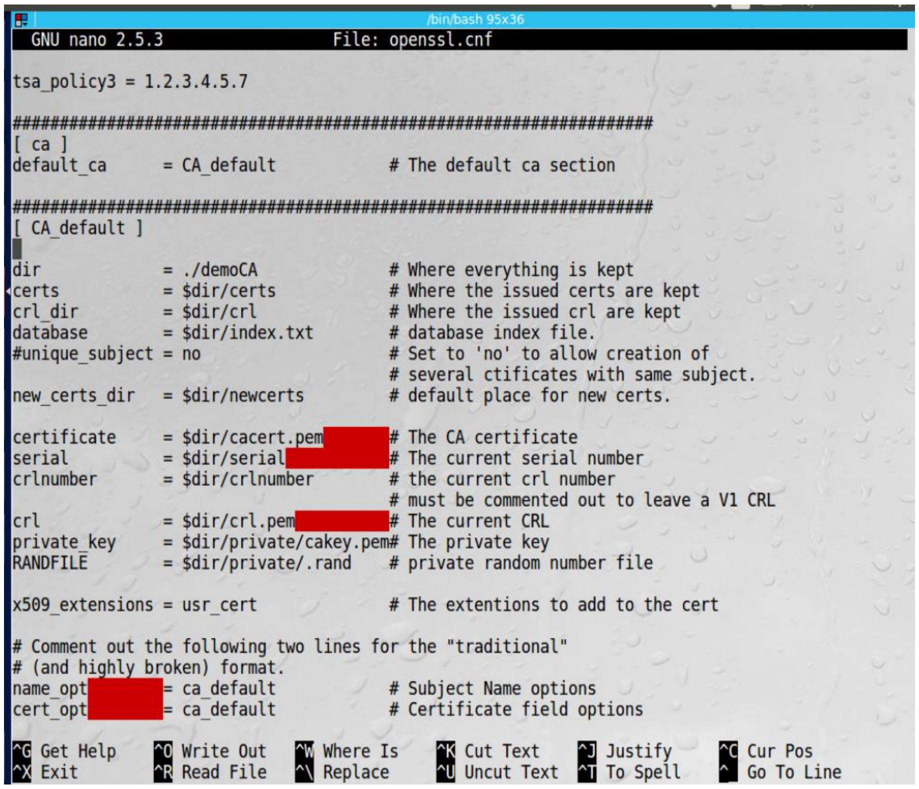
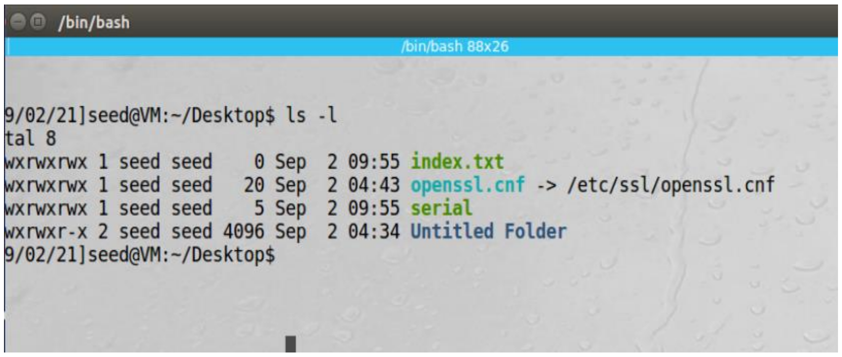
Task 1: Becoming a Certificate Authority (CA)

The first thing we do is copy the conf file template over to the working directory /Desktop, and edit the conf file to show the base directory:



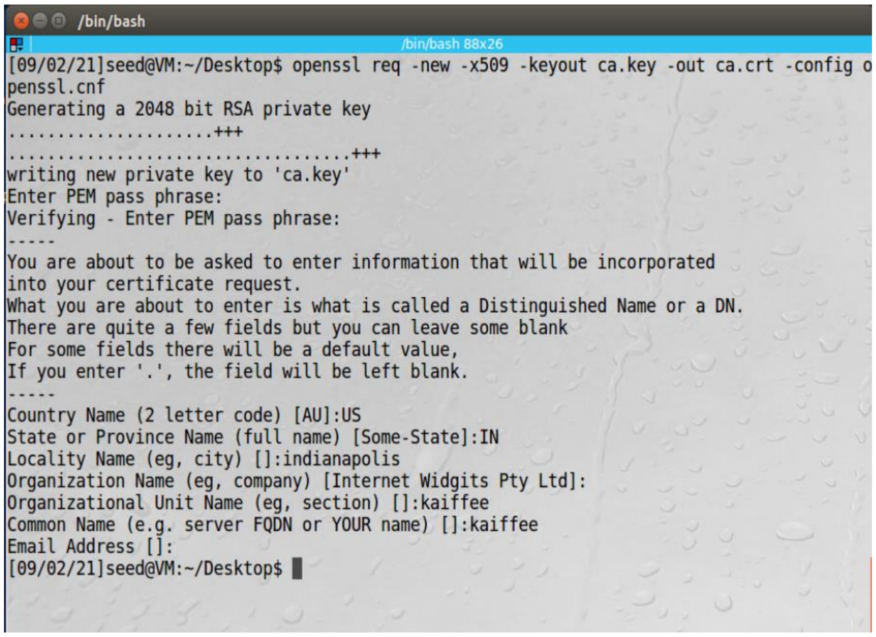
```
/bin/bash 95x36
  GNU nano 2.5.3                   File: openssl.cnf

tsa_policy3 = 1.2.3.4.5.7

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

####################################################################
[ 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 ctificates with same subject.
new_certs_dir   = $dir/newcerts         # default place for new certs.

certificate     = $dir/cacert.pem[redacted]# 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[redacted]# 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[redacted]= ca_default          # Subject Name options
cert_opt        = ca_default            # Certificate field options

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line
```

We then add the directories that we need, as well as the two files: index.txt and serial:



```
/bin/bash
/bin/bash 88x26

9/02/21]seed@VM:~/Desktop$ ls -l
tal 8
wxrwxrwx 1 seed seed    0 Sep  2 09:55 index.txt
wxrwxrwx 1 seed seed   20 Sep  2 04:43 openssl.cnf -> /etc/ssl/openssl.cnf
wxrwxrwx 1 seed seed    5 Sep  2 09:55 serial
wxrwxr-x 2 seed seed 4096 Sep  2 04:34 Untitled Folder
9/02/21]seed@VM:~/Desktop$
```

Next we generate the self-signed certificate and enter the relevant information, the passphrase is "deesdees":



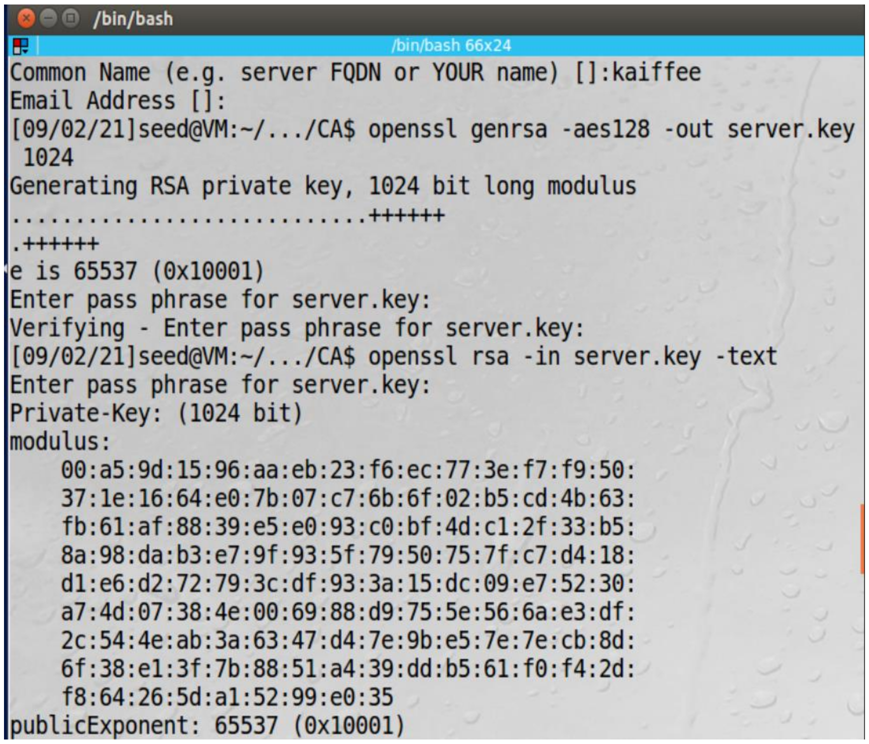
```
[09/02/21]seed@VM:~/Desktop$ openssl req -new -x509 -keyout ca.key -out ca.crt -config o
penssl.cnf
Generating a 2048 bit RSA private key
.....................+++
..................................+++
writing new private key to 'ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:IN
Locality Name (eg, city) []:indianapolis
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:kaiffee
Common Name (e.g. server FQDN or YOUR name) []:kaiffee
Email Address []:
[09/02/21]seed@VM:~/Desktop$ 
```

Observation: In this task we had to create the process to become a Certificate Authority. We first copying the template configuration file over into our working directory, then we created the necessary subfolders and files to support the CA. We then needed to create a self-signed certificate for the CA to use. This was acomplisthed using the openssl req command. The password used for this, and all other certs for this lab was "deesdees".

Explanation: A certificate is required part of the PKI encryption scheme. The CA is a trusted entity that verifies the authenticity of the user/server certificates. We didn't want to pay a commercial provider, like verisign, to act as our CA so we created our own using root.

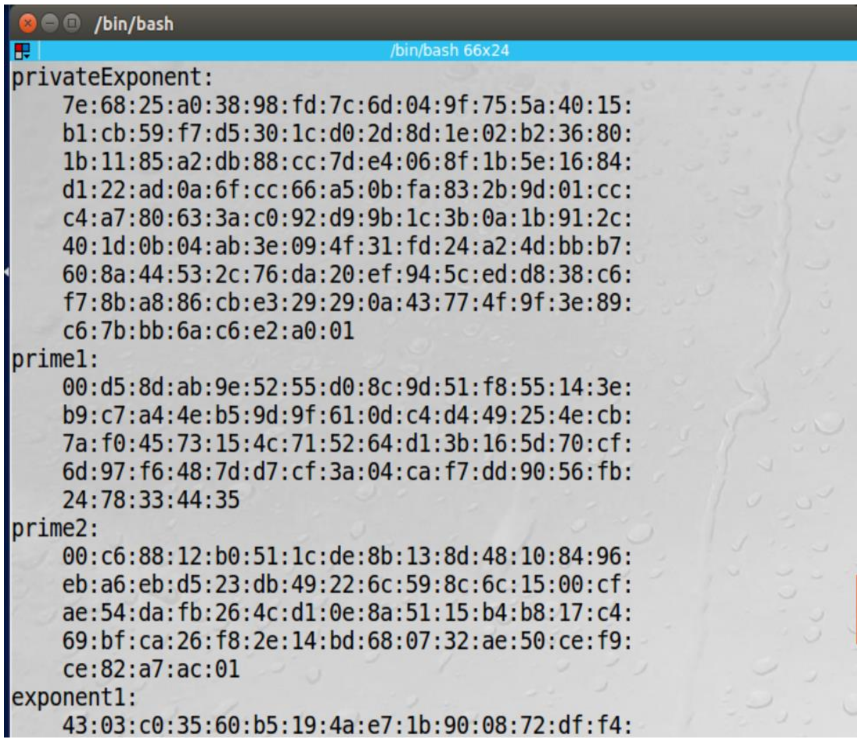
Task 2: Creating a Certificate for SEEDPKILab2020.com

Next we create the RSA public/private key pair, again we are using "deesdees" for the passphrase:

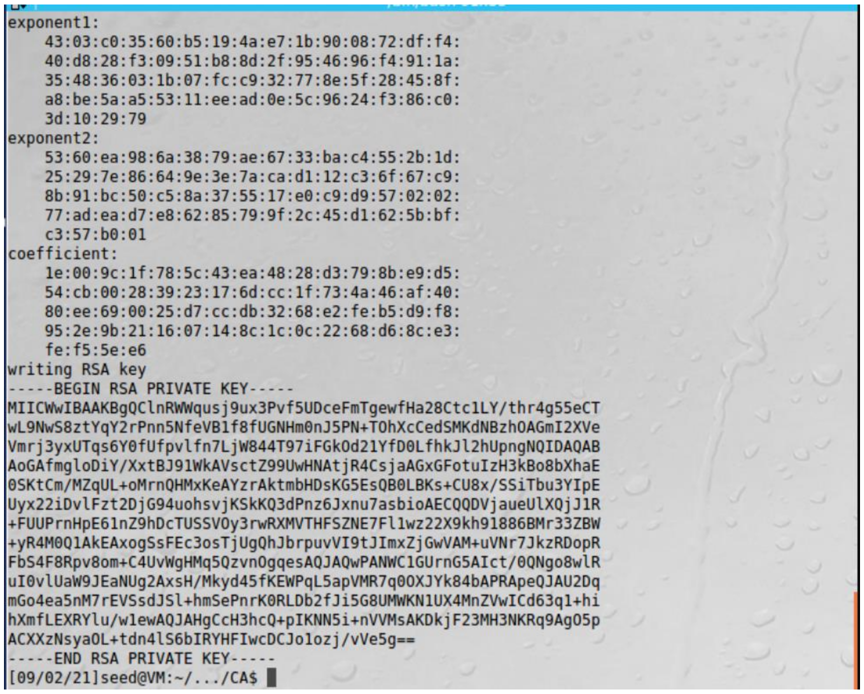


```
Common Name (e.g. server FQDN or YOUR name) []:kaiffee
Email Address []:
[09/02/21]seed@VM:~/.../CA$ openssl genrsa -aes128 -out server.key
 1024
Generating RSA private key, 1024 bit long modulus
...........................++++++
.++++++
e is 65537 (0x10001)
Enter pass phrase for server.key:
Verifying - Enter pass phrase for server.key:
[09/02/21]seed@VM:~/.../CA$ openssl rsa -in server.key -text
Enter pass phrase for server.key:
Private-Key: (1024 bit)
modulus:
    00:a5:9d:15:96:aa:eb:23:f6:ec:77:3e:f7:f9:50:
    37:1e:16:64:e0:7b:07:c7:6b:6f:02:b5:cd:4b:63:
    fb:61:af:88:39:e5:e0:93:c0:bf:4d:c1:2f:33:b5:
    8a:98:da:b3:e7:9f:93:5f:79:50:75:7f:c7:d4:18:
    d1:e6:d2:72:79:3c:df:93:3a:15:dc:09:e7:52:30:
    a7:4d:07:38:4e:00:69:88:d9:75:5e:56:6a:e3:df:
    2c:54:4e:ab:3a:63:47:d4:7e:9b:e5:7e:7e:cb:8d:
    6f:38:e1:3f:7b:88:51:a4:39:dd:b5:61:f0:f4:2d:
    f8:64:26:5d:a1:52:99:e0:35
publicExponent: 65537 (0x10001)
```

And the rest of the key:

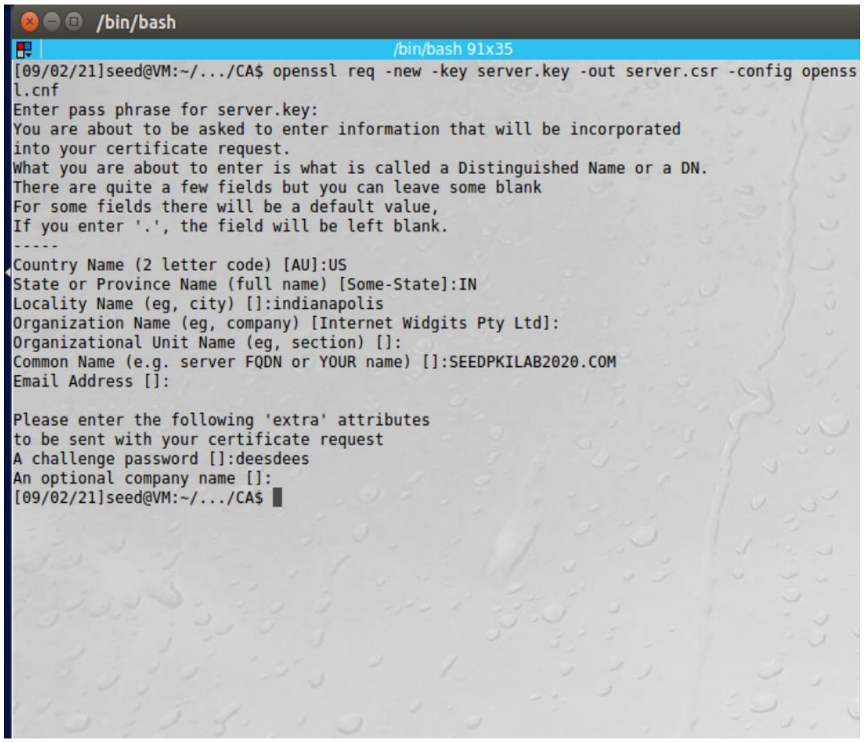


```
privateExponent:
    7e:68:25:a0:38:98:fd:7c:6d:04:9f:75:5a:40:15:
    b1:cb:59:f7:d5:30:1c:d0:2d:8d:1e:02:b2:36:80:
    1b:11:85:a2:db:88:cc:7d:e4:06:8f:1b:5e:16:84:
    d1:22:ad:0a:6f:cc:66:a5:0b:fa:83:2b:9d:01:cc:
    c4:a7:80:63:3a:c0:92:d9:9b:1c:3b:0a:1b:91:2c:
    40:1d:0b:04:ab:3e:09:4f:31:fd:24:a2:4d:bb:b7:
    60:8a:44:53:2c:76:da:20:ef:94:5c:ed:d8:38:c6:
    f7:8b:a8:86:cb:e3:29:29:0a:43:77:4f:9f:3e:89:
    c6:7b:bb:6a:c6:e2:a0:01
prime1:
    00:d5:8d:ab:9e:52:55:d0:8c:9d:51:f8:55:14:3e:
    b9:c7:a4:4e:b5:9d:9f:61:0d:c4:d4:49:25:4e:cb:
    7a:f0:45:73:15:4c:71:52:64:d1:3b:16:5d:70:cf:
    6d:97:f6:48:7d:d7:cf:3a:04:ca:f7:dd:90:56:fb:
    24:78:33:44:35
prime2:
    00:c6:88:12:b0:51:1c:de:8b:13:8d:48:10:84:96:
    eb:a6:eb:d5:23:db:49:22:6c:59:8c:6c:15:00:cf:
    ae:54:da:fb:26:4c:d1:0e:8a:51:15:b4:b8:17:c4:
    69:bf:ca:26:f8:2e:14:bd:68:07:32:ae:50:ce:f9:
    ce:82:a7:ac:01
exponent1:
    43:03:c0:35:60:b5:19:4a:e7:1b:90:08:72:df:f4:
```



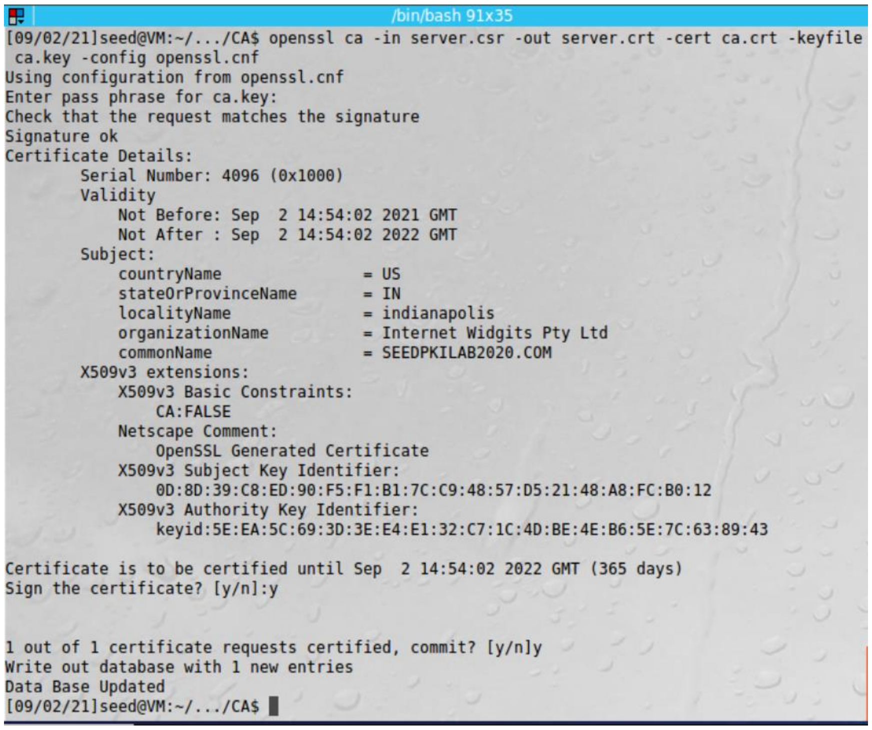
```
exponent1:
    43:03:c0:35:60:b5:19:4a:e7:1b:90:08:72:df:f4:
    40:d8:28:f3:09:51:b8:8d:2f:95:46:96:f4:91:1a:
    35:48:36:03:1b:07:fc:c9:32:77:8e:5f:28:45:8f:
    a8:be:5a:a5:53:11:ee:ad:0e:5c:96:24:f3:86:c0:
    3d:10:29:79
exponent2:
    53:60:ea:98:6a:38:79:ae:67:33:ba:c4:55:2b:1d:
    25:29:7e:86:64:9e:3e:7a:ca:d1:12:c3:6f:67:c9:
    8b:91:bc:50:c5:8a:37:55:17:e0:c9:d9:57:02:02:
    77:ad:ea:d7:e8:62:85:79:9f:2c:45:d1:62:5b:bf:
    c3:57:b0:01
coefficient:
    1e:00:9c:1f:78:5c:43:ea:48:28:d3:79:8b:e9:d5:
    54:cb:00:28:39:23:17:6d:cc:1f:73:4a:46:af:40:
    80:ee:69:00:25:d7:cc:db:32:68:e2:fe:b5:d9:f8:
    95:2e:9b:21:16:07:14:8c:1c:0c:22:68:d6:8c:e3:
    fe:f5:5e:e6
writing RSA key
-----BEGIN RSA PRIVATE KEY-----
MIICWwIBAAKBgQClnRWWqusj9ux3Pvf5UDceFmTgewfHa28Ctc1LY/thr4g55eCT
wL9NwS8ztYqY2rPnn5NfeVB1f8fUGNHm0nJ5PN+TOhXcCedSMKdNBzhOAGmI2XVe
Vmrj3yxUTqs6Y0fUfpvlfn7LjW844T97iFGkOd21YfD0LfhkJl2hUpngNQIDAQAB
AoGAfmgloDiY/XxtBJ91WkAVsctZ99UwHNAtjR4CsjaAGxGFotuIzH3kBo8bXhaE
0SKtCm/MZqUL+oMrnQHMxKeAYzrAktmbHDsKG5EsQB0LBKs+CU8x/SSiTbu3YIpE
Uyx22iDvlFzt2DjG94uohsvjKSkKQ3dPnz6Jxnu7asbioAECQQDVjaueUlXQjJ1R
+FUUPrnHpE61nZ9hDcTUSSVOy3rwRXMVTHFSZNE7Fl1wz22X9kh91886BMr33ZBW
+yR4M0Q1AkEAxogSsFEc3osTjUgQhJbrpuvVI9tJImxZjGwVAM+uVNr7JkzRDopR
FbS4F8Rpv8om+C4UvWgHMq5QzvnOggesAQJAQwPANWC1GUrnG5AIct/0QNgo8wlR
uI0vlUaW9JEaNUg2AxsH/Mkyd45fKEWPqL5apVMR7q0OXJYk84bAPRApeQJAU2Dq
mGo4ea5nM7rEVSsdJSl+hmSePnrK0RLDb2fJi5G8UMWKN1UX4MnZVwICd63q1+hi
hXmfLEXRYlu/w1ewAQJAHgCcH3hcQ+pIKNN5i+nVVMsAKDkjF23MH3NKRq9AgO5p
ACXXzNsyaOL+tdn4lS6bIRYHFIwcDCJo1ozj/vVe5g==
-----END RSA PRIVATE KEY-----
[09/02/21]seed@VM:~/.../CA$ 
```

Next, we generate a certificate signing request (CSR):



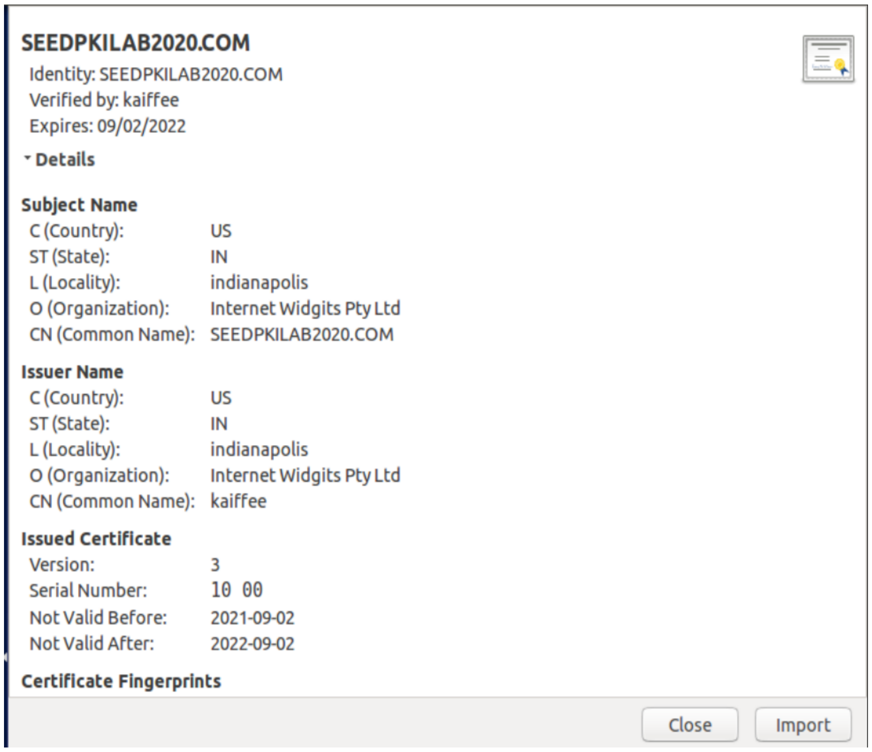
```
[09/02/21]seed@VM:~/.../CA$ openssl req -new -key server.key -out server.csr -config openss
l.cnf
Enter pass phrase for server.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:IN
Locality Name (eg, city) []:indianapolis
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:SEEDPKILAB2020.COM
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:deesdees
An optional company name []:
[09/02/21]seed@VM:~/.../CA$ 
```

And finally, we generate the certificates:



```
[09/02/21]seed@VM:~/.../CA$ 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: 4096 (0x1000)
        Validity
            Not Before: Sep  2 14:54:02 2021 GMT
            Not After : Sep  2 14:54:02 2022 GMT
        Subject:
            countryName               = US
            stateOrProvinceName       = IN
            localityName              = indianapolis
            organizationName          = Internet Widgits Pty Ltd
            commonName                = SEEDPKILAB2020.COM
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                0D:8D:39:C8:ED:90:F5:F1:B1:7C:C9:48:57:D5:21:48:A8:FC:B0:12
            X509v3 Authority Key Identifier:
                keyid:5E:EA:5C:69:3D:3E:E4:E1:32:C7:1C:4D:BE:4E:B6:5E:7C:63:89:43

Certificate is to be certified until Sep  2 14:54:02 2022 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
[09/02/21]seed@VM:~/.../CA$ 
```

And this is our new signed certificate, 1000.pem:



SEEDPKILAB2020.COM

Identity: SEEDPKILAB2020.COM
Verified by: kaiffee
Expires: 09/02/2022

Details

Subject Name

C (Country):	US
ST (State):	IN
L (Locality):	indianapolis
O (Organization):	Internet Widgits Pty Ltd
CN (Common Name):	SEEDPKILAB2020.COM

Issuer Name

C (Country):	US
ST (State):	IN
L (Locality):	indianapolis
O (Organization):	Internet Widgits Pty Ltd
CN (Common Name):	kaiffee

Issued Certificate

Version:	3
Serial Number:	10 00
Not Valid Before:	2021-09-02
Not Valid After:	2022-09-02

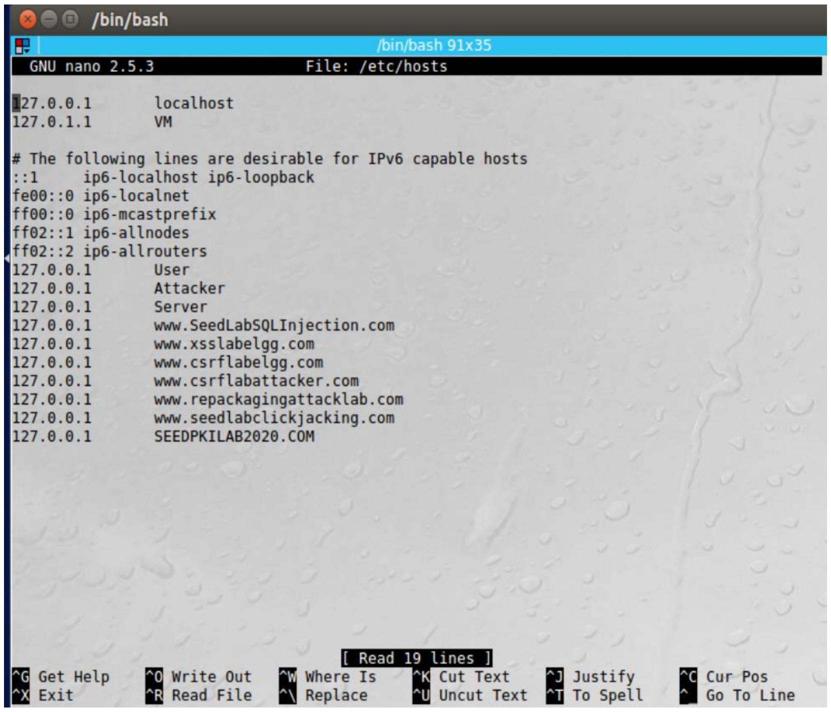
Certificate Fingerprints

Observation: In this task we created the certificates needed to secure our website, SEEDPKILab2020.com using https. We used the authority that we created in the previous task to issue the certificate. We followed three steps to create the certificate, first we had to generate the public/private key pair, which was stored in server.key. Second, we generated a signing request which included the public key and is sent to the CA. Third, we use the signing request, server.csr to generate the certificate using our CA. We used the ‘openssl ca’ command for this.

Explanation: The process for issuing valid certificates from a CA involves three parts, generating the public/private key pair, creating a certificate signing request using the public key, then issuing the certificate. With this certificate we will be able to secure our website in the next tasks.

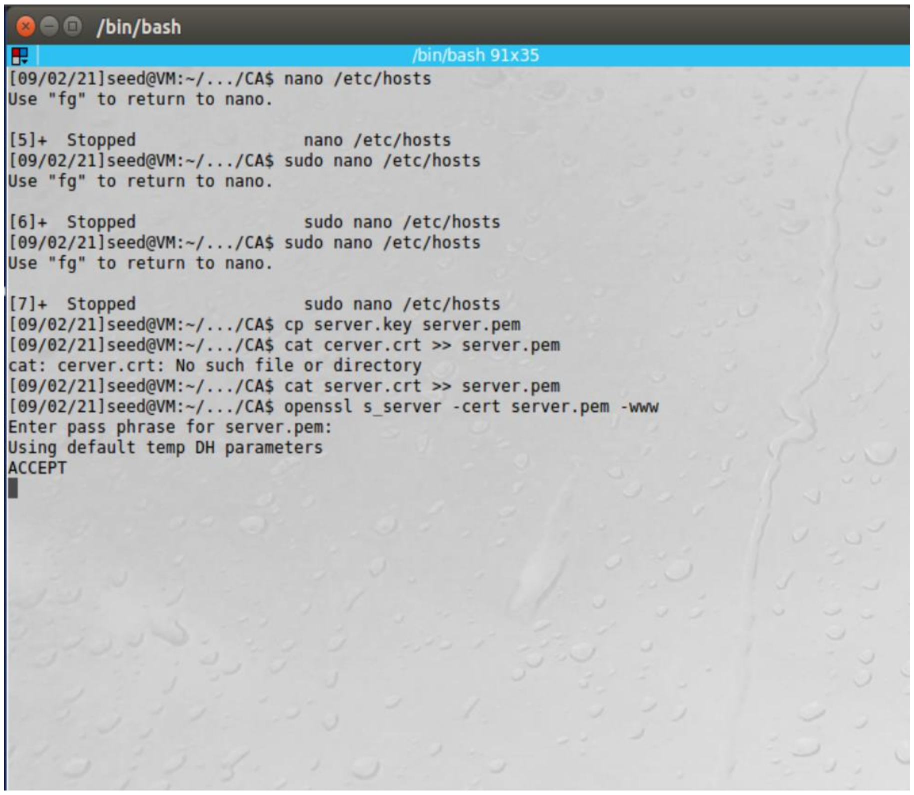
Task 3: Deploying Certificate in an HTTPS Web Server

The first thing we do is edit the /etc/hosts file to redirect SEEDPKILab2020.com to our localhost:



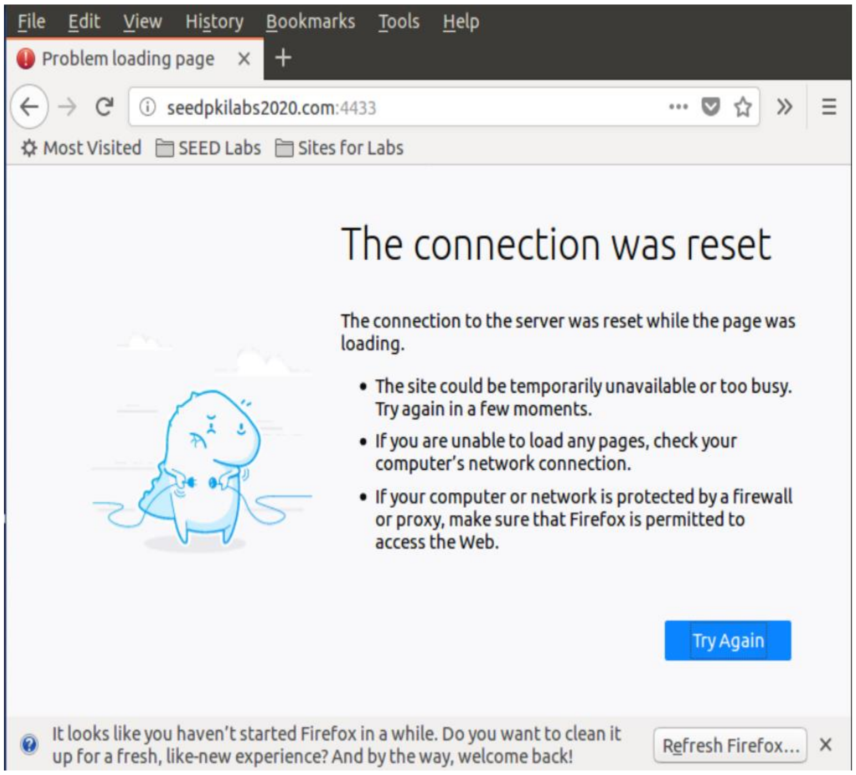
```
  GNU nano 2.5.3                  File: /etc/hosts

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.csrflabattacker.com
127.0.0.1       www.repackagingattacklab.com
127.0.0.1       www.seedlabclickjacking.com
127.0.0.1       SEEDPKILAB2020.COM

                                    [ Read 19 lines ]
^G Get Help     ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos
^X Exit         ^R Read File    ^\ Replace      ^U Uncut Text   ^T To Spell     ^_ Go To Line
```

Next we configure our webserver to use our certificate and will be available at https://SEEDPKILab2020.com:



```
[09/02/21]seed@VM:~/.../CA$ nano /etc/hosts
Use "fg" to return to nano.

[5]+  Stopped                 nano /etc/hosts
[09/02/21]seed@VM:~/.../CA$ sudo nano /etc/hosts
Use "fg" to return to nano.

[6]+  Stopped                 sudo nano /etc/hosts
[09/02/21]seed@VM:~/.../CA$ sudo nano /etc/hosts
Use "fg" to return to nano.

[7]+  Stopped                 sudo nano /etc/hosts
[09/02/21]seed@VM:~/.../CA$ cp server.key server.pem
[09/02/21]seed@VM:~/.../CA$ cat cerver.crt >> server.pem
cat: cerver.crt: No such file or directory
[09/02/21]seed@VM:~/.../CA$ cat server.crt >> server.pem
[09/02/21]seed@VM:~/.../CA$ openssl s_server -cert server.pem -www
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
```

When we visit our website, we get a warning from Firefox that our connection is not secure since we still need to install our certificate repository:



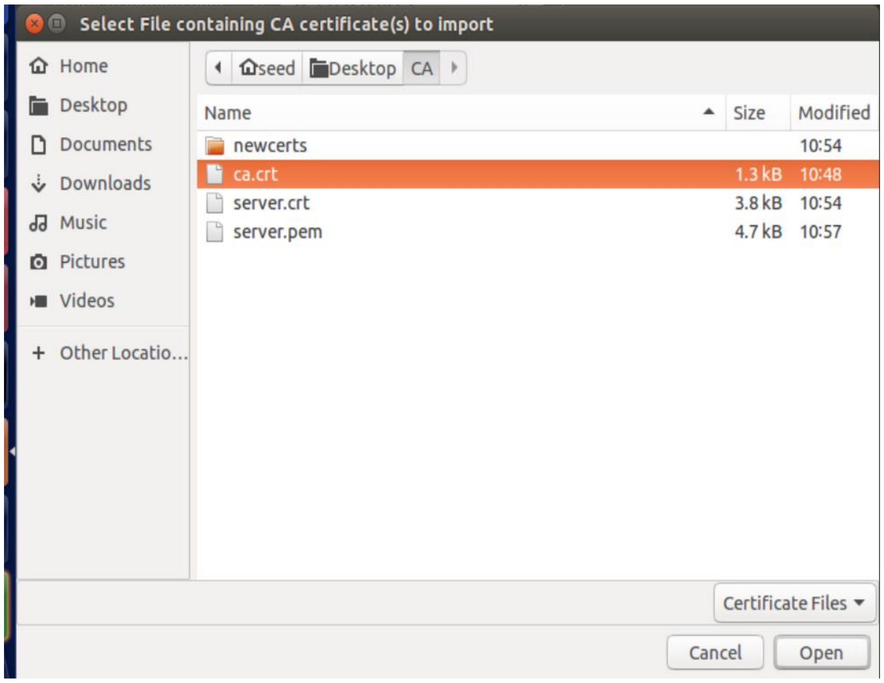
The connection was reset

The connection to the server was reset while the page was loading.

- The site could be temporarily unavailable or too busy. Try again in a few moments.
- If you are unable to load any pages, check your computer's network connection.
- If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the Web.

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back!

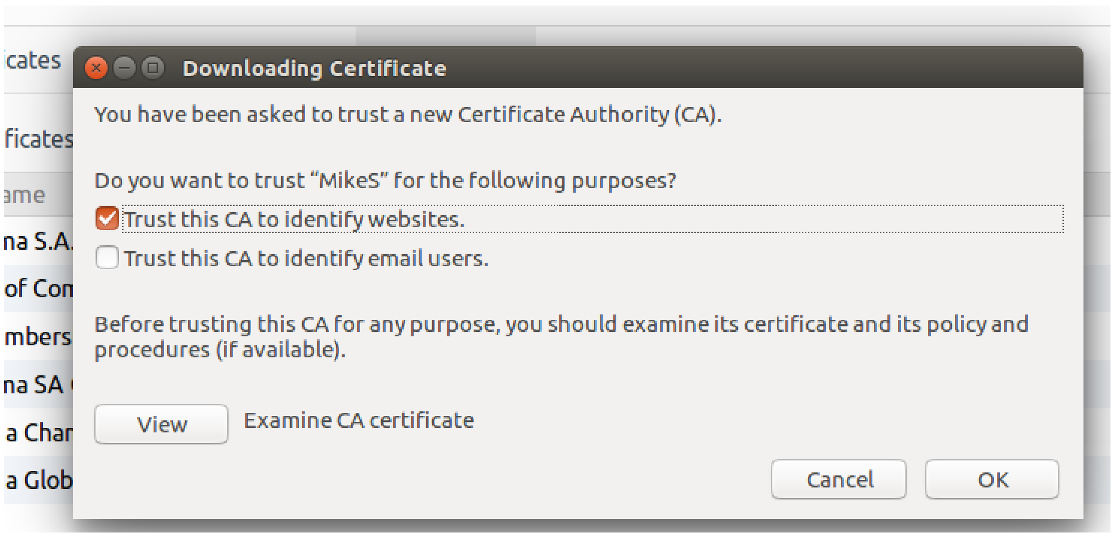
Now we installed the Certificate Authority that we made to Firefox:



Select File containing CA certificate(s) to import

Name	Size	Modified
newcerts		10:54
ca.crt	1.3 kB	10:48
server.crt	3.8 kB	10:54
server.pem	4.7 kB	10:57

Trusting the CA:



Downloading Certificate

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

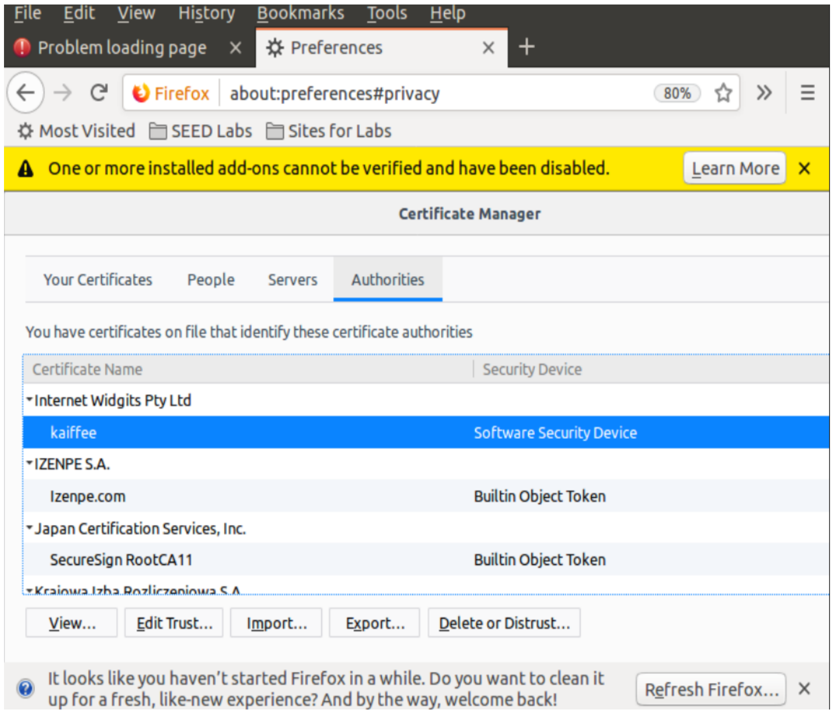
Do you want to trust "MikeS" for the following purposes?

[✓] 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).

We can see the certificate now is in our list of trusted certificates with "MikeS" as the signer:


Certificate Manager

Your Certificates | People | Servers | Authorities

You have certificates on file that identify these certificate authorities

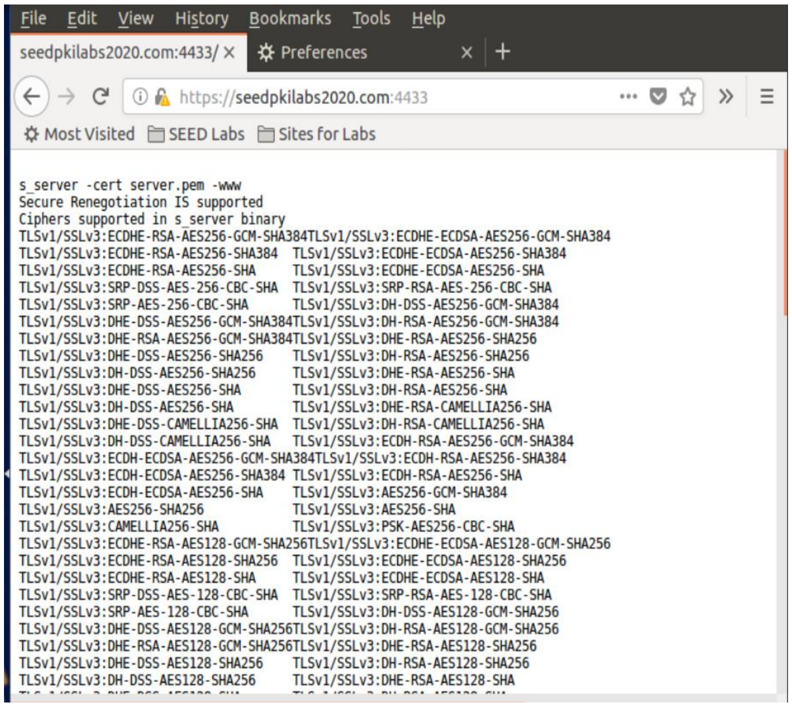
Certificate Name	Security Device
Internet Widgits Pty Ltd	
kaiffee	Software Security Device
IZENPE S.A.	
Izenpe.com	Builtin Object Token
Japan Certification Services, Inc.	
SecureSign RootCA11	Builtin Object Token

One or more installed add-ons cannot be verified and have been disabled.

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back!

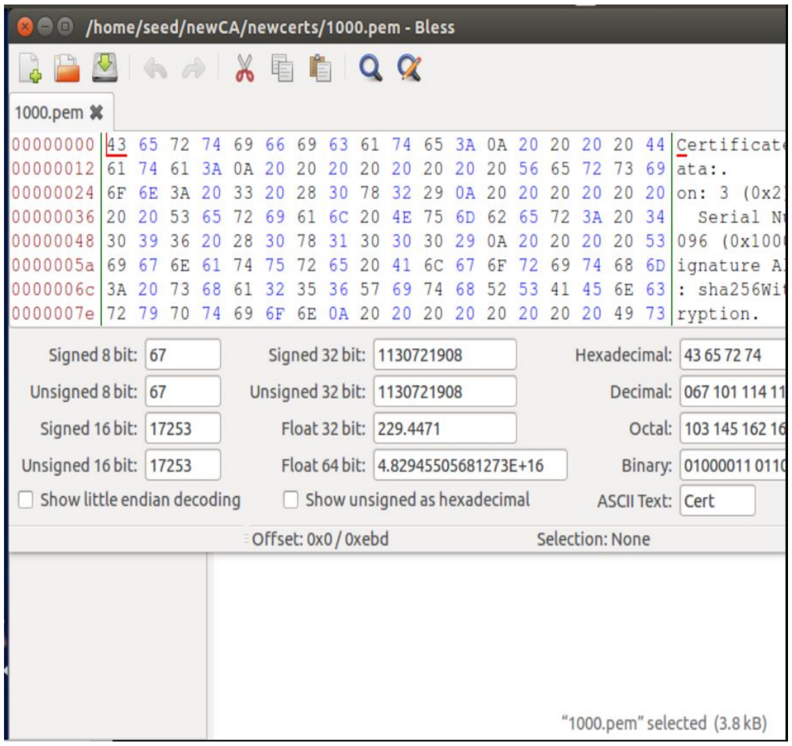
File Edit View History Bookmarks Tools Help
Problem loading page
Preferences
Firefox about:preferences#privacy
80%
Most Visited SEED Labs Sites for Labs
Learn More
View... Edit Trust... Import... Export... Delete or Distrust...
Refresh Firefox...

And when we try to visit our website again, we don't get a security warning:

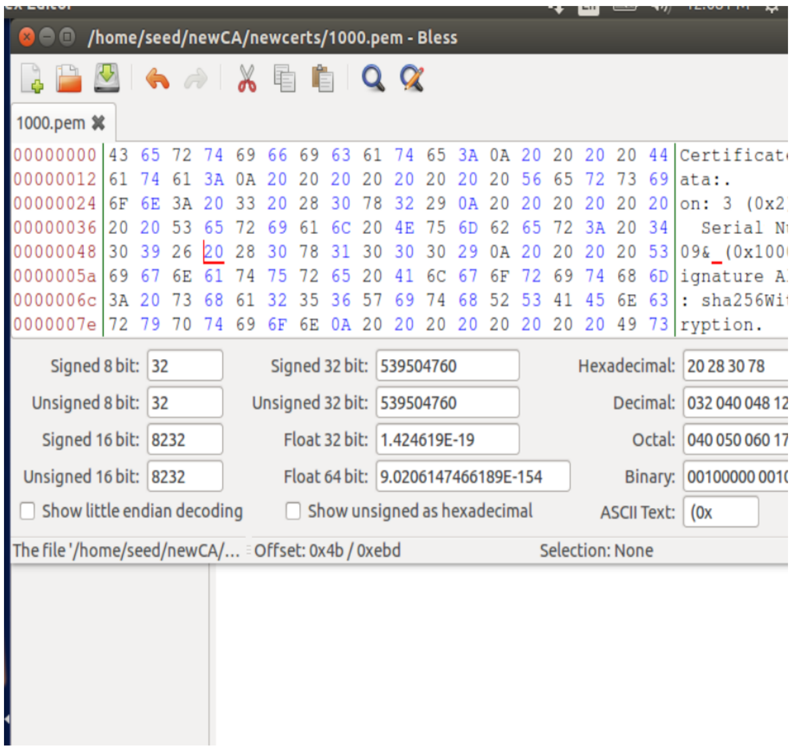

File Edit View History Bookmarks Tools Help
seedpkilabs2020.com:4433/
Preferences
https://seedpkilabs2020.com:4433
Most Visited SEED Labs Sites for Labs

```
s_server -cert server.pem -www
Secure Renegotiation IS supported
Ciphers supported in s_server binary
TLSv1/SSLv3:ECDHE-RSA-AES256-GCM-SHA384TLSv1/SSLv3:ECDHE-ECDSA-AES256-GCM-SHA384
TLSv1/SSLv3:ECDHE-RSA-AES256-SHA384  TLSv1/SSLv3:ECDHE-ECDSA-AES256-SHA384
TLSv1/SSLv3:ECDHE-RSA-AES256-SHA     TLSv1/SSLv3:ECDHE-ECDSA-AES256-SHA
TLSv1/SSLv3:SRP-DSS-AES-256-CBC-SHA  TLSv1/SSLv3:SRP-RSA-AES-256-CBC-SHA
TLSv1/SSLv3:SRP-AES-256-CBC-SHA      TLSv1/SSLv3:DH-DSS-AES256-GCM-SHA384
TLSv1/SSLv3:DHE-DSS-AES256-GCM-SHA384TLSv1/SSLv3:DH-RSA-AES256-GCM-SHA384
TLSv1/SSLv3:DHE-RSA-AES256-GCM-SHA384TLSv1/SSLv3:DHE-RSA-AES256-SHA256
TLSv1/SSLv3:DHE-DSS-AES256-SHA256    TLSv1/SSLv3:DH-RSA-AES256-SHA256
TLSv1/SSLv3:DH-DSS-AES256-SHA256     TLSv1/SSLv3:DHE-RSA-AES256-SHA
TLSv1/SSLv3:DHE-DSS-AES256-SHA       TLSv1/SSLv3:DH-RSA-AES256-SHA
TLSv1/SSLv3:DH-DSS-AES256-SHA        TLSv1/SSLv3:DHE-RSA-CAMELLIA256-SHA
TLSv1/SSLv3:DHE-DSS-CAMELLIA256-SHA  TLSv1/SSLv3:DH-RSA-CAMELLIA256-SHA
TLSv1/SSLv3:DH-DSS-CAMELLIA256-SHA   TLSv1/SSLv3:ECDH-RSA-AES256-GCM-SHA384
TLSv1/SSLv3:ECDH-ECDSA-AES256-GCM-SHA384TLSv1/SSLv3:ECDH-RSA-AES256-SHA384
TLSv1/SSLv3:ECDH-ECDSA-AES256-SHA384 TLSv1/SSLv3:ECDH-RSA-AES256-SHA
TLSv1/SSLv3:ECDH-ECDSA-AES256-SHA    TLSv1/SSLv3:AES256-GCM-SHA384
TLSv1/SSLv3:AES256-SHA256            TLSv1/SSLv3:AES256-SHA
TLSv1/SSLv3:CAMELLIA256-SHA          TLSv1/SSLv3:PSK-AES256-CBC-SHA
TLSv1/SSLv3:ECDHE-RSA-AES128-GCM-SHA256TLSv1/SSLv3:ECDHE-ECDSA-AES128-GCM-SHA256
TLSv1/SSLv3:ECDHE-RSA-AES128-SHA256  TLSv1/SSLv3:ECDHE-ECDSA-AES128-SHA256
TLSv1/SSLv3:ECDHE-RSA-AES128-SHA     TLSv1/SSLv3:ECDHE-ECDSA-AES128-SHA
TLSv1/SSLv3:SRP-DSS-AES-128-CBC-SHA  TLSv1/SSLv3:SRP-RSA-AES-128-CBC-SHA
TLSv1/SSLv3:SRP-AES-128-CBC-SHA      TLSv1/SSLv3:DH-DSS-AES128-GCM-SHA256
TLSv1/SSLv3:DHE-DSS-AES128-GCM-SHA256TLSv1/SSLv3:DH-RSA-AES128-GCM-SHA256
TLSv1/SSLv3:DHE-RSA-AES128-GCM-SHA256TLSv1/SSLv3:DHE-RSA-AES128-SHA256
TLSv1/SSLv3:DHE-DSS-AES128-SHA256    TLSv1/SSLv3:DH-RSA-AES128-SHA256
TLSv1/SSLv3:DH-DSS-AES128-SHA256     TLSv1/SSLv3:DHE-RSA-AES128-SHA
```

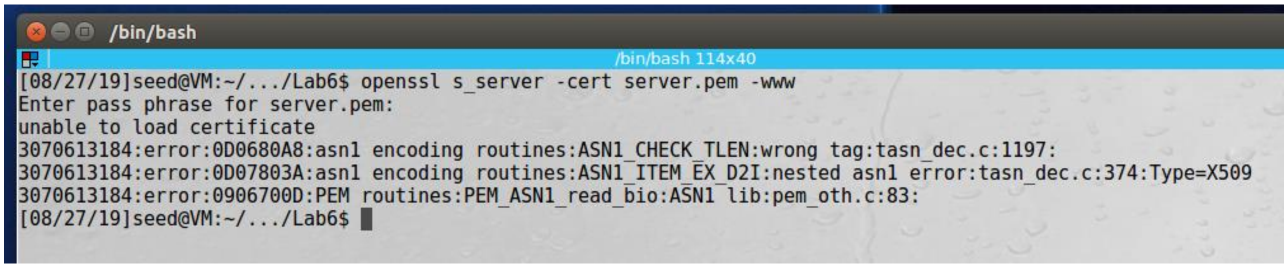
For the next part we will change one byte on server.pem, here is the file before we change the byte:


```
/home/seed/newCA/newcerts/1000.pem - Bless
1000.pem
00000000 43 65 72 74 69 66 69 63 61 74 65 3A 0A 20 20 20 20 44 Certificat
00000012 61 74 61 3A 0A 20 20 20 20 20 20 20 20 56 65 72 73 69 ata:.
00000024 6F 6E 3A 20 33 20 28 30 78 32 29 0A 20 20 20 20 20 20 on: 3 (0x2
00000036 20 20 53 65 72 69 61 6C 20 4E 75 6D 62 65 72 3A 20 34   Serial N
00000048 30 39 36 20 28 30 78 31 30 30 30 29 0A 20 20 20 20 53 096 (0x100
0000005a 69 67 6E 61 74 75 72 65 20 41 6C 67 6F 72 69 74 68 6D ignature A
0000006c 3A 20 73 68 61 32 35 36 57 69 74 68 52 53 41 45 6E 63 : sha256Wi
0000007e 72 79 70 74 69 6F 6E 0A 20 20 20 20 20 20 20 20 49 73 ryption.

Signed 8 bit: 67          Signed 32 bit: 1130721908             Hexadecimal: 43 65 72 74
Unsigned 8 bit: 67        Unsigned 32 bit: 1130721908           Decimal: 067 101 114 11
Signed 16 bit: 17253      Float 32 bit: 229.4471                Octal: 103 145 162 16
Unsigned 16 bit: 17253    Float 64 bit: 4.82945505681273E+16    Binary: 01000011 0110
Show little endian decoding   Show unsigned as hexadecimal      ASCII Text: Cert
Offset: 0x0 / 0xebd    Selection: None
"1000.pem" selected (3.8 kB)
```

And now we change the byte from a “36” to a “26”

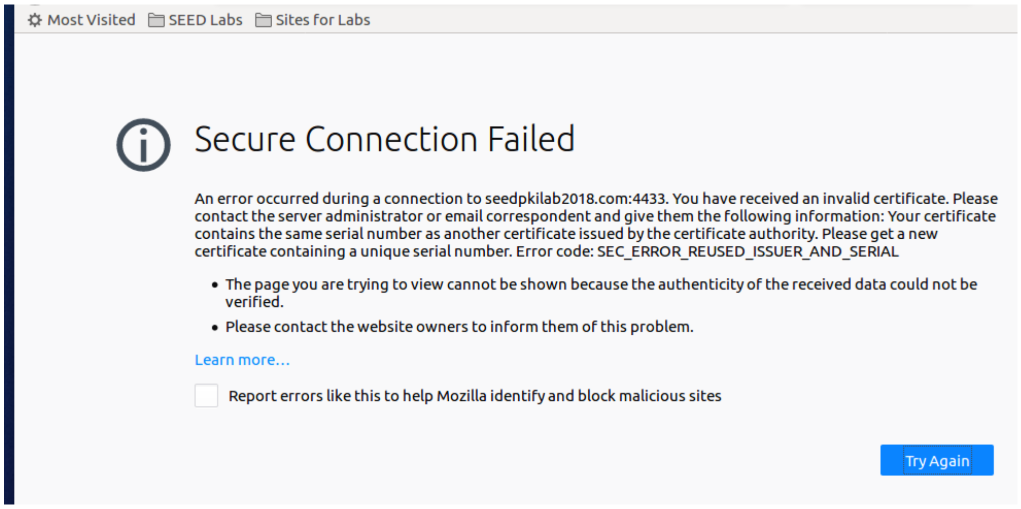

```
/home/seed/newCA/newcerts/1000.pem - Bless
1000.pem
00000000 43 65 72 74 69 66 69 63 61 74 65 3A 0A 20 20 20 20 44 Certificat
00000012 61 74 61 3A 0A 20 20 20 20 20 20 20 20 56 65 72 73 69 ata:.
00000024 6F 6E 3A 20 33 20 28 30 78 32 29 0A 20 20 20 20 20 20 on: 3 (0x2
00000036 20 20 53 65 72 69 61 6C 20 4E 75 6D 62 65 72 3A 20 34   Serial N
00000048 30 39 26 20 28 30 78 31 30 30 30 29 0A 20 20 20 20 53 09& (0x100
0000005a 69 67 6E 61 74 75 72 65 20 41 6C 67 6F 72 69 74 68 6D ignature A
0000006c 3A 20 73 68 61 32 35 36 57 69 74 68 52 53 41 45 6E 63 : sha256Wi
0000007e 72 79 70 74 69 6F 6E 0A 20 20 20 20 20 20 20 20 49 73 ryption.

Signed 8 bit: 32          Signed 32 bit: 539504760              Hexadecimal: 20 28 30 78
Unsigned 8 bit: 32        Unsigned 32 bit: 539504760            Decimal: 032 040 048 12
Signed 16 bit: 8232       Float 32 bit: 1.424619E-19            Octal: 040 050 060 17
Unsigned 16 bit: 8232     Float 64 bit: 9.0206147466189E-154    Binary: 00100000 0010
Show little endian decoding   Show unsigned as hexadecimal      ASCII Text: (0x
The file '/home/seed/newCA/...  Offset: 0x4b / 0xebd    Selection: None
```

And when we try to start the server it will not start:



```
[08/27/19]seed@VM:~/.../Lab6$ openssl s_server -cert server.pem -www
Enter pass phrase for server.pem:
unable to load certificate
3070613184:error:0D0680A8:asn1 encoding routines:ASN1_CHECK_TLEN:wrong tag:tasn_dec.c:1197:
3070613184:error:0D07803A:asn1 encoding routines:ASN1_ITEM_EX_D2I:nested asn1 error:tasn_dec.c:374:Type=X509
3070613184:error:0906700D:PEM routines:PEM_ASN1_read_bio:ASN1 lib:pem_oth.c:83:
[08/27/19]seed@VM:~/.../Lab6$
```

Now we can start our server, but when we browse to https://seedpkilab2020.com:4433, we get a failure:



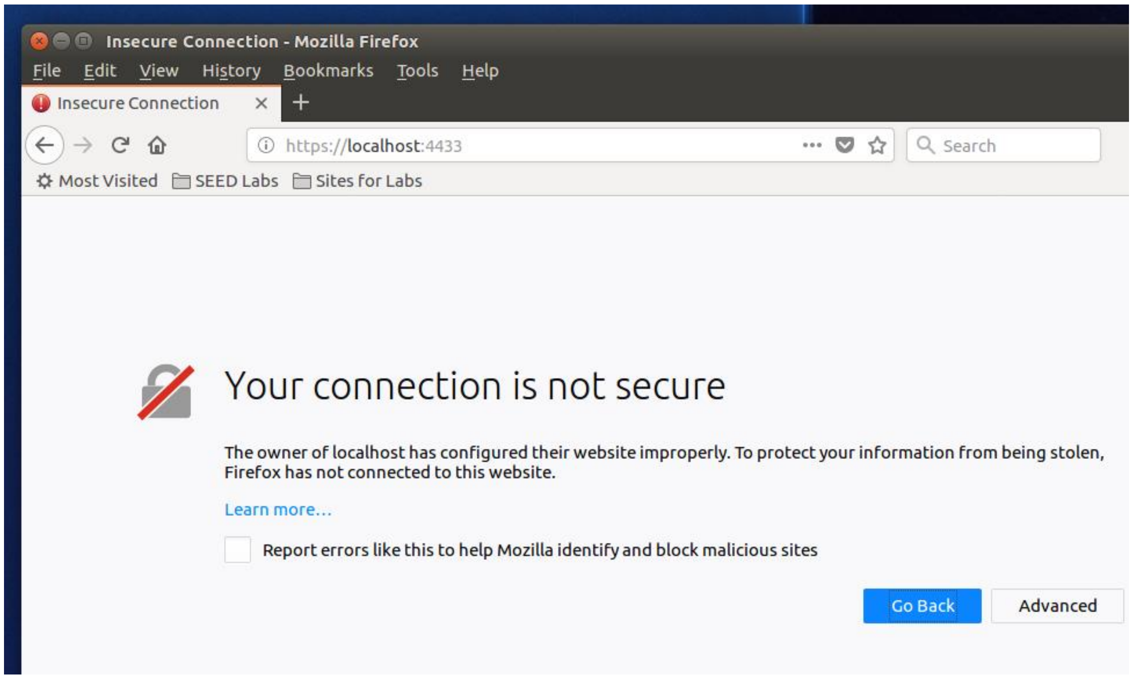
Secure Connection Failed

An error occurred during a connection to seedpkilab2018.com:4433. You have received an invalid certificate. Please contact the server administrator or email correspondent and give them the following information: Your certificate contains the same serial number as another certificate issued by the certificate authority. Please get a new certificate containing a unique serial number. Error code: SEC_ERROR_REUSED_ISSUER_AND_SERIAL

- The page you are trying to view cannot be shown because the authenticity of the received data could not be verified.
- Please contact the website owners to inform them of this problem.

Most Visited
SEED Labs
Sites for Labs
Learn more…
Report errors like this to help Mozilla identify and block malicious sites
Try Again

If we attempt to go to https://localhost:4433 we get a security warning because the certificate was registered for https://seedpkilab2020.com:4433 and not for localhost:



Your connection is not secure

The owner of localhost has configured their website improperly. To protect your information from being stolen, Firefox has not connected to this website.

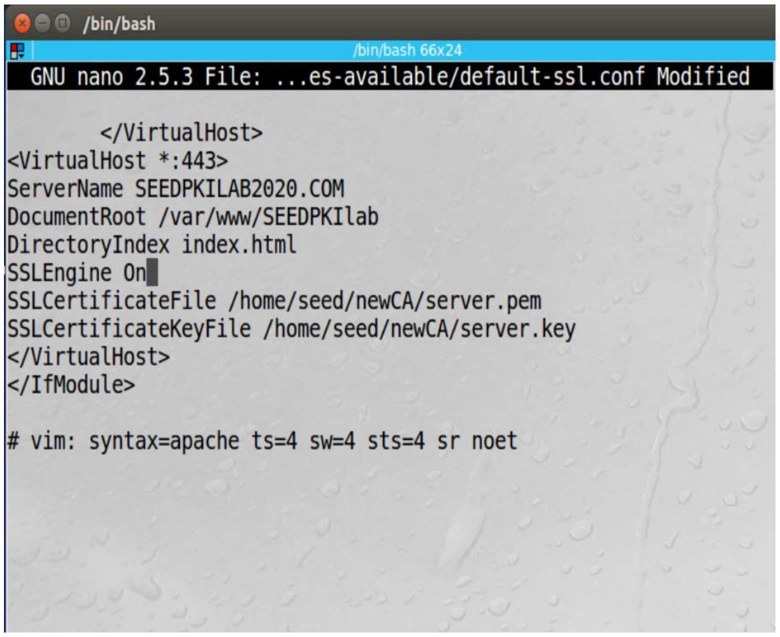
Insecure Connection - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Insecure Connection
https://localhost:4433
Search
Most Visited
SEED Labs
Sites for Labs
Learn more…
Report errors like this to help Mozilla identify and block malicious sites
Go Back
Advanced

Observation: In this task we deploy and test our certificate and then try to corrupt it to see the result. We first had to add the site to our /etc/hosts file, and then we combined the secret key and certificate into one file, server.pem and launched our web server using s_server. When we first visited the site in Firefox, we were prompted with a security warning because we hadn't added our CA to Firefox's trusted CAs. After we added the CA to Firefox to trust, we no longer get a security warning and can now see our website working properly. We then go into the server's certificate to corrupt a byte, restart our webserver to see if it still works, but Firefox tells us that our connection is no longer secure.

Explanation: Deploying the certificate to the website allowed us to visit our site securely using https. We first needed to at the CA that we created to Firefox since we didn't use a commercial provider like Verisign. When we corrupted our certificate file and tried again, after only changing one byte, Firefox gave us a security warning.

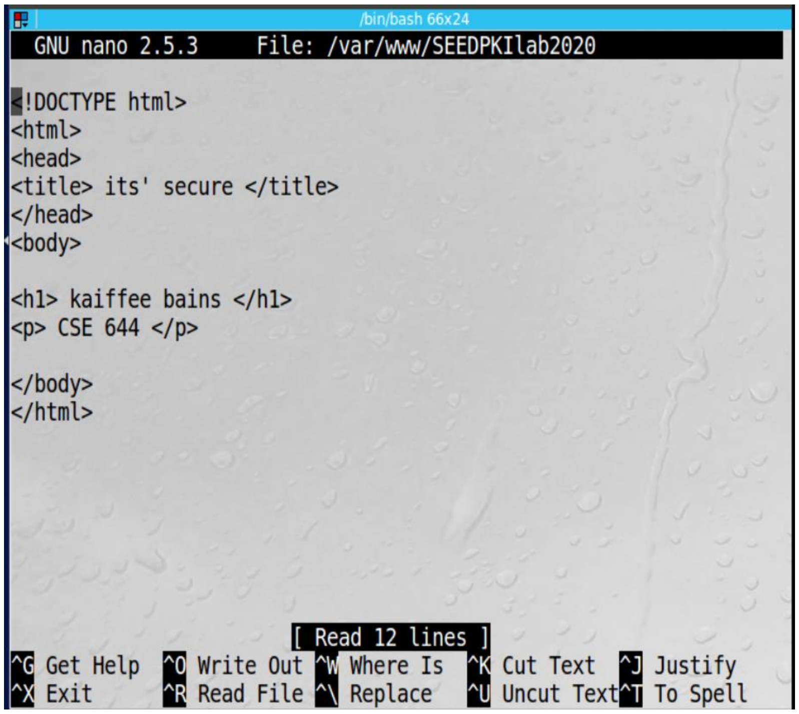
Task 4: Deploying Certificate in an Apache-Based HTTPS Web Server

First, we modify the default-ssl.conf file to include information about our website:



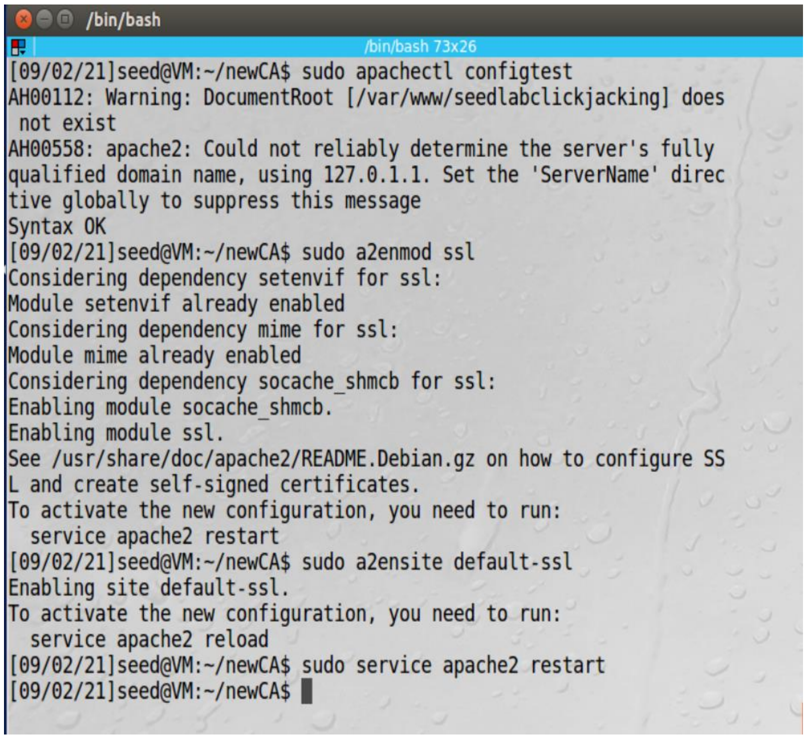
```
  GNU nano 2.5.3 File: ...es-available/default-ssl.conf Modified

        </VirtualHost>
<VirtualHost *:443>
ServerName SEEDPKILAB2020.COM
DocumentRoot /var/www/SEEDPKIlab
DirectoryIndex index.html
SSLEngine On
SSLCertificateFile /home/seed/newCA/server.pem
SSLCertificateKeyFile /home/seed/newCA/server.key
</VirtualHost>
</IfModule>

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

Next we create a simple website located at /var/www/SEEDPKILab2020:



```
/bin/bash 66x24
  GNU nano 2.5.3      File: /var/www/SEEDPKIlab2020

<!DOCTYPE html>
<html>
<head>
<title> its' secure </title>
</head>
<body>

<h1> kaiffee bains </h1>
<p> CSE 644 </p>

</body>
</html>

                         [ Read 12 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell
```

Next we run some commands to check that our site's security is properly configured:



```
/bin/bash
/bin/bash 73x26
[09/02/21]seed@VM:~/newCA$ 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
[09/02/21]seed@VM:~/newCA$ 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:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SS
L and create self-signed certificates.
To activate the new configuration, you need to run:
  service apache2 restart
[09/02/21]seed@VM:~/newCA$ sudo a2ensite default-ssl
Enabling site default-ssl.
To activate the new configuration, you need to run:
  service apache2 reload
[09/02/21]seed@VM:~/newCA$ sudo service apache2 restart
[09/02/21]seed@VM:~/newCA$
```

Observation: This task was similar to the previous except we used Apache as our webserver. Apache is a popular webserver and is widely used in the real world. We first had to edit the file at /etc/apache2/default-ssl.conf file to point to our website and certificates. Then we created a simple html file in the /var/www/SEEDPKILab2020 folder. We then ran some command to check if everything was configured properly, restarted our apache server, and visited the site in Firefox.

When we visited the site, it showed as secure in the browser, with valid certificates since we previously installed the CA.

Explanation: Apache is an easy to configure and deploy web server that we used to host our website SEEDPKILabs2020.com. We were able to have our secure website up and running in only a few minutes after configuring our certificates and browser to trust them.