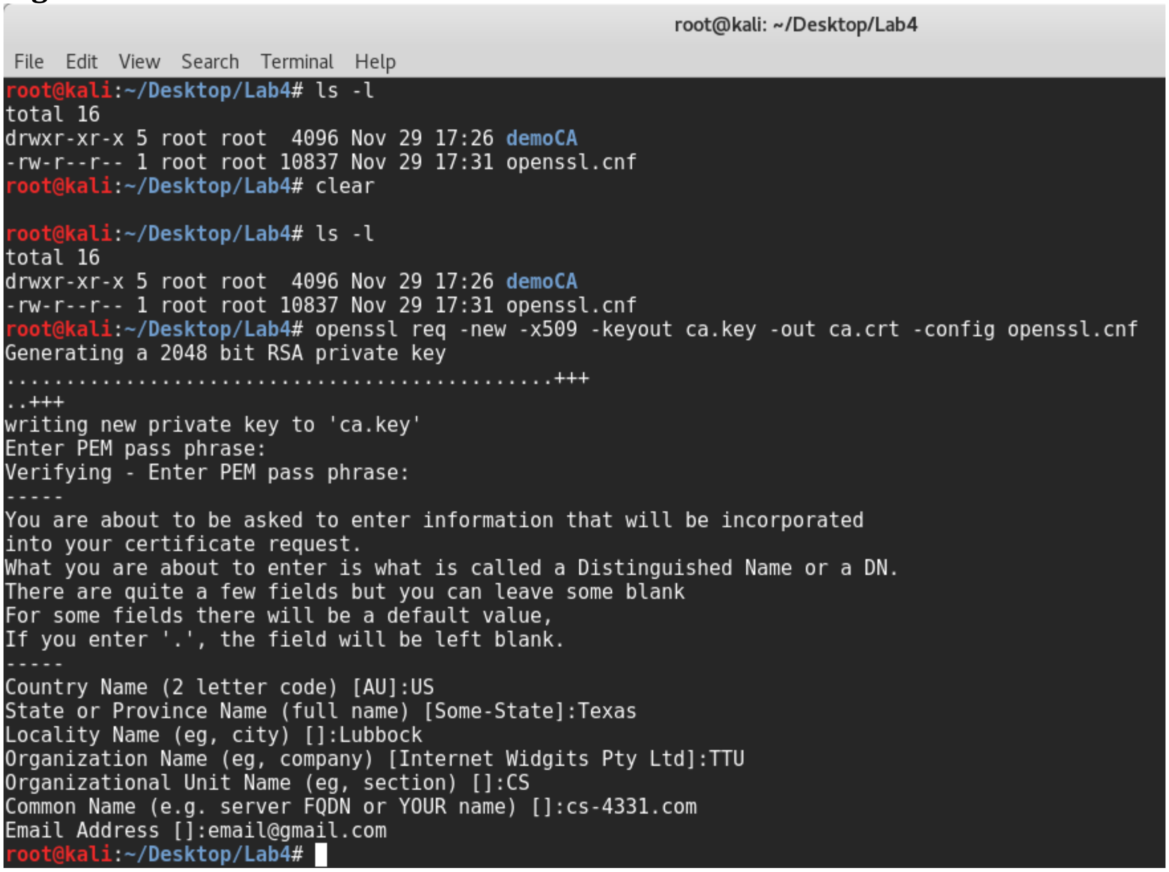
CS4331 John Toland
Lab4 11/30/16

Creating and Using Digital Certificates

Introduction:
This report will explain if full detail the process of acting as a certificate authority and requesting a signature from a certificate authority. Also the report will show the process of changing the local host name, and adding the certificate to the browser's list of certificates. Allowing the browser to display the webpage, which is the new name of the local host. This allows for a mock website to be used.

1. **Creating Lab4 and demoCA directories:** Before starting the process, I copied the file openssl.cnf to my directory Lab4 were the work takes place. I then created the demoCA directory and in demoCA I created the following directories: certs, crl, and newcerts. While still in demoCA I then created an empty file named index.txt and a file named serial with one entry of "1000."
2. **Creating the root certificate (CA):** Now I am able to use create the root certificate that is the certificate authority. The command creates a keypair for the CA key and places it in the CA certificate file and also uses the configuration I chose rather than the default configuration. After running the command the CA key requires a passphrase for encryption. Then the fields for the certificate must be filled out. See **Figure 2.1** below.

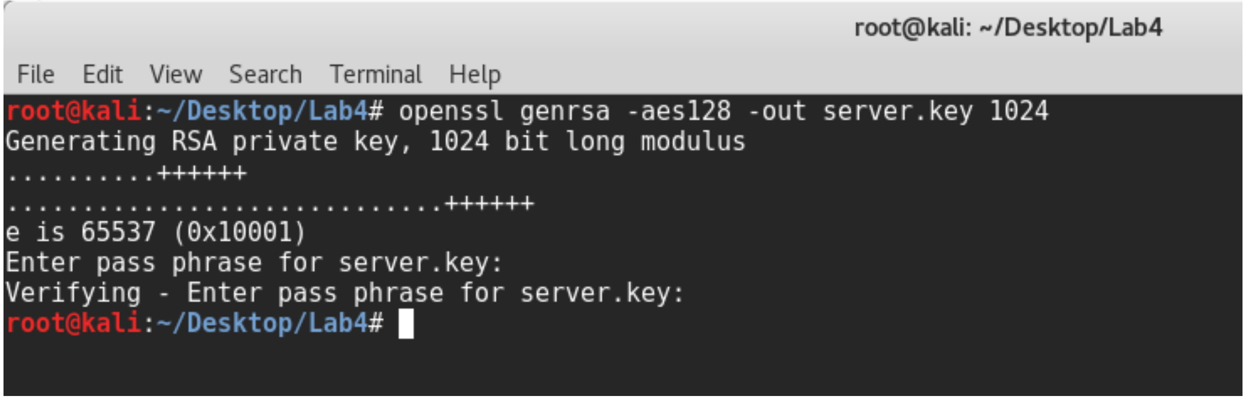
Figure 2.1


root@kali: ~/Desktop/Lab4
File Edit View Search Terminal Help

```
root@kali:~/Desktop/Lab4# ls -l
total 16
drwxr-xr-x 5 root root  4096 Nov 29 17:26 demoCA
-rw-r--r-- 1 root root 10837 Nov 29 17:31 openssl.cnf
root@kali:~/Desktop/Lab4# clear

root@kali:~/Desktop/Lab4# ls -l
total 16
drwxr-xr-x 5 root root  4096 Nov 29 17:26 demoCA
-rw-r--r-- 1 root root 10837 Nov 29 17:31 openssl.cnf
root@kali:~/Desktop/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]:US
State or Province Name (full name) [Some-State]:Texas
Locality Name (eg, city) []:Lubbock
Organization Name (eg, company) [Internet Widgits Pty Ltd]:TTU
Organizational Unit Name (eg, section) []:CS
Common Name (e.g. server FQDN or YOUR name) []:cs-4331.com
Email Address []:email@gmail.com
root@kali:~/Desktop/Lab4# 
```

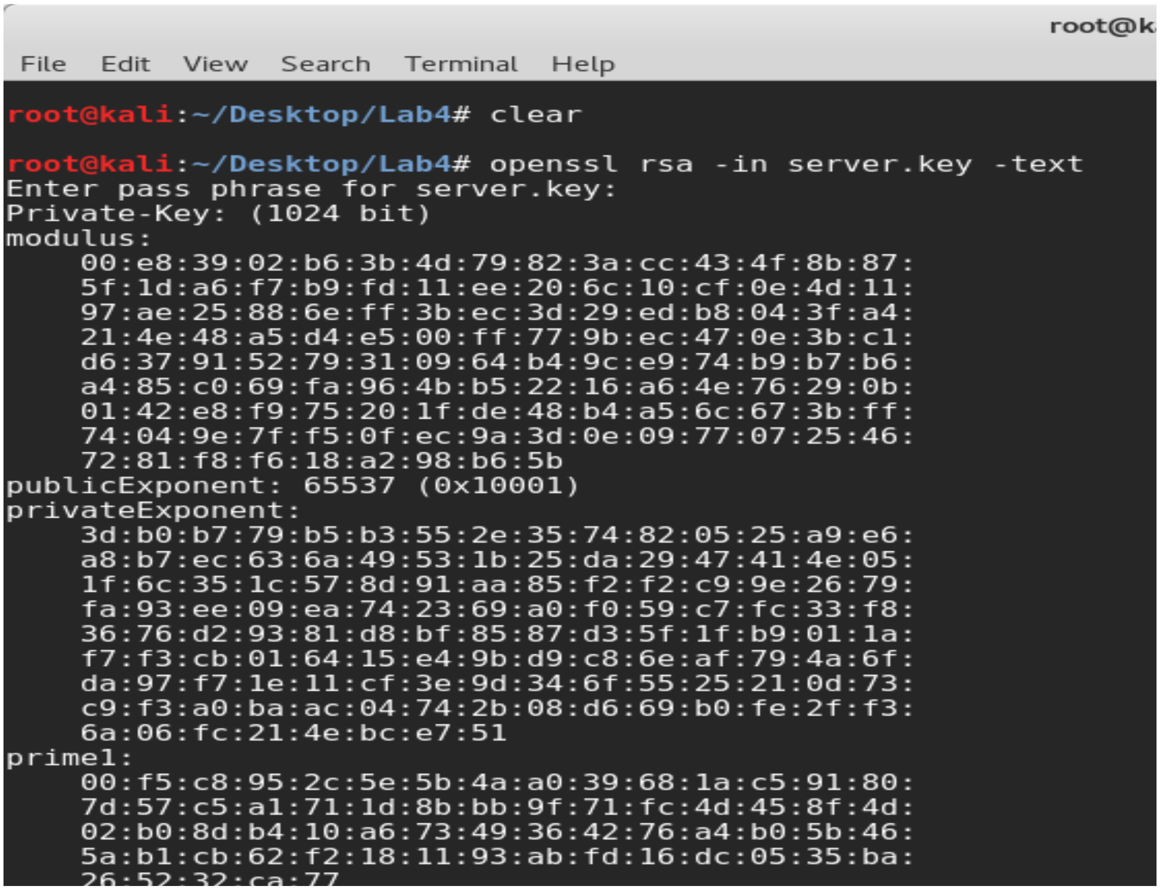
3. **Creating a Certificate Signing Request (CSR):** After creating the CA I can now sign certificates. The first client will be cs-4331.com. Before sending the CSR to the CA the client must create a private/public keypair. The CSR only contains the keypair and information about the client. The CA will create and sign the certificate upon receiving the CSR from the client. Since the keypair is encrypted with aes I also provide a passphrase. See **Figure 3.1** below for generating the keypair. Before creating the CSR I viewed the client keypair. To view the file it must be decrypted, see **Figure 3.2-3.3** below. I then generated the CSR see **Figure 3.4** below. The command also requires some information to be filled out, this time by the client. The "Common Name" field is very important in this case. It must be the clients url name "cs-4331.com."

Figure 3.1


root@kali: ~/Desktop/Lab4
File Edit View Search Terminal Help

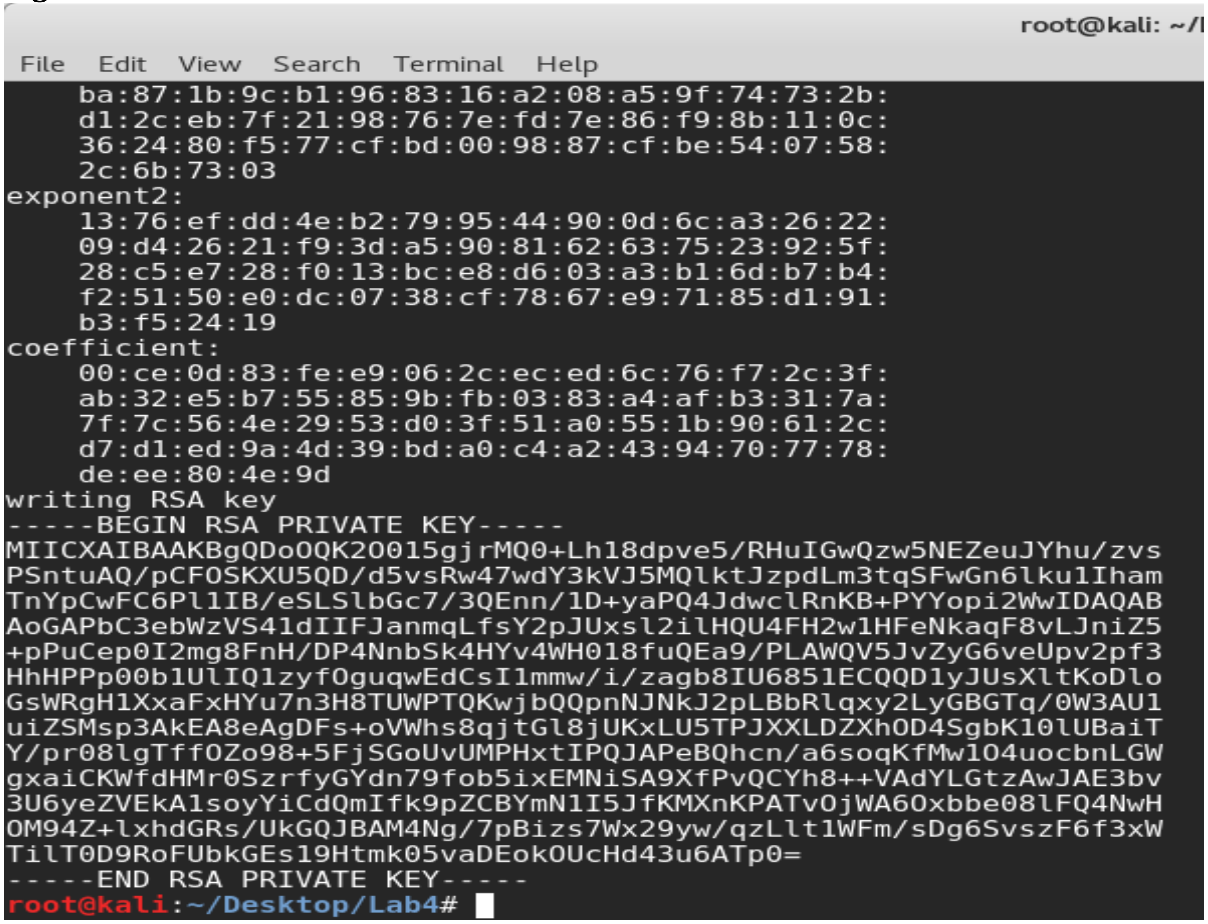
```
root@kali:~/Desktop/Lab4# openssl genrsa -aes128 -out server.key 1024
Generating RSA private key, 1024 bit long modulus
..........++++++
.............................++++++
e is 65537 (0x10001)
Enter pass phrase for server.key:
Verifying - Enter pass phrase for server.key:
root@kali:~/Desktop/Lab4# 
```

Figure 3.2



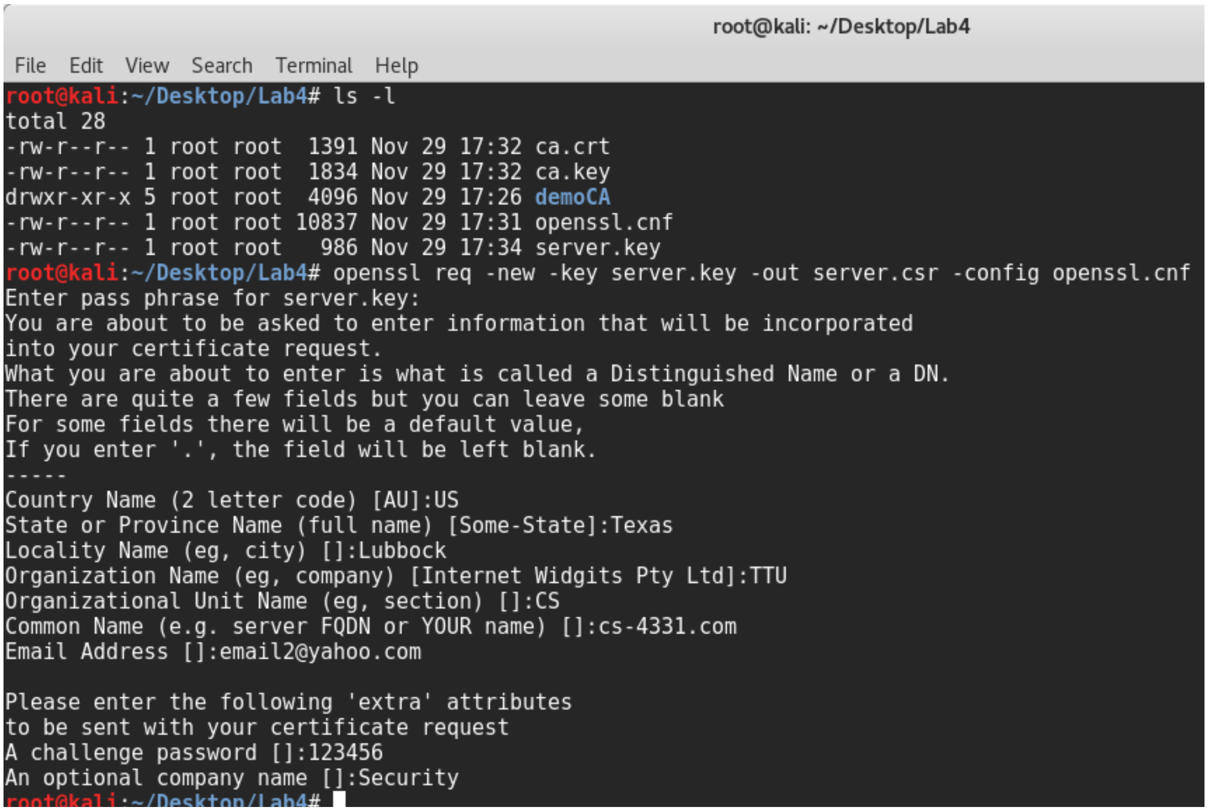
```
root@kali:~/Desktop/Lab4# clear

root@kali:~/Desktop/Lab4# openssl rsa -in server.key -text
Enter pass phrase for server.key:
Private-Key: (1024 bit)
modulus:
    00:e8:39:02:b6:3b:4d:79:82:3a:cc:43:4f:8b:87:
    5f:1d:a6:f7:b9:fd:11:ee:20:6c:10:cf:0e:4d:11:
    97:ae:25:88:6e:ff:3b:ec:3d:29:ed:b8:04:3f:a4:
    21:4e:48:a5:d4:e5:00:ff:77:9b:ec:47:0e:3b:c1:
    d6:37:91:52:79:31:09:64:b4:9c:e9:74:b9:b7:b6:
    a4:85:c0:69:fa:96:4b:b5:22:16:a6:4e:76:29:0b:
    01:42:e8:f9:75:20:1f:de:48:b4:a5:6c:67:3b:ff:
    74:04:9e:7f:f5:0f:ec:9a:3d:0e:09:77:07:25:46:
    72:81:f8:f6:18:a2:98:b6:5b
publicExponent: 65537 (0x10001)
privateExponent:
    3d:b0:b7:79:b5:b3:55:2e:35:74:82:05:25:a9:e6:
    a8:b7:ec:63:6a:49:53:1b:25:da:29:47:41:4e:05:
    1f:6c:35:1c:57:8d:91:aa:85:f2:f2:c9:9e:26:79:
    fa:93:ee:09:ea:74:23:69:a0:f0:59:c7:fc:33:f8:
    36:76:d2:93:81:d8:bf:85:87:d3:5f:1f:b9:01:1a:
    f7:f3:cb:01:64:15:e4:9b:d9:c8:6e:af:79:4a:6f:
    da:97:f7:1e:11:cf:3e:9d:34:6f:55:25:21:0d:73:
    c9:f3:a0:ba:ac:04:74:2b:08:d6:69:b0:fe:2f:f3:
    6a:06:fc:21:4e:bc:e7:51
prime1:
    00:f5:c8:95:2c:5e:5b:4a:a0:39:68:1a:c5:91:80:
    7d:57:c5:a1:71:1d:8b:bb:9f:71:fc:4d:45:8f:4d:
    02:b0:8d:b4:10:a6:73:49:36:42:76:a4:b0:5b:46:
    5a:b1:cb:62:f2:18:11:93:ab:fd:16:dc:05:35:ba:
    26:52:32:ca:77
```

Figure 3.3



```
    ba:87:1b:9c:b1:96:83:16:a2:08:a5:9f:74:73:2b:
    d1:2c:eb:7f:21:98:76:7e:fd:7e:86:f9:8b:11:0c:
    36:24:80:f5:77:cf:bd:00:98:87:cf:be:54:07:58:
    2c:6b:73:03
exponent2:
    13:76:ef:dd:4e:b2:79:95:44:90:0d:6c:a3:26:22:
    09:d4:26:21:f9:3d:a5:90:81:62:63:75:23:92:5f:
    28:c5:e7:28:f0:13:bc:e8:d6:03:a3:b1:6d:b7:b4:
    f2:51:50:e0:dc:07:38:cf:78:67:e9:71:85:d1:91:
    b3:f5:24:19
coefficient:
    00:ce:0d:83:fe:e9:06:2c:ec:ed:6c:76:f7:2c:3f:
    ab:32:e5:b7:55:85:9b:fb:03:83:a4:af:b3:31:7a:
    7f:7c:56:4e:29:53:d0:3f:51:a0:55:1b:90:61:2c:
    d7:d1:ed:9a:4d:39:bd:a0:c4:a2:43:94:70:77:78:
    de:ee:80:4e:9d
writing RSA key
-----BEGIN RSA PRIVATE KEY-----
MIICXAIBAAKBgQDoOQK2O015gjrMQ0+Lh18dpve5/RHuIGwQzw5NEZeuJYhu/zvs
PSntuAQ/pCFOSKXU5QD/d5vsRw47wdY3kVJ5MQlktJzpdLm3tqSFwGn6lku1Iham
TnYpCwFC6Pl1IB/eSLSlbGc7/3QEnn/1D+yaPQ4JdwclRnKB+PYYopi2WwIDAQAB
AoGAPbC3ebWzVS41dIIFJanmqLfsY2pJUxsl2ilHQU4FH2w1HFeNkaqF8vLJniZ5
+pPuCep0I2mg8FnH/DP4NnbSk4HYv4WH018fuQEa9/PLAWQV5JvZyG6veUpv2pf3
HhHPPp00b1UlIQ1zyfOguqwEdCsI1mmw/i/zagb8IU6851ECQQD1yJUsXltKoDlo
GsWRgH1XxaFxHYu7n3H8TUWPTQKwjbQQpnNJNkJ2pLBbRlqxy2LyGBGTq/0W3AU1
uiZSMsp3AkEA8eAgDFs+oVWhs8qjtGl8jUKxLU5TPJXXLDZXhOD4SgbK10lUBaiT
Y/pr08lgTffOZo98+5FjSGoUvUMPHxtIPQJAPeBQhcn/a6soqKfMw1O4uocbnLGW
gxaiCKWfdHMr0SzrfyGYdn79fob5ixEMNiSA9XfPvQCYh8++VAdYLGtzAwJAE3bv
3U6yeZVEkA1soyYiCdQmIfk9pZCBYmN1I5JfKMXnKPATvOjWA6Oxbbe08lFQ4NwH
OM94Z+lxhdGRs/UkGQJBAM4Ng/7pBizs7Wx29yw/qzLlt1WFm/sDg6SvszF6f3xW
TilT0D9RoFUbkGEs19Htmk05vaDEokOUcHd43u6ATp0=
-----END RSA PRIVATE KEY-----
root@kali:~/Desktop/Lab4# 
```

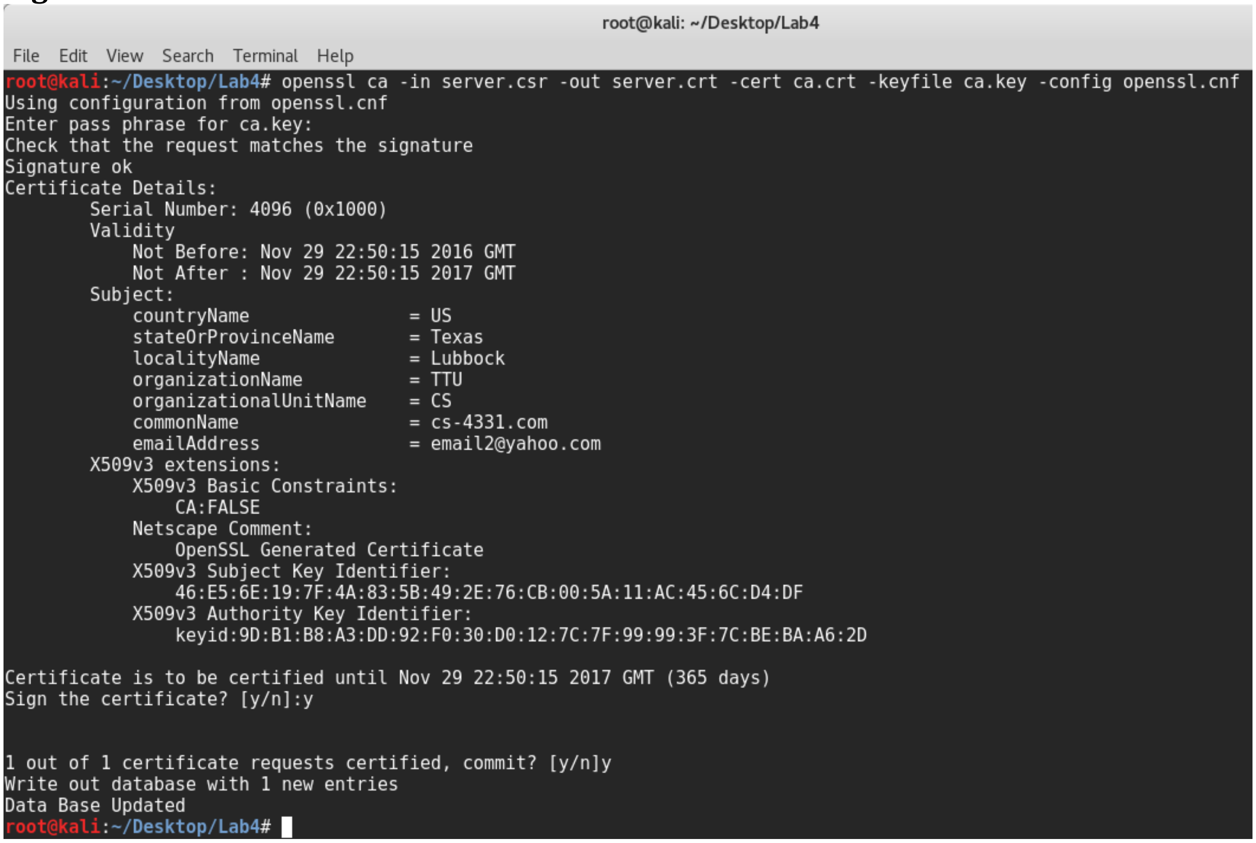
Figure 3.4


root@kali: ~/Desktop/Lab4
File Edit View Search Terminal Help

```
root@kali:~/Desktop/Lab4# ls -l
total 28
-rw-r--r-- 1 root root  1391 Nov 29 17:32 ca.crt
-rw-r--r-- 1 root root  1834 Nov 29 17:32 ca.key
drwxr-xr-x 5 root root  4096 Nov 29 17:26 demoCA
-rw-r--r-- 1 root root 10837 Nov 29 17:31 openssl.cnf
-rw-r--r-- 1 root root   986 Nov 29 17:34 server.key
root@kali:~/Desktop/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]:US
State or Province Name (full name) [Some-State]:Texas
Locality Name (eg, city) []:Lubbock
Organization Name (eg, company) [Internet Widgits Pty Ltd]:TTU
Organizational Unit Name (eg, section) []:CS
Common Name (e.g. server FQDN or YOUR name) []:cs-4331.com
Email Address []:email2@yahoo.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:123456
An optional company name []:Security
root@kali:~/Desktop/Lab4# 
```

4. **Generating the Certificate:** After the CA receives the CSR the CA will create the certificate and place it in the server certificate file. See **Figure 4.1** below. **Note:** If there is a problem when creating the certificates openssl maybe causing this due to the certain fields in your request do not match the fields of the CA. The problem can be addressed by changing the policy or requests to match the CA. The easiest way is to change the policy to "policy_anyting" in the openssl.cnf file.

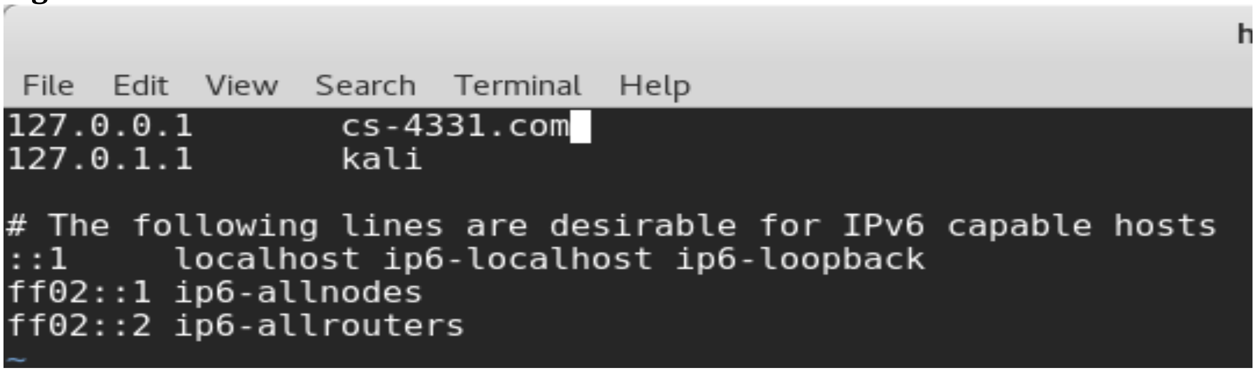
Figure 4.1


root@kali: ~/Desktop/Lab4
File Edit View Search Terminal Help

```
root@kali:~/Desktop/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: 4096 (0x1000)
        Validity
            Not Before: Nov 29 22:50:15 2016 GMT
            Not After : Nov 29 22:50:15 2017 GMT
        Subject:
            countryName               = US
            stateOrProvinceName       = Texas
            localityName              = Lubbock
            organizationName          = TTU
            organizationalUnitName    = CS
            commonName                = cs-4331.com
            emailAddress              = email2@yahoo.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                46:E5:6E:19:7F:4A:83:5B:49:2E:76:CB:00:5A:11:AC:45:6C:D4:DF
            X509v3 Authority Key Identifier:
                keyid:9D:B1:B8:A3:DD:92:F0:30:D0:12:7C:7F:99:99:3F:7C:BE:BA:A6:2D

Certificate is to be certified until Nov 29 22:50:15 2017 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
root@kali:~/Desktop/Lab4# 
```

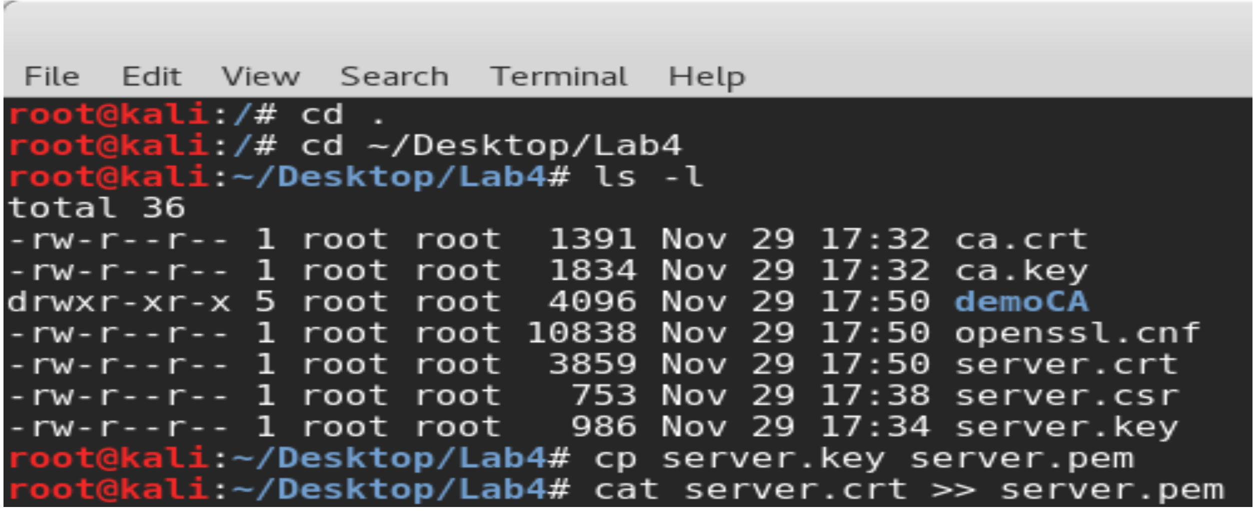
5. **Preparing CS-4331.com to use the certificate:** First I added cs-4331.com to the host name in /etc/host. Allowing the system to tell the client that cs-4331.com is located at 127.0.0.1. See **Figure 5.1** below. Then I prepared to launch a simple web server with the certificate generated in the previous steps. Before launching the web server I combined the secret key and the certificate into one file called server.pem. See **Figure 5.2** below. After combining the keys I then launched the web server. I was unable to view the webpage cs-4331.com because the CA's certificate was not in the browser's list of certificates. See **Figure 5.3** below. To be able to view the webpage I added the CA certificate to the certificates already accepted by the browser by importing the ca.crt and selecting the option "Trust this CA to identify websites." See **Figure 5.4** below. I then browse to the webpage again to view the client's secured website. See **Figure 5.5** below.

Figure 5.1



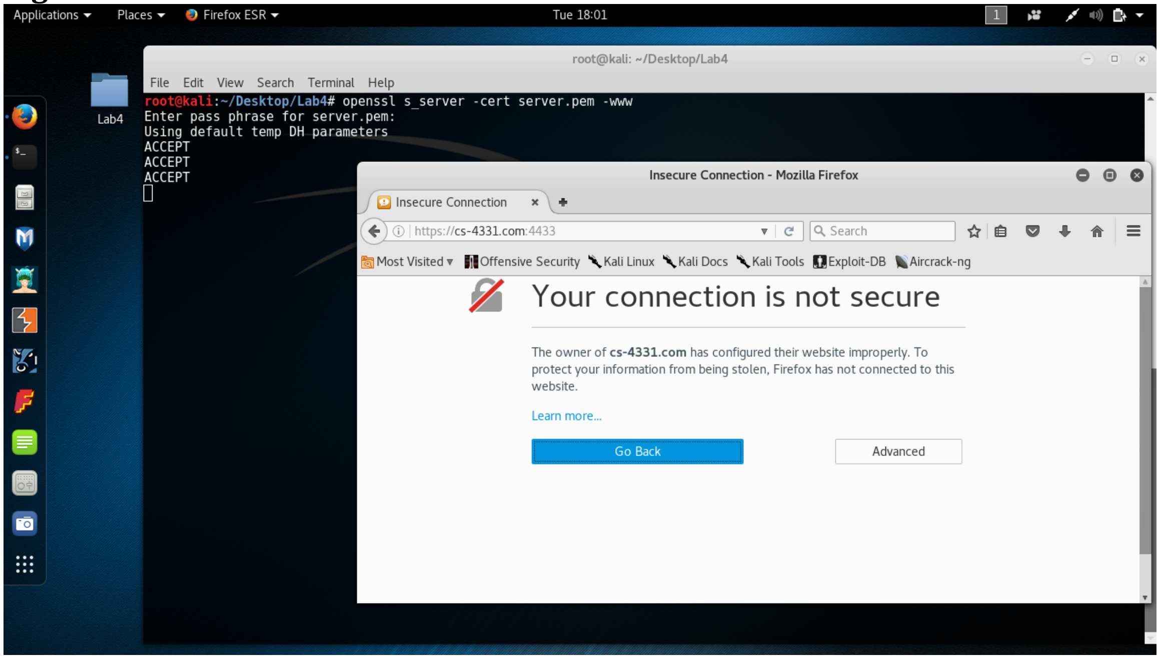
```
127.0.0.1       cs-4331.com
127.0.1.1       kali

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
~
```

Figure 5.2



```
root@kali:/# cd .
root@kali:/# cd ~/Desktop/Lab4
root@kali:~/Desktop/Lab4# ls -l
total 36
-rw-r--r-- 1 root root  1391 Nov 29 17:32 ca.crt
-rw-r--r-- 1 root root  1834 Nov 29 17:32 ca.key
drwxr-xr-x 5 root root  4096 Nov 29 17:50 demoCA
-rw-r--r-- 1 root root 10838 Nov 29 17:50 openssl.cnf
-rw-r--r-- 1 root root  3859 Nov 29 17:50 server.crt
-rw-r--r-- 1 root root   753 Nov 29 17:38 server.csr
-rw-r--r-- 1 root root   986 Nov 29 17:34 server.key
root@kali:~/Desktop/Lab4# cp server.key server.pem
root@kali:~/Desktop/Lab4# cat server.crt >> server.pem
```

Figure 5.3



Applications
Places
Firefox ESR
Tue 18:01
Lab4
root@kali: ~/Desktop/Lab4
File Edit View Search Terminal Help

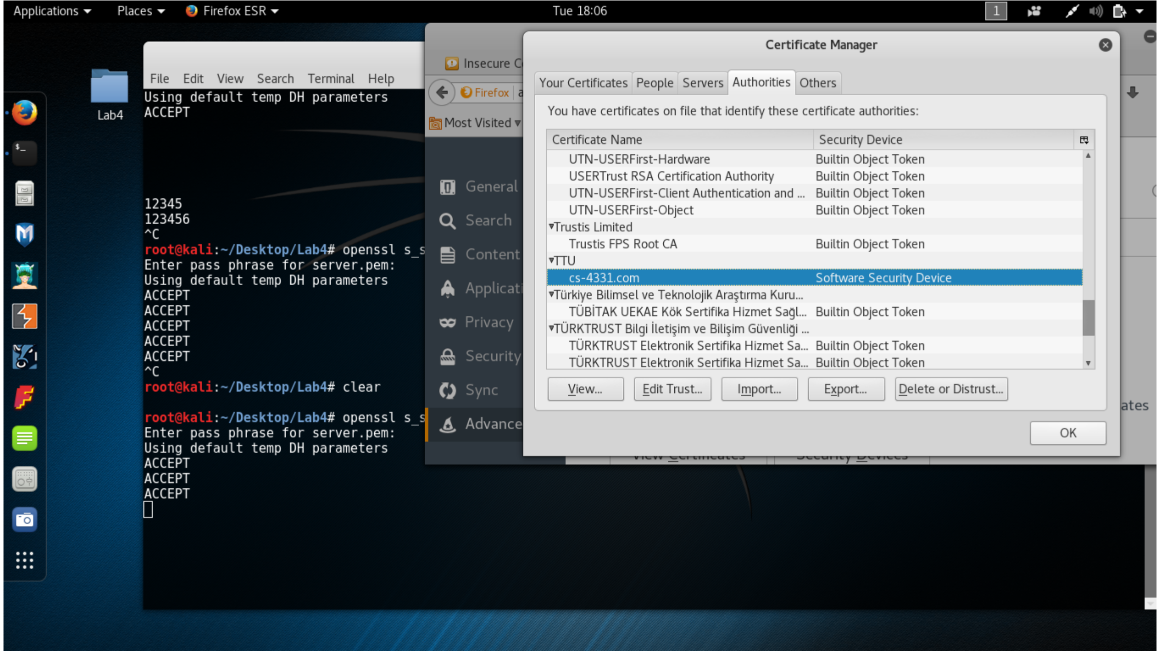
```
root@kali:~/Desktop/Lab4# openssl s_server -cert server.pem -www
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
ACCEPT
ACCEPT
```

Your connection is not secure

The owner of **cs-4331.com** has configured their website improperly. To protect your information from being stolen, Firefox has not connected to this website.

Learn more…

Figure 5.4

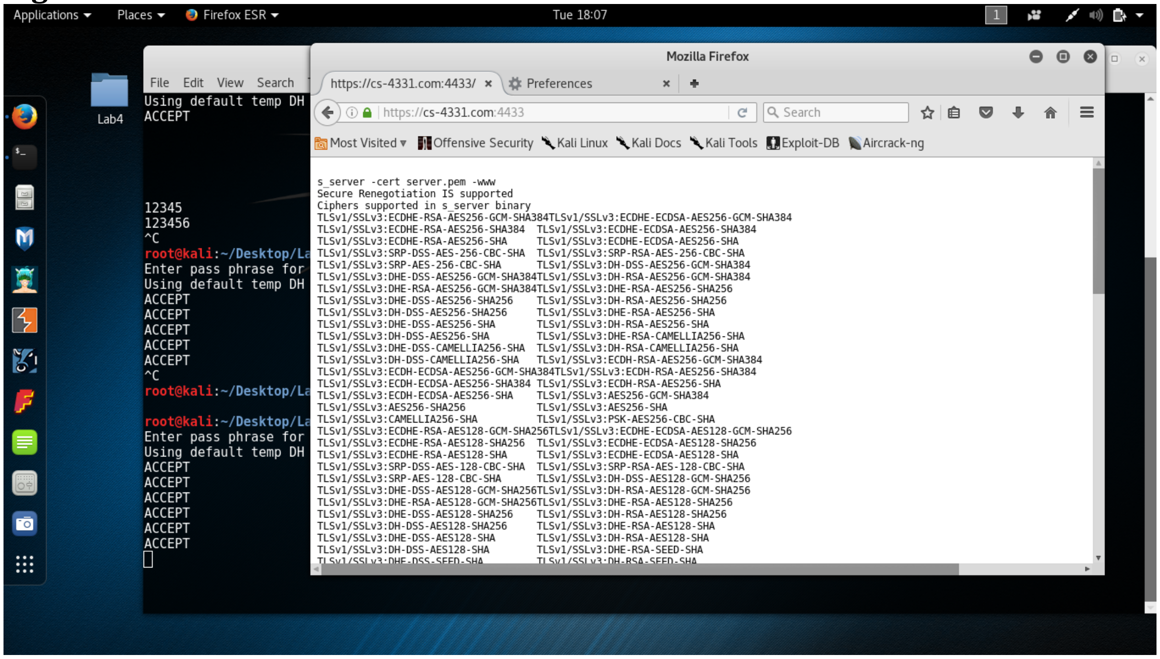


```
Using default temp DH parameters
ACCEPT




12345
123456
^C
root@kali:~/Desktop/Lab4# openssl s_s
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
ACCEPT
ACCEPT
ACCEPT
ACCEPT
^C
root@kali:~/Desktop/Lab4# clear

root@kali:~/Desktop/Lab4# openssl s_s
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
ACCEPT
ACCEPT
```

You have certificates on file that identify these certificate authorities:

Certificate Name	Security Device
UTN-USERFirst-Hardware	Builtin Object Token
USERTrust RSA Certification Authority	Builtin Object Token
UTN-USERFirst-Client Authentication and …	Builtin Object Token
UTN-USERFirst-Object	Builtin Object Token
▾Trustis Limited	
Trustis FPS Root CA	Builtin Object Token
▾TTU	
cs-4331.com	Software Security Device
▾Türkiye Bilimsel ve Teknolojik Araştırma Kuru…	
TÜBİTAK UEKAE Kök Sertifika Hizmet Sağl…	Builtin Object Token
▾TÜRKTRUST Bilgi İletişim ve Bilişim Güvenliği …	
TÜRKTRUST Elektronik Sertifika Hizmet Sa…	Builtin Object Token
TÜRKTRUST Elektronik Sertifika Hizmet Sa…	Builtin Object Token

Figure 5.5



```
Using default temp DH
ACCEPT



12345
123456
^C
root@kali:~/Desktop/La
Enter pass phrase for
Using default temp DH
ACCEPT
ACCEPT
ACCEPT
ACCEPT
ACCEPT
^C
root@kali:~/Desktop/La

root@kali:~/Desktop/La
Enter pass phrase for
Using default temp DH
ACCEPT
ACCEPT
ACCEPT
ACCEPT
ACCEPT
ACCEPT
```

```
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
TLSv1/SSLv3:DHE-DSS-AES128-SHA       TLSv1/SSLv3:DH-RSA-AES128-SHA
TLSv1/SSLv3:DH-DSS-AES128-SHA        TLSv1/SSLv3:DHE-RSA-SEED-SHA
TLSv1/SSLv3:DHE-DSS-SEED-SHA         TLSv1/SSLv3:DH-RSA-SEED-SHA
```

Conclusion:

In conclusion the browser will now display the client's secured website because it has the CA certificate in its list of trusted certificates. The browser trusts the CA certificate to identify websites. The browser follows the trail of certificates starting with the clients certificate, seeing that the client certificate is signed by the CA certificate which is a CA trusted by the browser to identify websites, the browser will display the client secured website.