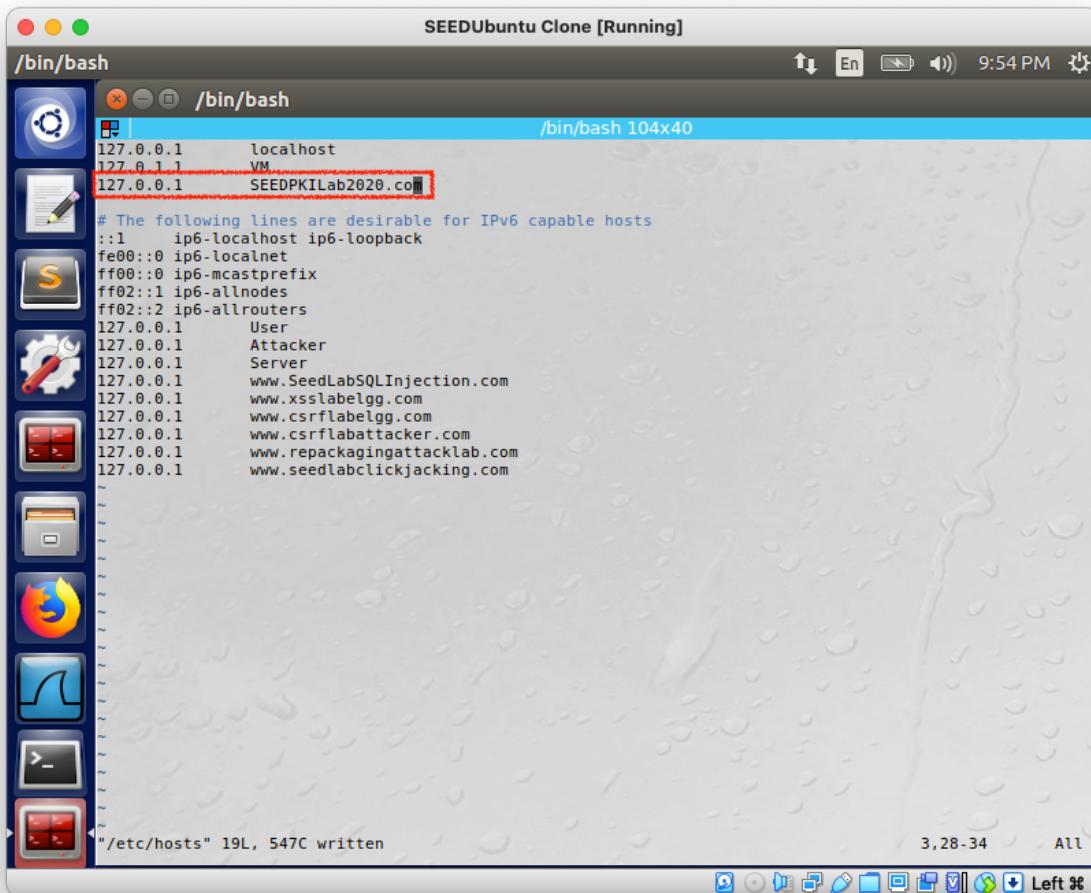
A cyan box highlights the word "server.crt" in the final command line. The terminal window has a red border around its main content area.

Fig. 2.3 Completion of CSR

In this case, I configured the policy section ***openssl.cnf*** to match anything. Fig. 2.3 shows the successful completion of the CA signing a CSR. We see ***server.crt*** as a result (highlighted in the cyan box).

# Task 3: Deploying Certificate in an HTTPS Web Server

## Step 1: Configuring DNS.



The screenshot shows a terminal window titled "SEEDUbuntu Clone [Running] /bin/bash". The window contains a command-line interface where a user has added a new host entry to the "/etc/hosts" file. The entry "127.0.0.1 SEEDPKILab2020.com" is highlighted with a red box. The terminal also displays a list of other localhost mappings and various website addresses. The bottom status bar shows the command was run in "/etc/hosts" and 547C were written.

```
SEEDUbuntu Clone [Running]
/bin/bash
127.0.0.1      localhost
127.0.1.1      VM
127.0.0.1      SEEDPKILab2020.com
# 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
"/etc/hosts" 19L, 547C written
3,28-34     All
```

Fig. 3.1 Adding SEEDPKILab2020.com to /etc/hosts

Fig. 3.1 shows the addition of the mapping of localhost to **SEEDPKILab2020.com**.

## Step 2: Configuring the web server.

```

SEEDUbuntu Clone [Running]
/bin/bash
/bin/bash 104x40
62:8f:e7:a9:b0:4b:f0:87:d2:47:bb:b1:20:ae:e4:c5:24:be:
94:c0:0f:b5:33:69:46:4f:95:41:92:f2:ad:96:04:dc:b8:f9:
02:39:12:17:1e:c7:82:67:28:24:80:fd:3e:0a:ea:f9:ce:50:
26:e7:9f:a2:5d:68:56:0d:fb:e4:0b:95:75:6a:2e:71:36:8f:
d0:91:40:6d:9b:6d:e6:bb:67:a6:0e:f1:fd:5b:32:5e:ac:75:
ca:d0:cb:13:fc:2a:68:b4:56:e5:a1:86:9f:df:13:de:53:e6:
bd:bf:d5:7b:d5:5e:49:bf:f7:15:c6:46:7f:2e:c8:58:87:f4:
cf:57:df:80:6e:2c:40:a6:f5:54:cc:a8:b3:24:ae:1b:a2:0e:
0f:42:d8:e6:d2:b4:6e:96:64:b4:00:ca:92:6b:38:13:16:77:
db:82:e0:39:72:d5:71:d7:90:98:44:90:54:93:b8:e2:5e:4c:
a1:89:b1:3d
-----BEGIN CERTIFICATE-----
MIIDfjCCAmagAwIBAgICEAEwDQYJKoZIhvvcNAQELBQAwgYUxCzAJBgNVBAYTAlNH
MRIwEAYDVQQIDAUTaW5nYXBvcmUxEjAQBgNVBAcMCVNpbmdhcG9yZTENMAsGA1UE
CgwEU1VURDENMAsGA1UECwwEbGFiNDEUMBIGA1UEAwLc3V0ZC5LZHOUczCxGjAY
BgkqhkiG9w0BC0EW3NiGraZWR1LnNnMB4DTIxMDIy0DAyNTewNl0xDiTiyMDIy
0DAyNTewNl0wgYQxCzAJBgNVBAYTA1NHRiwEAYDVQHQDALTaW5nYXBvcmUxZAN
BgNVBAoMbNbraWxhYjENMAsGA1UECwwEbGFiNDEbMBkGA1UEAwSu0VFRFBJS0xh
YjIwMjAuY29tMSQwIgYJKoZIhvvcNA0kBFhVzWWkcGtpbGF1mJyMEB1ZHOUc2cw
gZ8wDQYJKoZIhvvcNAQEBBQADgYBAMIGJAoGBAPE1ymyPnvAexH+b4SGCrV1MuJAU
8+h0hF0IH57dmyqIqZEM0jVsDahRwZX6j7Xky3+1wc2ACgxAAAnuzyIPIjx0sta
t0yAT7BkoK8jKR7H0Tx3BMQtjvE0SD2qPY7y6CWMQqoN+ZCH4ImFx0V/tynSWcek
0hmY91Qo0KBIAV8rAgMBAAGjezB5MAkGA1UdEwQCMAAwLAYJYIZIAy40gENBB8W
HU9wZWSTU0wgR2VuZXJhdGVkIE1cnRpZmljYXR1MB0GA1UdDgQWB8T2xdeo2/S1
2Aq0trTuTtzqAq4isDAfBgNVHSMEGDAwgBRWCDvF8XEcXJFvAcr0Dk1/YApflzAN
BgkqhkiG9w0BAoQFAAOCAQEAXm0BmZTnTjkwroPLD+UB0KDmskfaHFZbGI82bjis
x10GxsXVSFpI4iWLn1zjI25M3lywuduF0eIFGmyC9fBsSdg27hwc2dMy0/ngbBL
8IfSR7uxIK7kxSS+IMAPtTNpRk+VQZLyRZYE3Lj5AjksFx7Hgmc0JID9Pgrq+c5Q
JuefolloVg375AuVdWouCTaP0JFAbTtt5rtnpg7x/VsyQxlytDLE/wqaLRW5aGG
n9813lpmb/Ve9Ye5b/3FcZGfy/IWlf0z1fttgG4sQKb1VMyosySuG610D8LY5tK0
bpZktADKkms4ExZ324Lq0XLVcdeQmESQVJ044l5MoYmxPQ==
-----END CERTIFICATE-----
[02/27/21]seed@VM:~/.../lab4$ openssl s_server -cert server.pem -www
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
ACCEPT
ACCEPT
ACCEPT

```

Fig. 3.2 Running OpenSSL HTTPS server

Fig. 3.2 shows the stdout of running the openssl https server and mozilla attempts to connect to it. We run the command as such ***openssl s\_server -cert server.pem -www***. The ***www*** flag helps in debugging and sends a status message back to the client when it connects. This includes lots of information about the ciphers used and various session parameters. The output is in HTML format so this option will normally be used with a web browser.

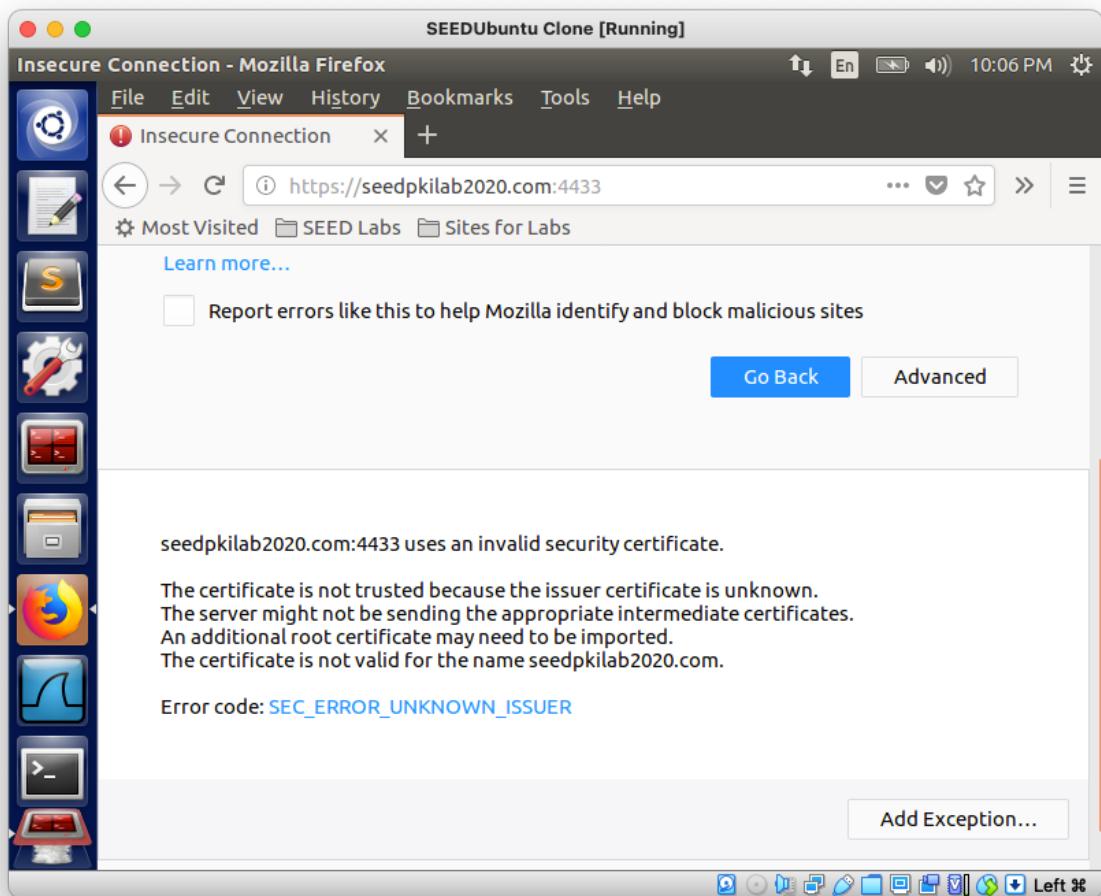


Fig. 3.3 Mozilla Browser Connection to <https://seedpkilab2020.com>

Fig. 3.3 shows an attempt by Mozilla Browser to connect to <https://seedpkilab2020.com>. This warning is expected as the browser does not recognise the Issuer of the website's certificate after checking it against Mozilla's list of trusted CAs.

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

Load **ca.crt** into Mozilla Firefox Browser by doing the following:

Edit -> Preference -> Privacy & Security -> View Certificates. -> Authorities -> Import -> Load **ca.crt**

## Step 4. Testing our HTTPS website.

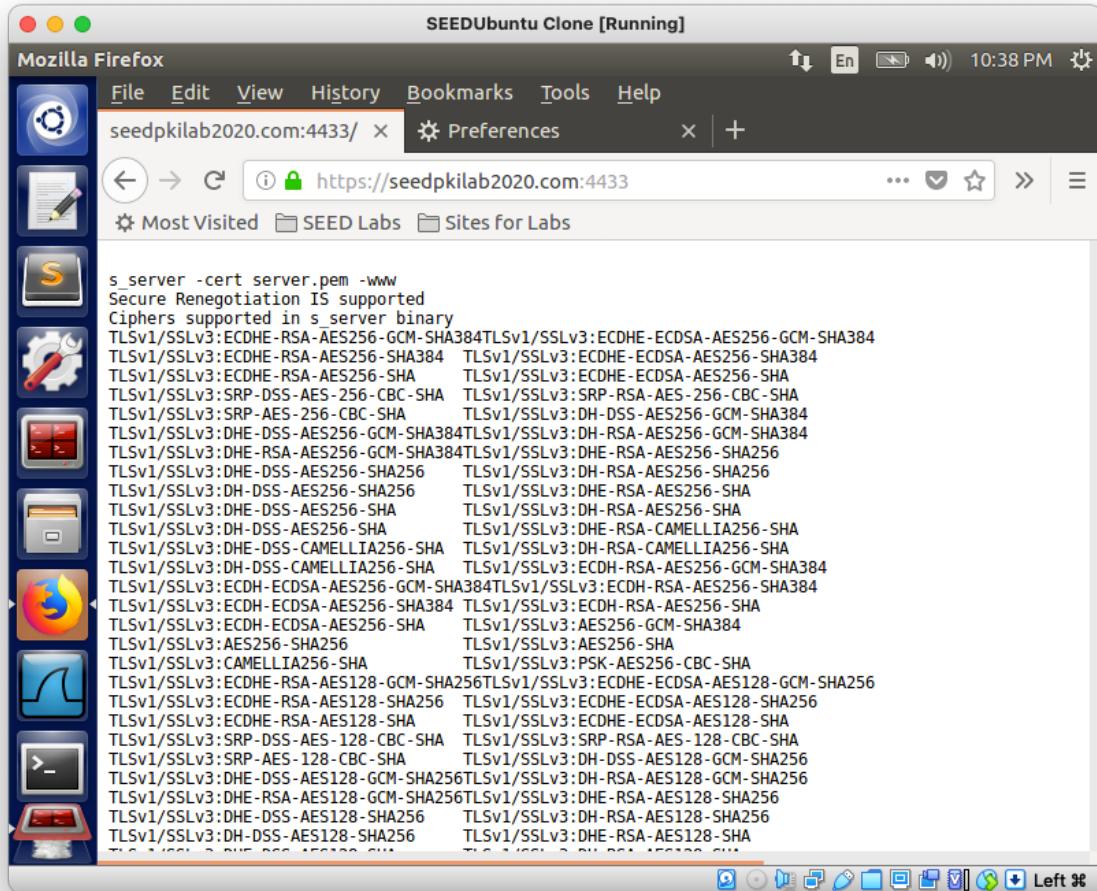
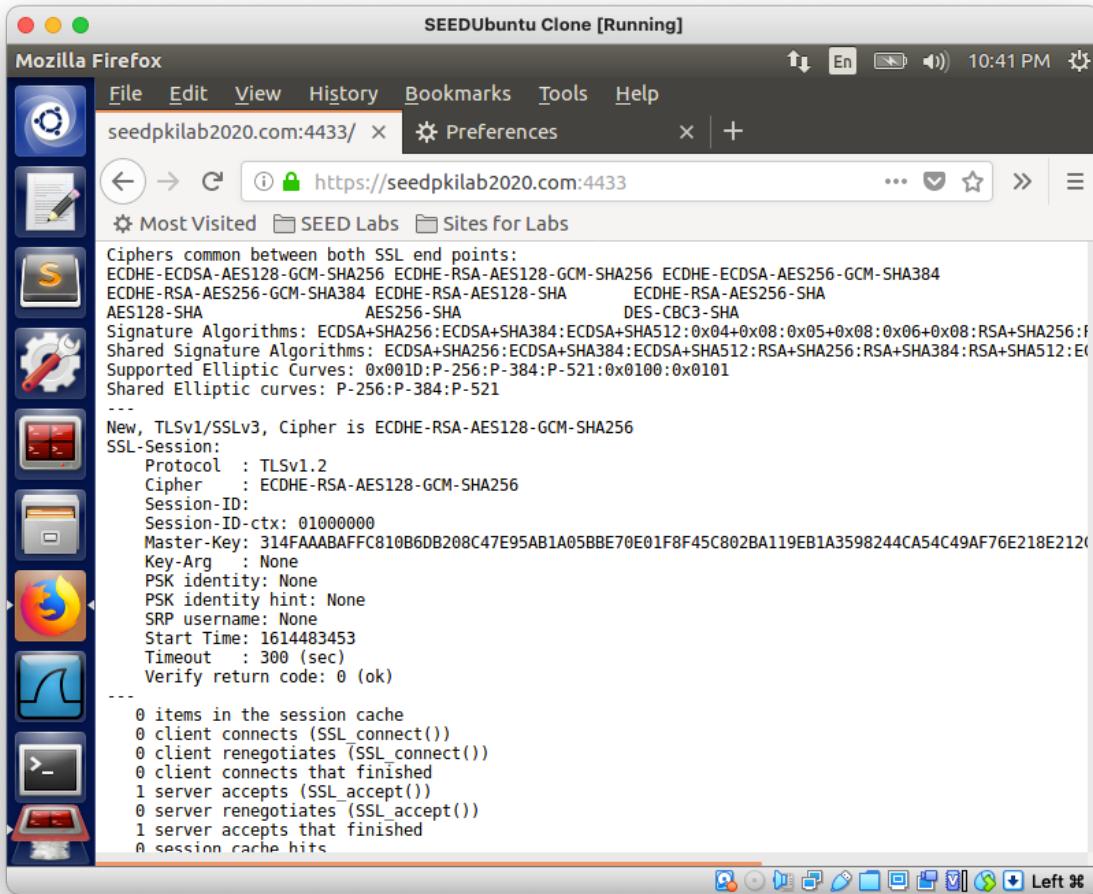


Fig. 3.4 First portion of information sent back to Mozilla Client



The screenshot shows a Mozilla Firefox window titled "SEEDUbuntu Clone [Running]". The address bar displays "seedpkilab2020.com:4433/" and the URL "https://seedpkilab2020.com:4433". The main content area shows the following text output from an SSL/TLS session:

```

Ciphers common between both SSL end points:
ECDHE-ECDSA-AES128-GCM-SHA256 ECDHE-RSA-AES128-GCM-SHA256 ECDHE-ECDSA-AES256-GCM-SHA384
ECDHE-RSA-AES256-GCM-SHA384 ECDHE-RSA-AES128-SHA ECDHE-RSA-AES256-SHA
AES128-SHA AES256-SHA DES-CBC3-SHA
Signature Algorithms: ECDSA+SHA256:ECDSA+SHA384:ECDSA+SHA512:0x04+0x08:0x05+0x08:0x06+0x08:RSA+SHA256:+
Shared Signature Algorithms: ECDSA+SHA256:ECDSA+SHA384:ECDSA+SHA512:RSA+SHA256:RSA+SHA384:RSA+SHA512:E
Supported Elliptic Curves: 0x001D:P-256:P-384:P-521:0x0100:0x0101
Shared Elliptic curves: P-256:P-384:P-521
---
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES128-GCM-SHA256
SSL-Session:
Protocol : TLSv1.2
Cipher   : ECDHE-RSA-AES128-GCM-SHA256
Session-ID:
Session-ID-ctx: 01000000
Master-Key: 314FAAAABAFFC810B6DB208C47E95AB1A05BBE70E01F8F45C802BA119EB1A3598244CA54C49AF76E218E212C
Key-Ag  : None
PSK identity: None
PSK identity hint: None
SRP username: None
Start Time: 1614483453
Timeout   : 300 (sec)
Verify return code: 0 (ok)
---
0 items in the session cache
0 client connects (SSL_connect())
0 client renegotiates (SSL_connect())
0 client connects that finished
1 server accepts (SSL_accept())
0 server renegotiates (SSL_accept())
1 server accepts that finished
0 session cache hits

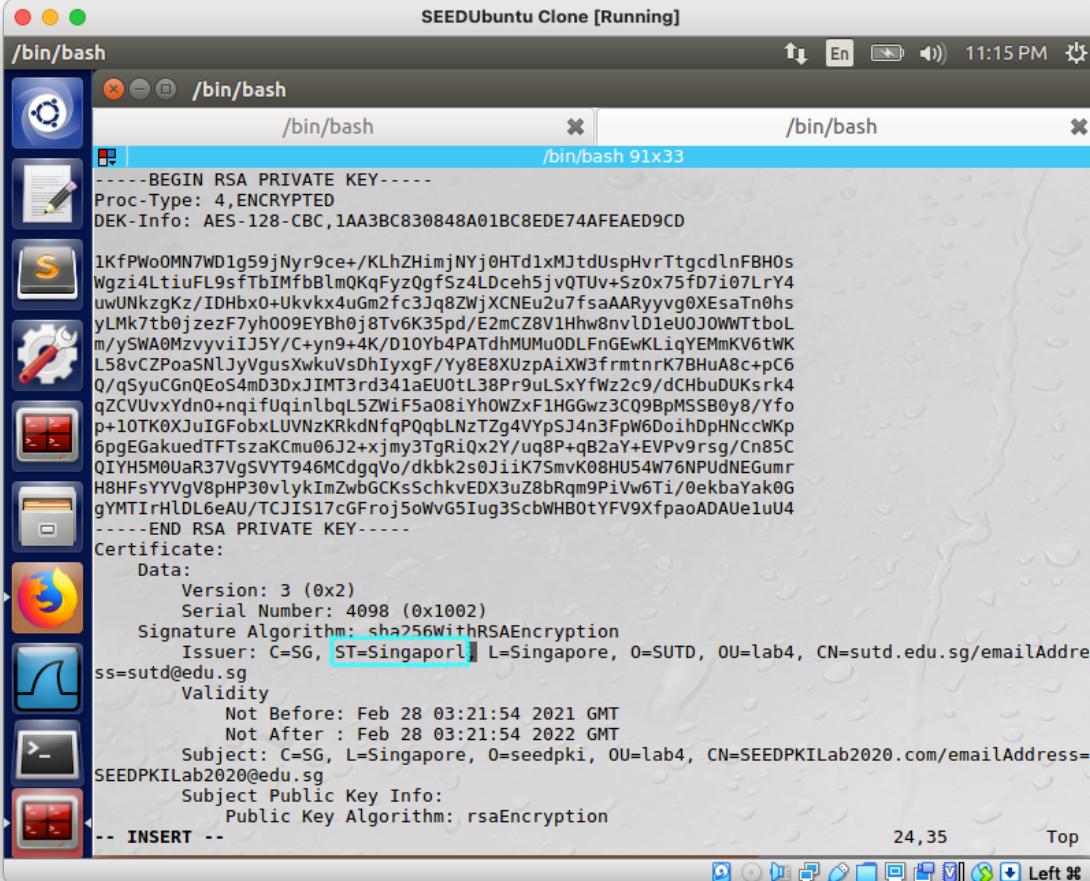
```

Fig. 3.5 Second portion of information sent back to Mozilla Client

Figures 3.4 and 3.5 show the entire set of status messages sent back to Mozilla Client from the openssl HTTPS server. Fig. 3.4 shows the available cipher suites supported by the openssl server. The start of Fig. 3.5 shows the cipher suits common to both SSL endpoints (openssl server and Mozilla Client). The second half of Fig. 3.5 shows the new SSL-Session setup between Mozilla Client and OpenSSL server.

## Additional Tasks

1. Modify a single byte of server.pem, and restart the server, and reload the URL. What do you observe?



```
SEEDUbuntu Clone [Running]
/bin/bash
/bin/bash
/bin/bash 91x33
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,1AA3BC830848A01BC8EDE74AFEAED9CD

1KfPWoOMN7WD1g59jNyr9ce+/KLhZHimjNYj0HTd1xMJtdUspHvrTtgcdlnFBH0s
Wgzi4ltiuFL9sfTbIMfbBmQKqFyzQgfSz4LDceh5jvQTUv+SzOx75fd7i07LrY4
uwUNkzgKz/IDHbx0+Ukvkx4uGm2fc3Jq8ZwjXCNEu2u7fsaAAryyvg0xEsaTn0hs
yLMk7tb0jzezF7yh009EYBh0j8Tv6K35pd/E2mCZ8V1Hhw8nvld1eU0j0WWTTboL
m/ySWA0MzvyviiJ5Y/C+yn9+4K/D10Yb4PATdhMUMu0DLFnGEwkLiqYEMmkV6tWK
L58vCZPoaSNLJyVgusXwkuVsDhIyxgF/Yy8E8XUzpAiXW3frmtnrK7BHua8c+pc6
0/qSyuCnQeoS4mD3DxJIMT3rd341aEU0tL38Pr9uL5xYfwz2c9/dChbuDUksrk4
qZCVUvxYdn0+nqifUqinlbql5ZWiF5a08iYh0WZxF1HGGwz3CQ9BpMSSB0y8/Yfo
p+10TK0XJu1GFobxLUVNzKRkdNfqPQqlLNzTzg4VYpSJ4n3Fpw6DoihDpHNccWkp
6pgEGakuedFTTszAKCmu06J2+xjmy3TgrRiQx2Y/uq8P+qB2aY+EVpv9rsq/Cn85C
QIYH5M0UaR37VgSVYT946MCdgqVo/dkbk2s0JiiK7SmvK08HU54W76NPUDNEGumr
H8HFsvYYVgV8pHP30vlykImZwbGCKsSchkVEDX3uZ8bRqm9PiVwGTi/0ekbaYak0G
gYMTIrHldL6eAU/TCJIS17cGFr0j5oWVG5Iug3ScbWHBoTYFV9XfpaoADAUeluuU4
-----END RSA PRIVATE KEY-----
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number: 4098 (0x1002)
    Signature Algorithm: sha256WithRSAEncryption
      Issuer: C=SG, ST=Singapore, L=Singapore, O=SUTD, OU=lab4, CN=sutd.edu.sg/emailAddress=sutd@edu.sg
      Validity
        Not Before: Feb 28 03:21:54 2021 GMT
        Not After : Feb 28 03:21:54 2022 GMT
      Subject: C=SG, L=Singapore, O=seedpki, OU=lab4, CN=SEEDPKILab2020.com/emailAddress=SEEDPKILab2020@edu.sg
      Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
-- INSERT --
```

Fig. 3.4.1 Changing ST field in **server.pem**

In Fig. 3.4.1, we changed the ST field to **Singaporl** and reloaded the openssl server before connecting to it via Mozilla as shown in Fig. 3.4.2.

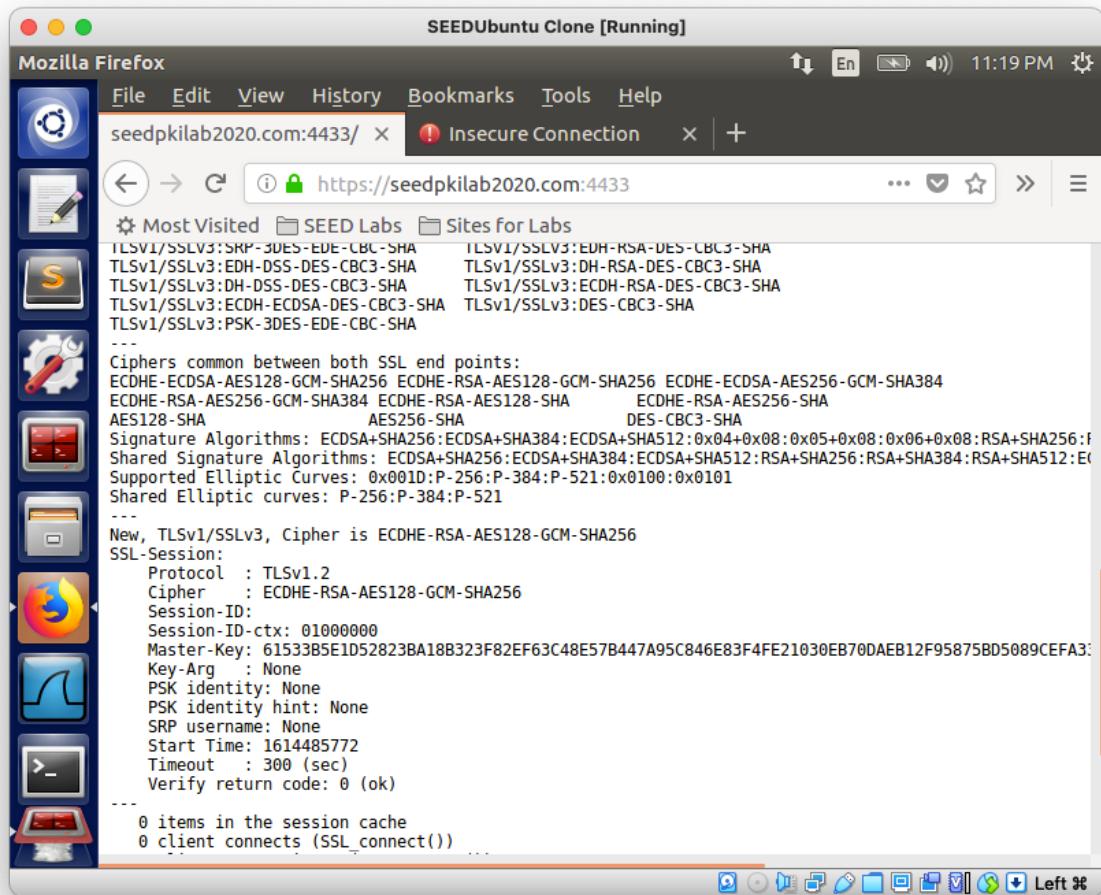
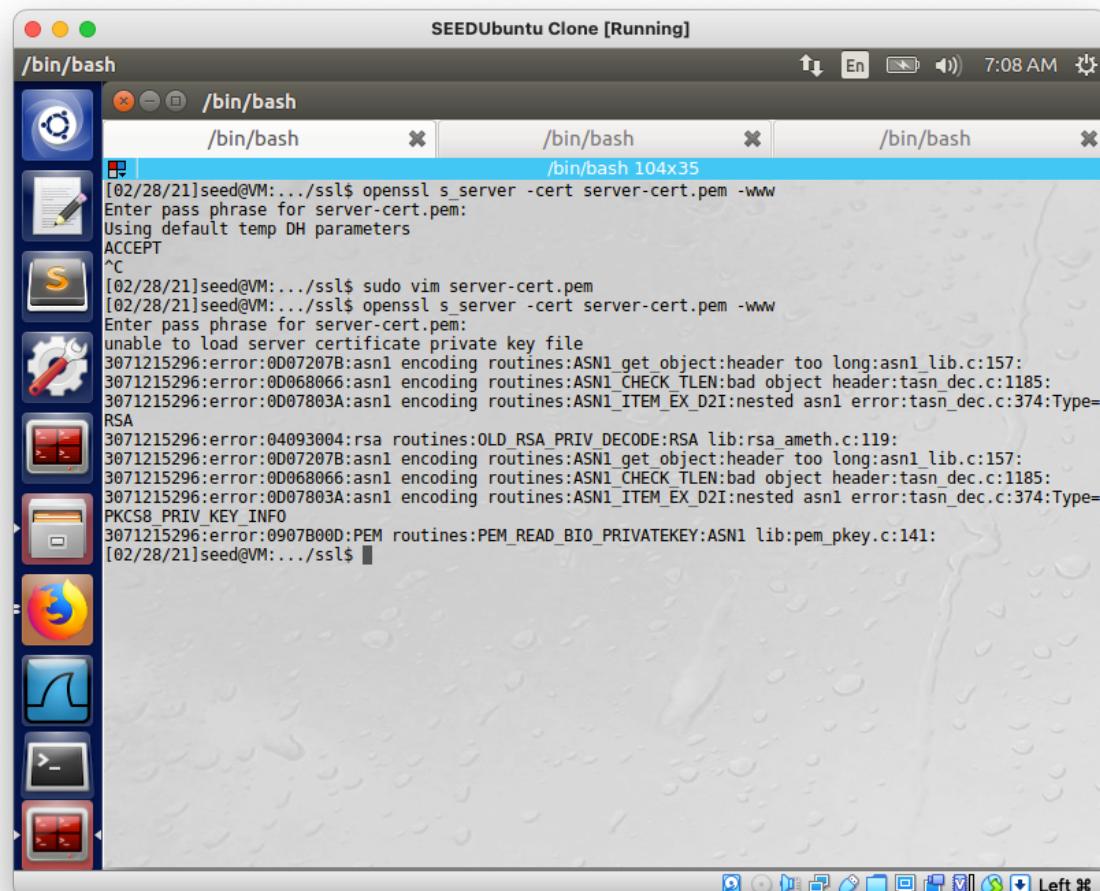


Fig. 3.4.2 Verbose Connection to <https://seedpkilab2020.com>

However, changing a byte in the server key portion in **server-cert.pem** (server.pem) causes openssl to throw errors as shown in Fig. 3.4.3.



The screenshot shows a terminal window titled "SEEDUbuntu Clone [Running]" with three tabs, all labeled "/bin/bash". The central tab is active and displays the following command and its output:

```
[02/28/21]seed@VM:.../ssl$ openssl s_server -cert server-cert.pem -www  
Enter pass phrase for server-cert.pem:  
Using default temp DH parameters  
ACCEPT  
^c  
[02/28/21]seed@VM:.../ssl$ sudo vim server-cert.pem  
[02/28/21]seed@VM:.../ssl$ openssl s_server -cert server-cert.pem -www  
Enter pass phrase for server-cert.pem:  
unable to load server certificate private key file  
3071215296:error:0007207B:asn1 encoding routines:ASN1_get_object:header too long:asn1_lib.c:157:  
3071215296:error:00068066:asn1 encoding routines:ASN1_CHECK_TLEN:bad object header:tasn_dec.c:1185:  
3071215296:error:0007803A:asn1 encoding routines:ASN1_ITEM_EX_D2I:nested asn1 error:tasn_dec.c:374:Type= RSA  
3071215296:error:04093004:rsa routines:OLD_RSA_PRIV_DECODE:RSA lib:rsa_ameth.c:119:  
3071215296:error:0007207B:asn1 encoding routines:ASN1_get_object:header too long:asn1_lib.c:157:  
3071215296:error:00068066:asn1 encoding routines:ASN1_CHECK_TLEN:bad object header:tasn_dec.c:1185:  
3071215296:error:0007803A:asn1 encoding routines:ASN1_ITEM_EX_D2I:nested asn1 error:tasn_dec.c:374:Type= PKCS8_PRIV_KEY_INFO  
3071215296:error:0907B00D:PEM routines:PEM_READ_bio_PRIVATEKEY:ASN1 lib:pem_pkey.c:141:  
[02/28/21]seed@VM:.../ssl$
```

Fig. 3.4.3 Error thrown after modifying a single byte in key inside server.pem

2. Since SEEDPKILab2020.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.

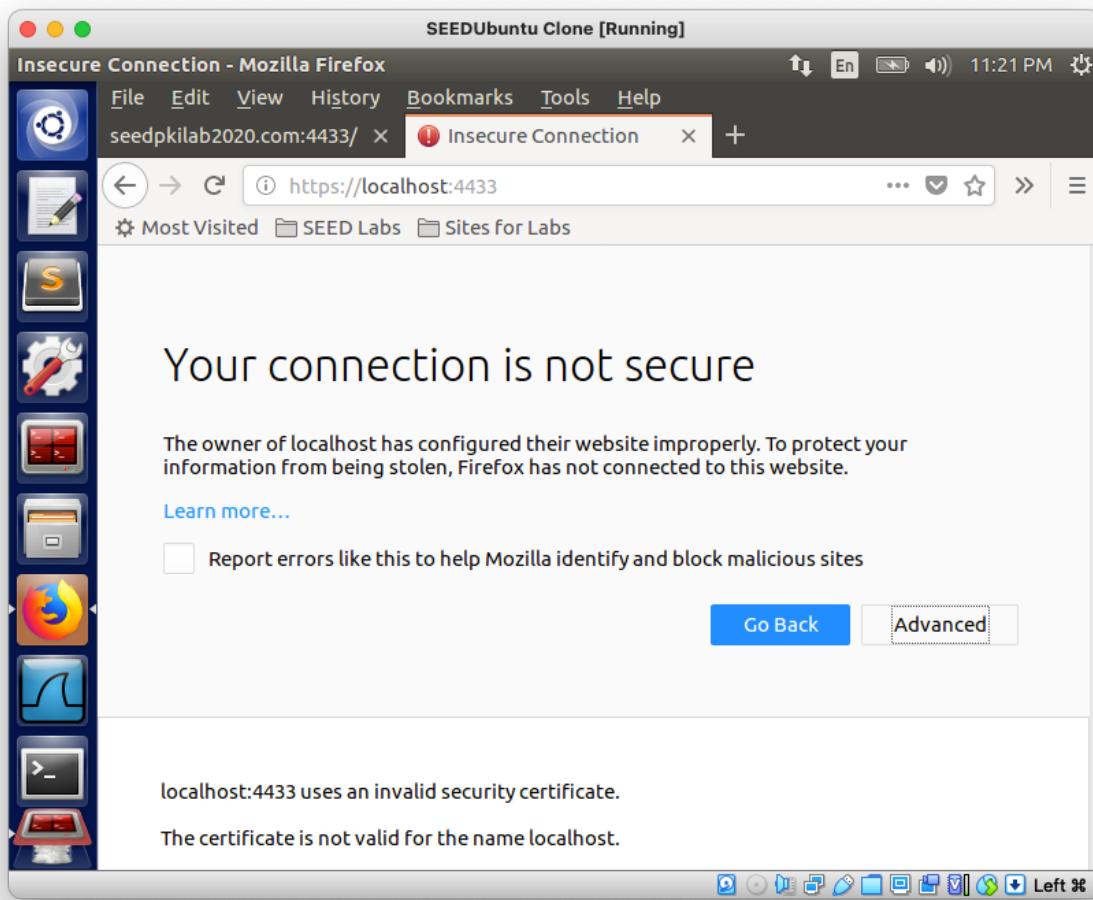


Fig. 3.4.3 Connection to <https://localhost:4433>

This is expected because our certificate is only for **seedpkilab2020.com** (CN) and not **localhost** (hostname). This fails the second validation and hence the browser shows a warning to the user, explicitly asking for permission to trust this website.

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

SEEDUbuntu Clone [Running]

```
# the close notify alert from client. When you need a different shutdown
# approach you can use one of the following variables:
# o ssl-unclean-shutdown:
#     This forces an unclean shutdown when the connection is closed, i.e. no
#     SSL close notify alert is send or allowed to received. This 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 SEEDPKILab2020.com
  DocumentRoot /var/www/seedpki
  DirectoryIndex index.html
  SSLEngine On
  SSLCertificateFile /etc/apache2/ssl/server-cert.pem
  SSLCertificateKeyFile /etc/apache2/ssl/server-key.pem
</VirtualHost>
</IfModule>

vim: syntax=apache ts=4 sw=4 sts=4 sr noet
142,1          Bot
```

Fig. 4.1 **/etc/apache2/sites-available/default-ssl.conf**

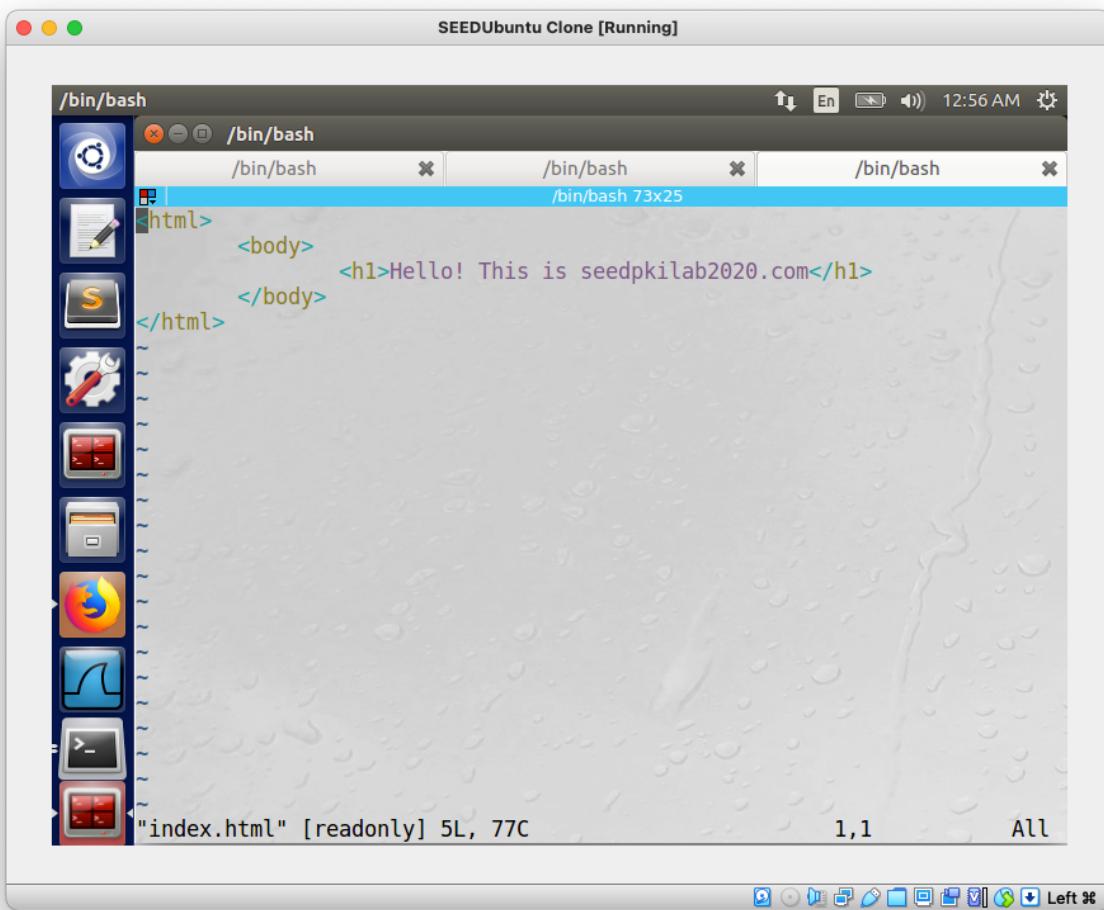
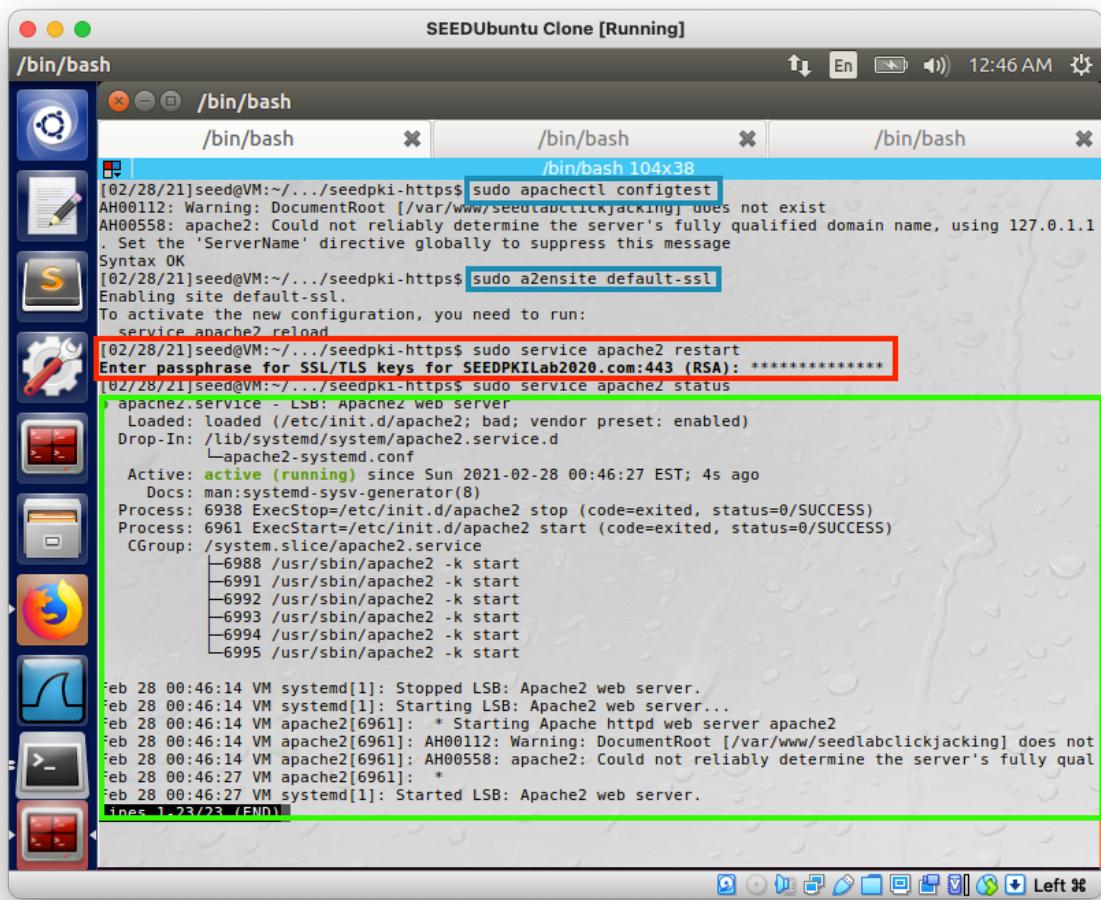


Fig. 4.2 Landing Page of <https://seedpkilab.com>

First, we add ***SEEDPKILab2020.com*** as a VirtualHost in ***/etc/apache2/sites-available/default-ssl.conf*** as seen in Fig. 4.1. Here, we specify the following:

1. All files related to this website to be kept in ***/var/www/seedpki*** (specified by DocumentRoot)
2. ***https://seedpkilab2020.com***'s landing page to be ***index.html*** (DirectoryIndex).
  - a. This is a simple html file as shown in Fig. 4.2 which loads when a client visits the website.
3. Apache will find ***seedpkilab2020***'s certificate and key in ***/etc/apache2/ssl/***
  - a. Server key : ***server-key.pem***
  - b. Server Certificate : ***server-cert.pem***



The screenshot shows a terminal window titled "SEEDUbuntu Clone [Running]" with four tabs labeled "/bin/bash". The terminal is displaying a command-line session:

```
[02/28/21]seed@VM:~/.../seedpki-https$ sudo apache2ctl 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' directive globally to suppress this message
Syntax OK
[02/28/21]seed@VM:~/.../seedpki-https$ sudo a2ensite default-ssl
Enabling site default-ssl.
To activate the new configuration, you need to run:
service apache2 reload
[02/28/21]seed@VM:~/.../seedpki-https$ sudo service apache2 restart
Enter passphrase for SSL/TLS keys for SEEDPKILab2020.com:443 (RSA): *****
[02/28/21]seed@VM:~/.../seedpki-https$ sudo service apache2 status
apache2.service - LSB: Apache2 Web server
   Loaded: loaded (/etc/init.d/apache2; bad; vendor preset: enabled)
   Drop-In: /lib/systemd/system/apache2.service.d
             └─apache2-systemd.conf
     Active: active (running) since Sun 2021-02-28 00:46:27 EST; 4s ago
       Docs: man:systemd-sysv-generator(8)
   Process: 6938 ExecStop=/etc/init.d/apache2 stop (code=exited, status=0/SUCCESS)
   Process: 6961 ExecStart=/etc/init.d/apache2 start (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/apache2.service
           ├─6988 /usr/sbin/apache2 -k start
           ├─6991 /usr/sbin/apache2 -k start
           ├─6992 /usr/sbin/apache2 -k start
           ├─6993 /usr/sbin/apache2 -k start
           ├─6994 /usr/sbin/apache2 -k start
           └─6995 /usr/sbin/apache2 -k start

Feb 28 00:46:14 VM systemd[1]: Stopped LSB: Apache2 web server.
Feb 28 00:46:14 VM systemd[1]: Starting LSB: Apache2 web server...
Feb 28 00:46:14 VM apache2[6961]: * Starting Apache httpd web server apache2
Feb 28 00:46:14 VM apache2[6961]: AH00112: Warning: DocumentRoot [/var/www/seedlabclickjacking] does not
Feb 28 00:46:14 VM apache2[6961]: AH00558: apache2: Could not reliably determine the server's fully qual
Feb 28 00:46:27 VM apache2[6961]: *
Feb 28 00:46:27 VM systemd[1]: Started LSB: Apache2 web server.
lines 1-23/23 (END)
```

A red box highlights the command "Enter passphrase for SSL/TLS keys for SEEDPKILab2020.com:443 (RSA): \*\*\*\*\*". A green box highlights the entire output of the "sudo service apache2 status" command.

Fig. 4.1 Commands Used to restart Apache Server

Now that we have properly configured the necessary files for <https://seedpkilab2020.com>. We will follow the following steps to configure **apache2** itself. The four steps are shown in Fig. 4.2. We are sure that **apache2** is able to find the appropriate files when I was prompted to input the passphrase for SSL keys for the website (highlighted in red box). After restarting **apache2** service, we verify that there are no errors by checking its status with the command **sudo service apache2 status**. We further confirm we have successfully deployed the certificate in Apache-Based HTTPS Website by visiting the site itself as shown in Fig. 4.2 below. As expected, it loads the **index.html** that we created.

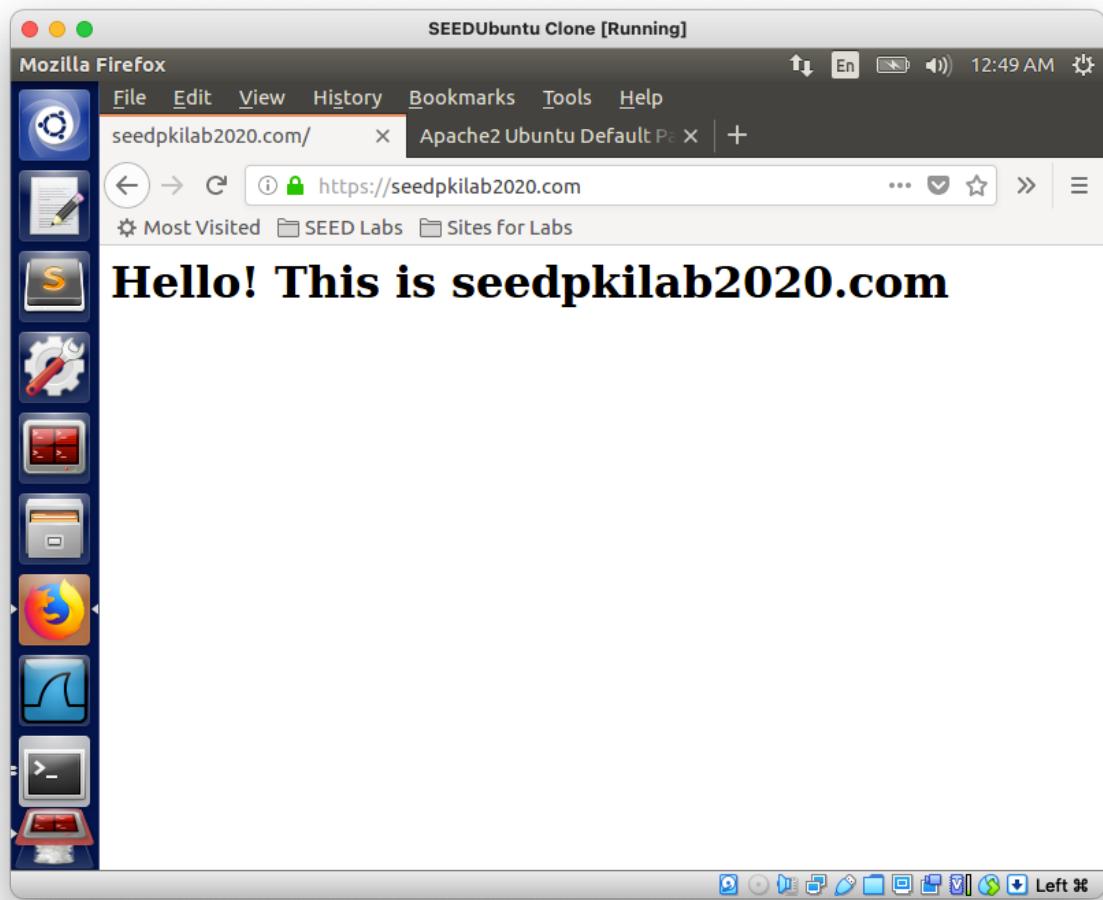
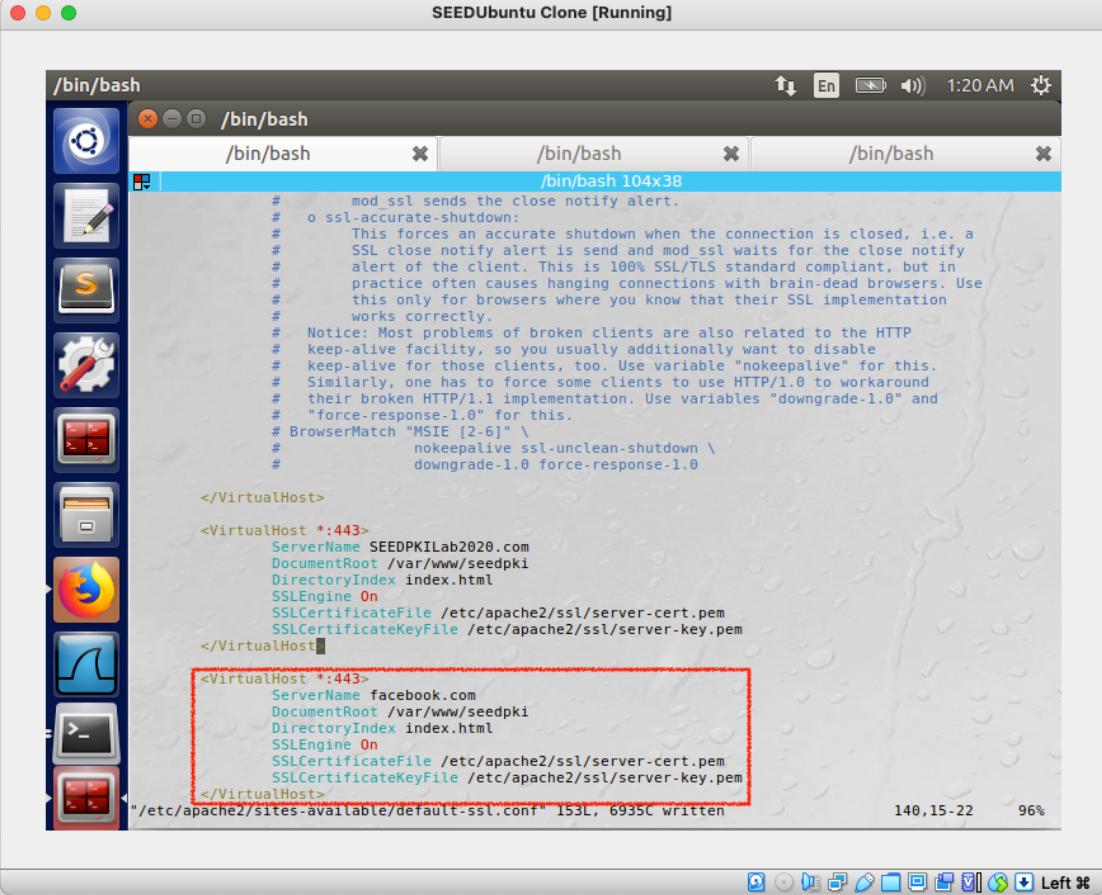


Fig. 4.2 *index.html* of seedpkilab2020.com

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

## Step 1: Setting up the malicious website.

I will choose to impersonate **facebook.com** and as instructed, I first add the corresponding VirtualHost code block into **/etc/apache2/sites-available/default-ssl.conf** as shown in Fig 5.1 below.



The screenshot shows a terminal window titled "SEEDUbuntu Clone [Running]" with three tabs open, all showing the same bash prompt: "/bin/bash". The terminal is displaying Apache configuration code. A specific section of the code is highlighted with a red rectangular box:

```
# 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 SEEDPKILab2020.com
    DocumentRoot /var/www/seedpki
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /etc/apache2/ssl/server-cert.pem
    SSLCertificateKeyFile /etc/apache2/ssl/server-key.pem
</VirtualHost>

<VirtualHost *:443>
    ServerName facebook.com
    DocumentRoot /var/www/seedpki
    DirectoryIndex index.html
    SSLEngine On
    SSLCertificateFile /etc/apache2/ssl/server-cert.pem
    SSLCertificateKeyFile /etc/apache2/ssl/server-key.pem
</VirtualHost>
```

The line "ServerName facebook.com" is part of the highlighted configuration. The bottom status bar of the terminal window indicates the file being edited is "/etc/apache2/sites-available/default-ssl.conf", with 153L and 6935C written.

Fig. 5.1 Addition of VirtualHost (facebook.com)

The screenshot shows a desktop environment with a window titled "SEEDUbuntu Clone [Running]". Inside, a terminal window titled "/bin/bash" displays the following log output:

```
[02/28/21]seed@VM:~/.seedpki-https$ sudo vim /etc/apache2/sites-available/default-ssl.conf
[02/28/21]seed@VM:~/.seedpki-https$ 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' directive globally to suppress this message
Syntax OK
[02/28/21]seed@VM:~/.seedpki-https$ 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
[02/28/21]seed@VM:~/.seedpki-https$ sudo a2ensite default-ssl
Site default-ssl already enabled
[02/28/21]seed@VM:~/.seedpki-https$ sudo service apache2 restart
Enter passphrase for SSL/TLS keys for facebook.com:443 (RSA): *****
[02/28/21]seed@VM:~/.seedpki-https$ sudo service apache2 status
● apache2.service - LSB: Apache2 web server
   Loaded: loaded (/etc/init.d/apache2; bad; vendor preset: enabled)
   Drop-In: /lib/systemd/system/apache2.service.d
             └─apache2-systemd.conf
     Active: active (running) since Sun 2021-02-28 01:23:53 EST; 5s ago
       Docs: man:systemd-sysv-generator(8)
   Process: 7407 ExecStop=/etc/init.d/apache2 stop (code=exited, status=0/SUCCESS)
   Process: 7429 ExecStart=/etc/init.d/apache2 start (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/apache2.service
           ├─7450 /usr/sbin/apache2 -k start
           ├─7453 /usr/sbin/apache2 -k start
           ├─7454 /usr/sbin/apache2 -k start
           ├─7455 /usr/sbin/apache2 -k start
           ├─7456 /usr/sbin/apache2 -k start
           └─7457 /usr/sbin/apache2 -k start
Feb 28 01:23:46 VM systemd[1]: Stopped LSB: Apache2 web server.
Feb 28 01:23:46 VM systemd[1]: Starting LSB: Apache2 web server...
```

Fig. 5.2 Restarting apache2

Next, we add the IP-hostname mapping to **/etc/hosts** with the following line **127.0.0.1 facebook.com** as depicted in Fig. 5.5 (last image). Figure 5.3 shows the attempt to visit **https://facebook.com** on **apache2** on the attacker VM itself and Figure 5.4 shows the attempt to visit **facebook.com**.

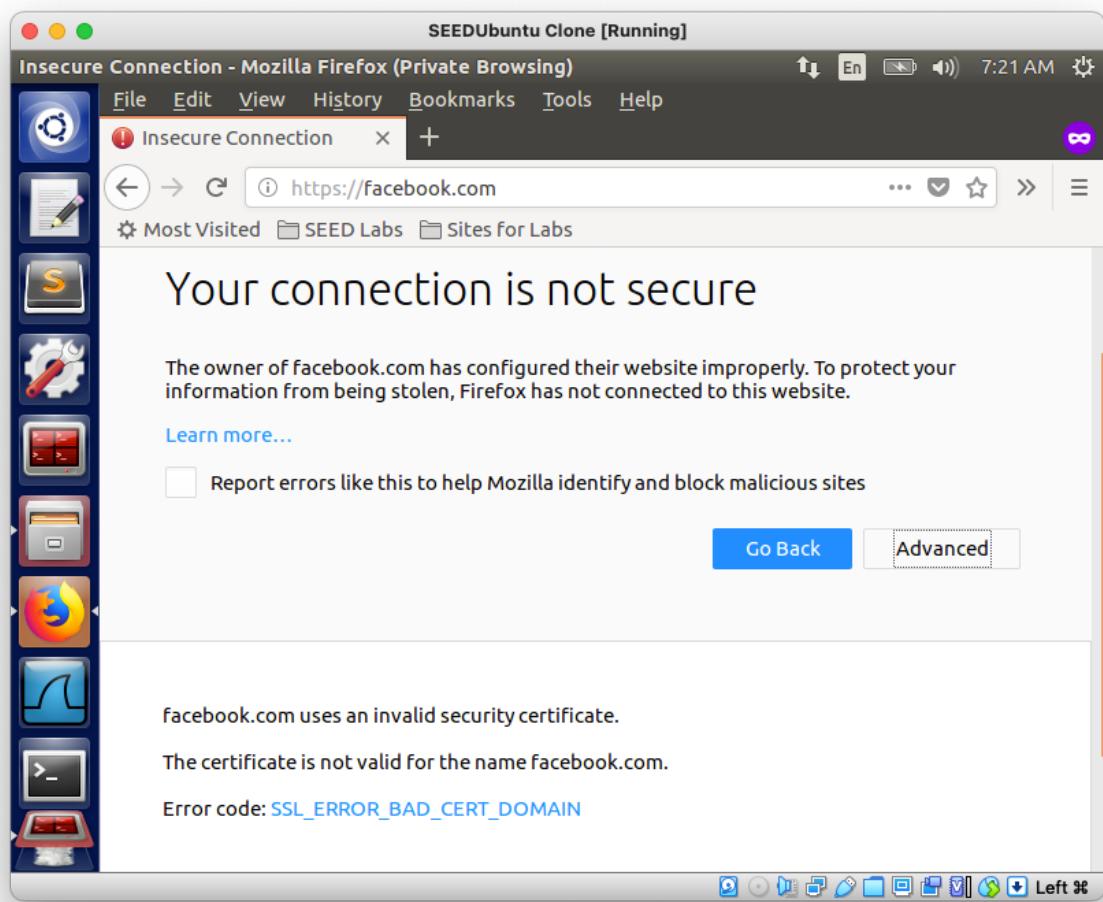


Fig. 5.3 Attempt to visit <https://facebook.com>

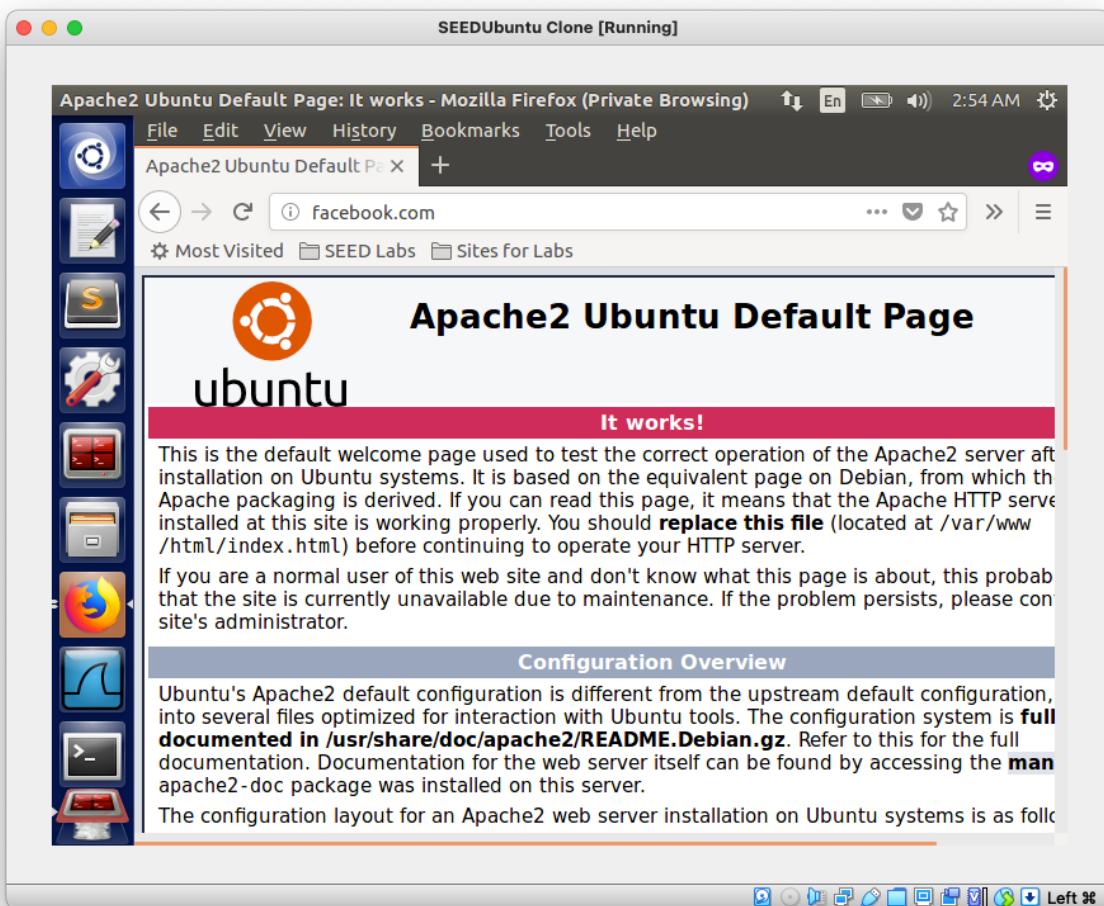


Fig. 5.4 Attempt to visit facebook.com

SEEDUbuntu Clone [Running]

/bin/bash

/bin/bash

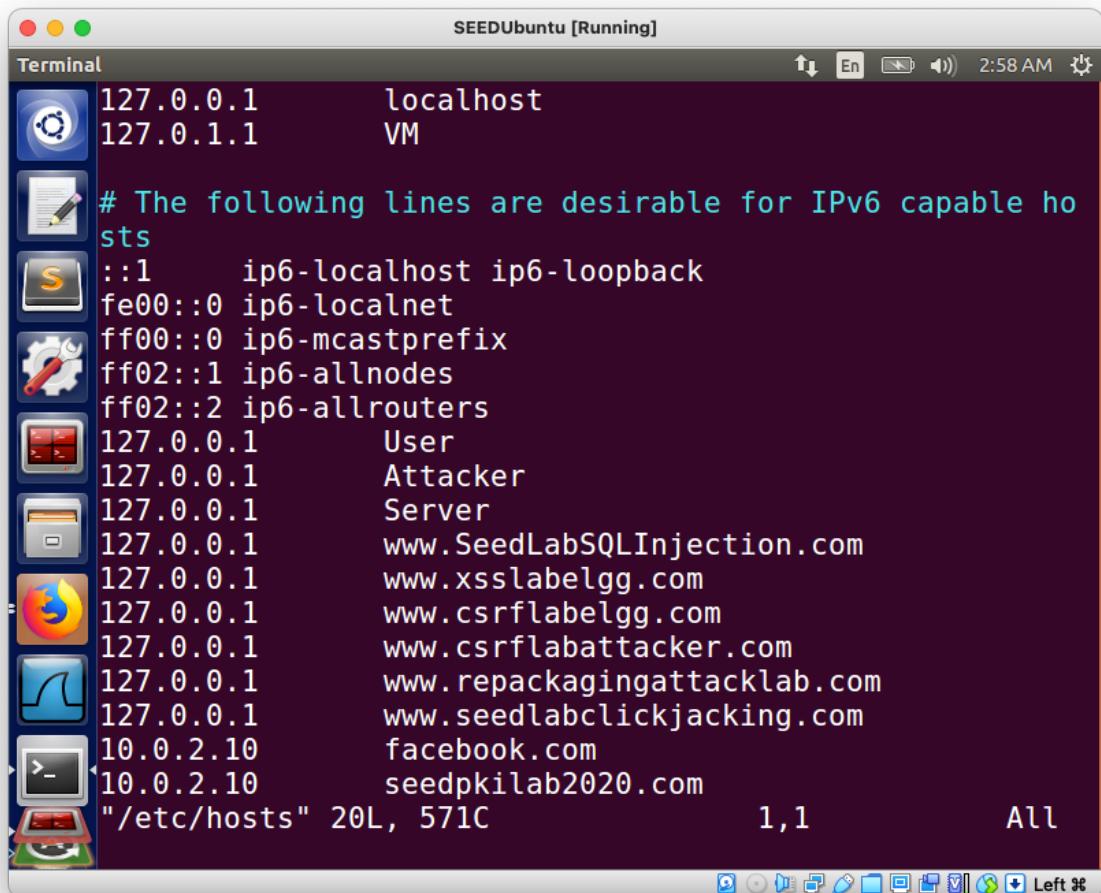
/bin/bash

/bin/bash 104x38

```
Zs0isoC6uhkCfD6qJiHHUbk9EGR0YDCHqLJP6700t9MGbm7daXciYbI0LhuQ/LvV  
vCYqzQ33XTVnw4wPUTAK/BDWphv48U0F0zIWafcfqxS/Ctxx+fBebhWRVr7CzqX  
VzRADKUszks/GmxgmM7Gxa3xWZxoy3k9k+kqYajyersFHYZ  
----END CERTIFICATE----  
[02/28/21]seed@VM:~/.../facebook$ date  
Sun Feb 28 02:25:17 EST 2021  
[02/28/21]seed@VM:~/.../facebook$ sudo vim /etc/apache2/sites-available/default-ssl.conf  
[02/28/21]seed@VM:~/.../facebook$ sudo vim /etc/hosts  
[02/28/21]seed@VM:~/.../facebook$ sudo service apache2 restart  
Failed to restart apache2.service: Unit apache2.service not found.  
[02/28/21]seed@VM:~/.../facebook$ sudo service apache2 restart  
Enter passphrase for SSL/TLS keys for instagram.com:443 (RSA): *****  
[02/28/21]seed@VM:~/.../facebook$ sudo vim /etc/apache2/sites-available/default-ssl.conf  
[02/28/21]seed@VM:~/.../facebook$ sudo vim /etc/hosts  
[02/28/21]seed@VM:~/.../facebook$ sudo service apache2 restart  
Enter passphrase for SSL/TLS keys for facebook.com:443 (RSA): *****  
[02/28/21]seed@VM:~/.../facebook$ cat /etc/hosts  
127.0.0.1      localhost  
127.0.1.1      VM  
  
# The following lines are desirable for IPv6 capable hosts  
::1      ip6-localhost ip6-loopback  
fe00::0 ip6-localnet  
ff00::0 ip6-mcastprefix  
ff02::1 ip6-allnodes  
ff02::2 ip6-allrouters  
127.0.0.1      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.repackagingattacklab.com  
127.0.0.1      www.seedlabclickjacking.com  
127.0.0.1      SEENDPktLab2020.com  
127.0.0.1      facebook.com  
[02/28/21]seed@VM:~/.../facebook$
```

Fig. 5.5 Addition to `/etc/hosts`

## Step 2: Becoming the man in the middle



The screenshot shows a terminal window titled "SEEDUbuntu [Running]" with the command "cat /etc/hosts" run. The output displays the contents of the /etc/hosts file, which maps various IP addresses to domain names. The file includes entries for localhost, VM, and several malicious websites.

```
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
10.0.2.10      facebook.com
10.0.2.10      seedpkilab2020.com

"/etc/hosts" 20L, 571C          1,1          All
```

Fig 5.6 **/etc/hosts** file on victim VM

On the victim VM, we map the attacker IP (**10.0.2.10**) to the malicious site **facebook.com**.

### Step 3: Browse the target website.

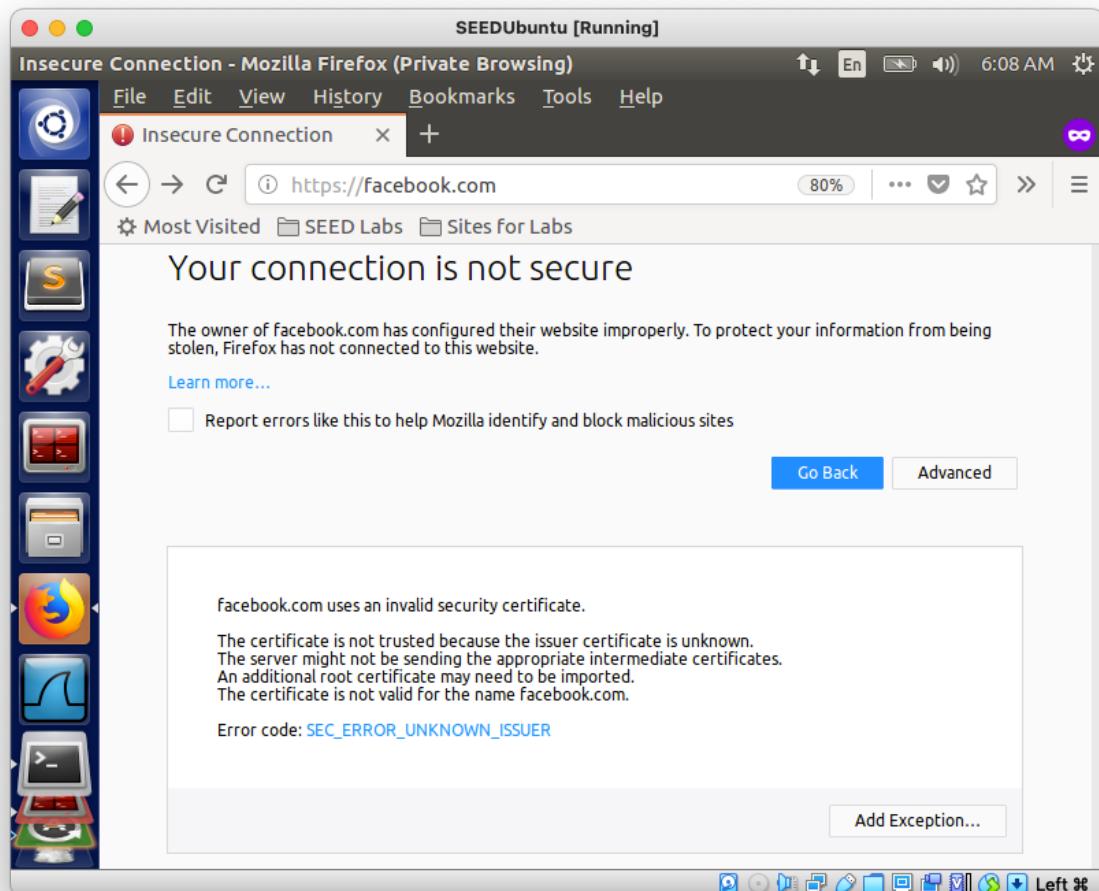


Fig. 5.7 Victim VM's attempt to connect to facebook.com

This is expected because the subject on the Certificate (sent by the attacker) is for seedpkilab2020.com instead of facebook. This explains the error ***The certificate is not valid for the name facebook.com***. The first 3 warnings are due to the attacker certificate's CA not being inside the victim browser's list of trusted CAs.

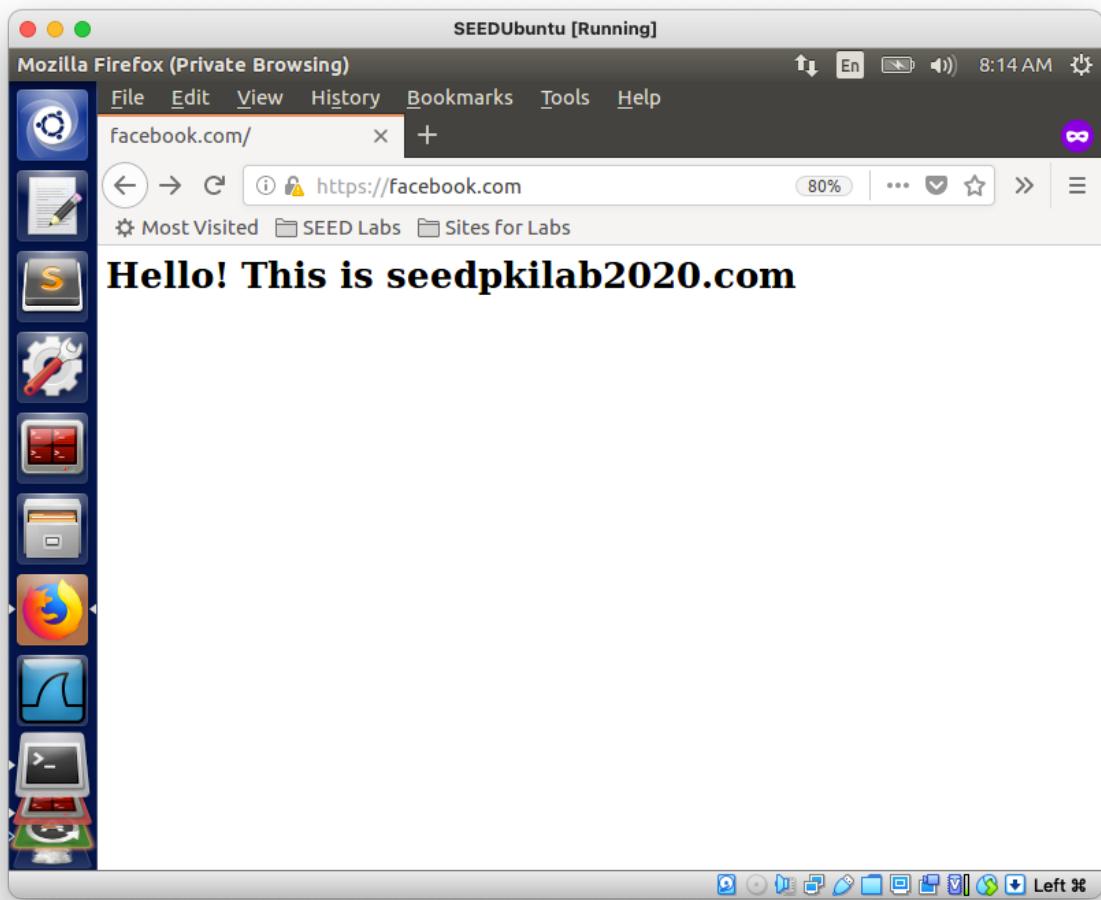


Fig 5.8 Victim ignores warning

Fig 5.8 shows the victim rerouted to the malicious site after ignoring the warning.

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

Let us first understand why the MITM (emulating a DNS attack) in task 5 fails:

1. Common Name in the certificate (seedpkilab2020) is not the same as the malicious site (facebook.com)
2. The CA that signed this malicious site's certificate is not in the list of trusted CAs.

In this task, we assume the root CA has been compromised by an attacker, and its private key is stolen. I will follow the steps detailed in Task 5 with several changes:

1. Generate **facebook.com** CSR with the common name explicitly set to **facebook.com**
  - a. Refer to Fig. 6.1's cyan highlighted box
2. Signing **facebook.com** CSR with CA key as shown in Fig 6.2

We then visit <https://facebook.com> and we are successfully rerouted to the malicious **facebook.com**. Since we did not change **index.html** used in task 5, it is expected for us to see the landing page message as **Hello! This is seedpkilab2020.com** as shown in Fig. 6.4. This works because the CA is in the machine's trusted list of CA, thus suppressing the warning.

```
[02/28/21]seed@VM:~/.../lab4$ ls
ca.crt  demoCA    facebook.csr  openssl.cnf  server.crt  server.key
ca.key   facebook.crt  facebook.key  seedpki-https  server.csr  server.pem
[02/28/21]seed@VM:~/.../lab4$ openssl req -new -key facebook.key -out facebook.csr -config openssl.cnf
Enter pass phrase for facebook.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:SG
State or Province Name (full name) [Some-State]::
Locality Name (eg, city) []:Singapore
Organization Name (eg, company) [Internet Widgits Pty Ltd]:facebook^C
[02/28/21]seed@VM:~/.../lab4$ openssl req -new -key facebook.key -out facebook.csr -config openssl.cnf
Enter pass phrase for facebook.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:California
Locality Name (eg, city) []:San Diego
Organization Name (eg, company) [Internet Widgits Pty Ltd]:FB
Organizational Unit Name (eg, section) []:Cyber
Common Name (e.g. server FQDN or YOUR name) []:facebook.com
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
[02/28/21]seed@VM:~/.../lab4$
```

Fig 6.1 Creating \*facebook.com\* CSR

SEEDUbuntu Clone [Running]

/bin/bash

/bin/bash

/bin/bash

/bin/bash

/bin/bash 104x38

```
ca.crt demoCA facebook.csr openssl.cnf server.crt server.key
ca.key facebook.crt facebook.key seedpki-https server.csr server.pem
[02/28/21]seed@VM:~/.../lab4$ openssl ca -in facebook.csr -out facebook.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: 4100 (0x1004)
    Validity
        Not Before: Feb 28 12:48:08 2021 GMT
        Not After : Feb 28 12:48:08 2022 GMT
    Subject:
        countryName             = US
        stateOrProvinceName     = California
        localityName            = San Diego
        organizationName         = FB
        organizationalUnitName   = Cyber
        commonName               = facebook.com
X509v3 extensions:
    X509v3 Basic Constraints:
        CA:FALSE
    Netscape Comment:
        OpenSSL Generated Certificate
    X509v3 Subject Key Identifier:
        AE:39:C7:A2:8C:75:3C:1F:A4:A2:5E:30:C2:3C:D2:41:FB:29:E2:97
    X509v3 Authority Key Identifier:
        keyid:56:08:3B:C5:F1:71:1C:5C:91:6F:01:CA:F4:0E:4D:7F:60:0A:5F:97
Certificate is to be certified until Feb 28 12:48:08 2022 GMT (365 days)
Sign the certificate? [y/n]:y
1 out of 1 certificate requests certified, commit? [y/n]
Write out database with 1 new entries
Data Base Updated
[02/28/21]seed@VM:~/.../lab4$
```

Fig 6.3 Signing of \*facebook.com\* CSR

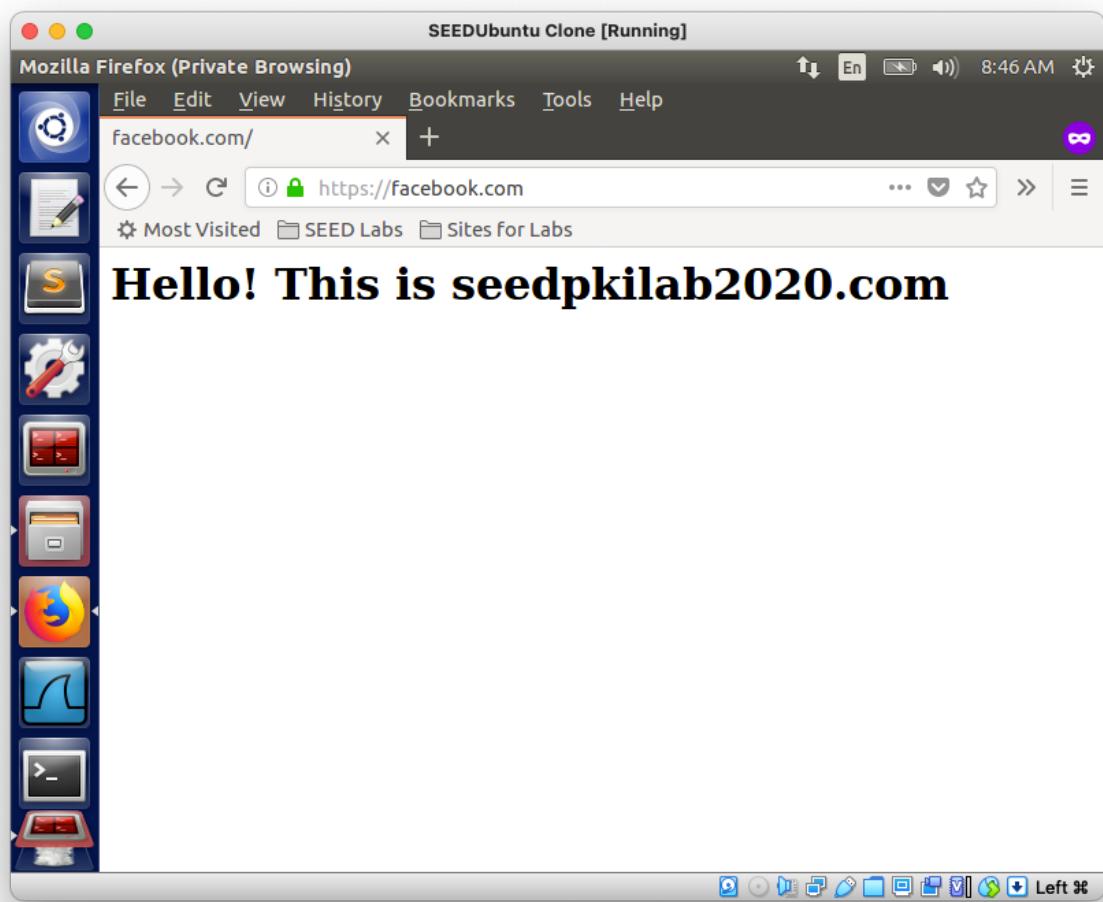


Fig 6.4 Malicious facebook.com