Public-Key Cryptography and PKI

Name: Alizeh Jafri

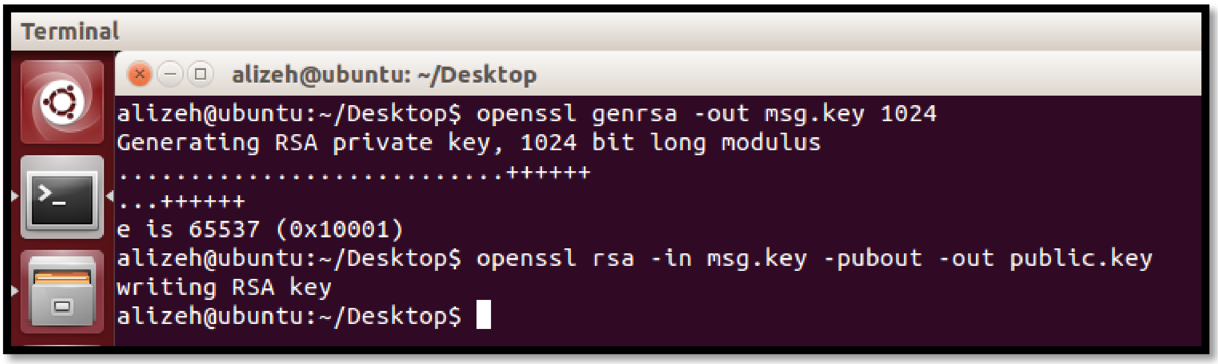
Introduction:

The objective of this lab to get familiar with the concepts in the Public-Key encryption, digital signature, and Public-Key Infrastructure (PKI).

Task 1:

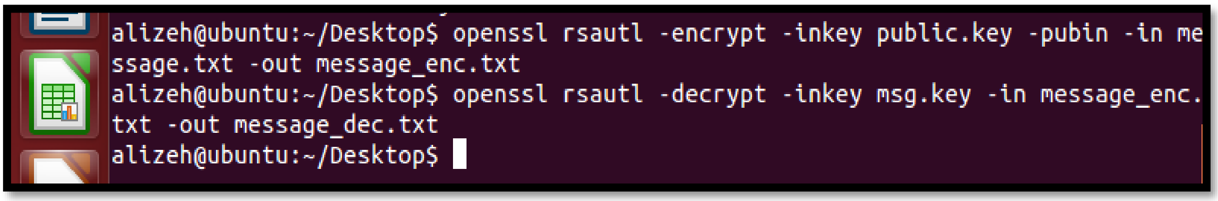
In task 1, we will get familiar with RSA encryption in openssl.

A file **'message.txt'** was created that contains some random message. Also,1024-bit RSA public/ private key pair was generated. The screen shot below shows how to generate an 1024-bit RSA public and private key pair:



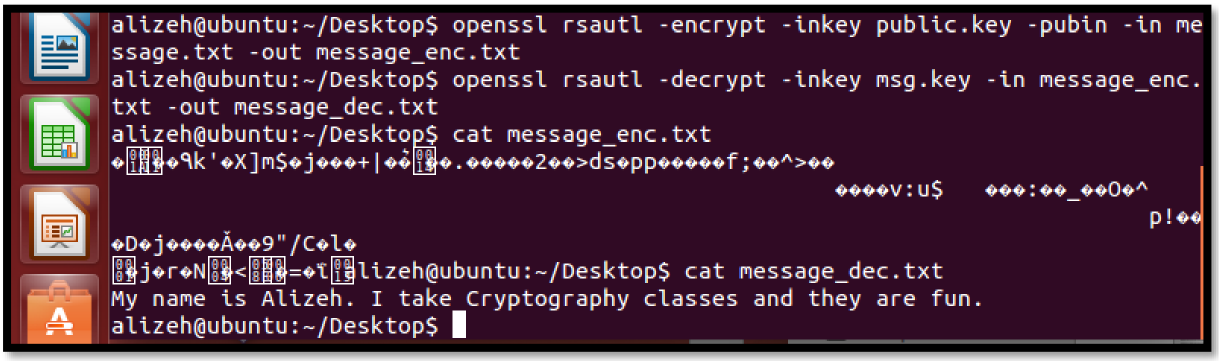
```
alizeh@ubuntu:~/Desktop$ openssl genrsa -out msg.key 1024
Generating RSA private key, 1024 bit long modulus
...........................++++++
...++++++
e is 65537 (0x10001)
alizeh@ubuntu:~/Desktop$ openssl rsa -in msg.key -pubout -out public.key
writing RSA key
alizeh@ubuntu:~/Desktop$ 
```

After getting the public key and private key, the file **'message.txt'** was encrypted using the public key and the output was saved in **'message_enc.txt'**. Next, the file **'message_enc.txt'** was decrypted using the private key as shown in the screen shot below:



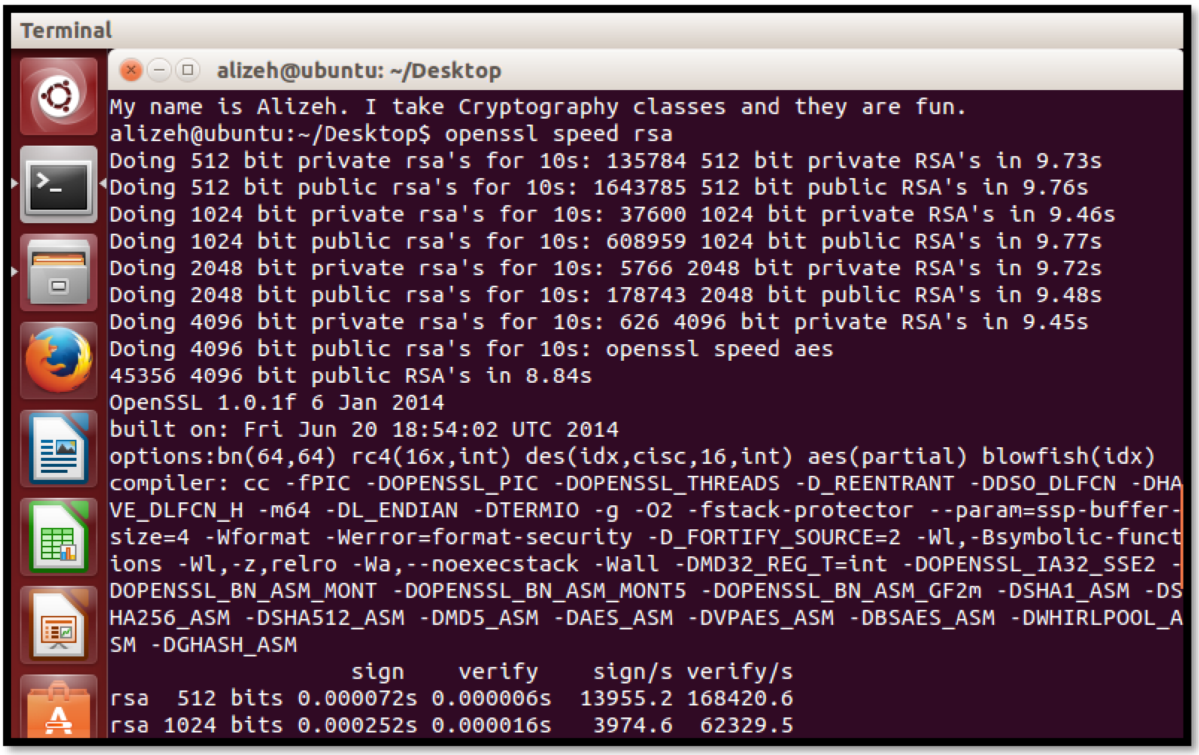
```
alizeh@ubuntu:~/Desktop$ openssl rsautl -encrypt -inkey public.key -pubin -in me
ssage.txt -out message_enc.txt
alizeh@ubuntu:~/Desktop$ openssl rsautl -decrypt -inkey msg.key -in message_enc.
txt -out message_dec.txt
alizeh@ubuntu:~/Desktop$ 
```

The encrypted and decrypted text can be seen in the screenshot below:



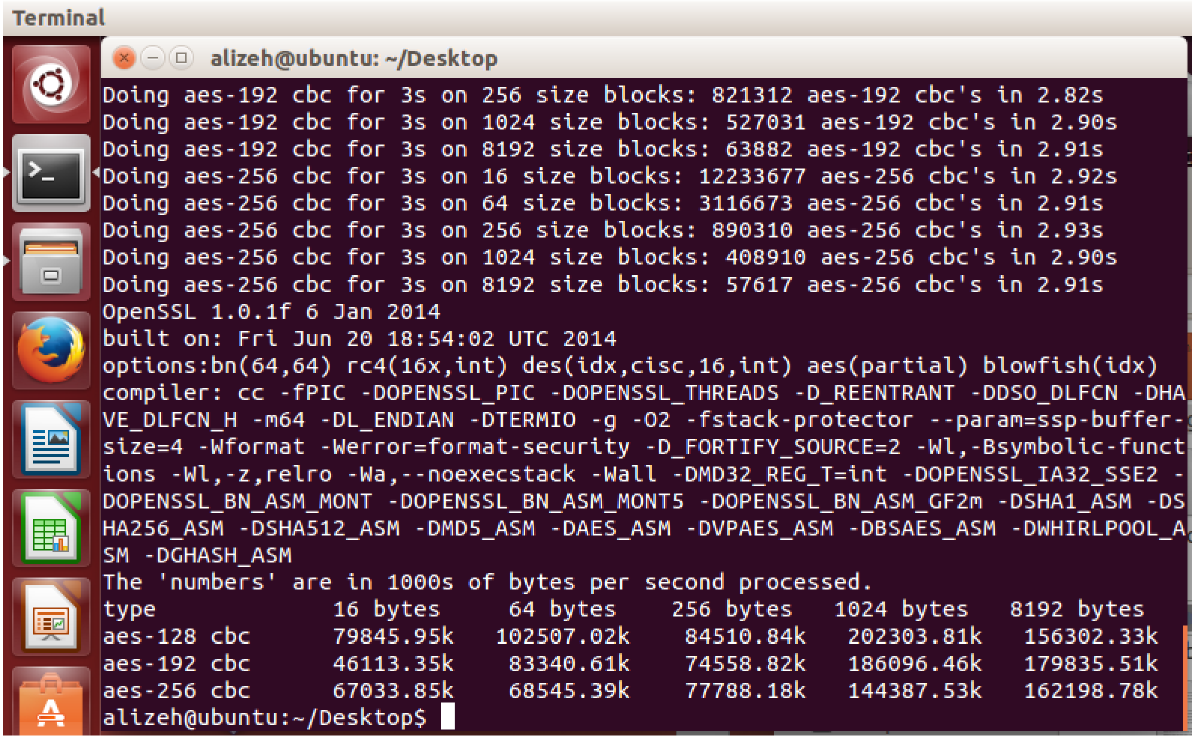
```
alizeh@ubuntu:~/Desktop$ openssl rsautl -encrypt -inkey public.key -pubin -in me
ssage.txt -out message_enc.txt
alizeh@ubuntu:~/Desktop$ openssl rsautl -decrypt -inkey msg.key -in message_enc.
txt -out message_dec.txt
alizeh@ubuntu:~/Desktop$ cat message_enc.txt
�□□��9k'�X]m$�j���+|��□�.�����2��>ds�pp�����f;��^>��
                                                   ����v:u$   ���:��_��O�^
                                                                        p!��
�D�j����Ă��9"/C�l�
□�j�r�N□�<□□�=�ĭ□lizeh@ubuntu:~/Desktop$ cat message_dec.txt
My name is Alizeh. I take Cryptography classes and they are fun.
alizeh@ubuntu:~/Desktop$ 
```

a) Openssl speed rsa (screen shot shown below)



```
My name is Alizeh. I take Cryptography classes and they are fun.
alizeh@ubuntu:~/Desktop$ openssl speed rsa
Doing 512 bit private rsa's for 10s: 135784 512 bit private RSA's in 9.73s
Doing 512 bit public rsa's for 10s: 1643785 512 bit public RSA's in 9.76s
Doing 1024 bit private rsa's for 10s: 37600 1024 bit private RSA's in 9.46s
Doing 1024 bit public rsa's for 10s: 608959 1024 bit public RSA's in 9.77s
Doing 2048 bit private rsa's for 10s: 5766 2048 bit private RSA's in 9.72s
Doing 2048 bit public rsa's for 10s: 178743 2048 bit public RSA's in 9.48s
Doing 4096 bit private rsa's for 10s: 626 4096 bit private RSA's in 9.45s
Doing 4096 bit public rsa's for 10s: openssl speed aes
45356 4096 bit public RSA's in 8.84s
OpenSSL 1.0.1f 6 Jan 2014
built on: Fri Jun 20 18:54:02 UTC 2014
options:bn(64,64) rc4(16x,int) des(idx,cisc,16,int) aes(partial) blowfish(idx)
compiler: cc -fPIC -DOPENSSL_PIC -DOPENSSL_THREADS -D_REENTRANT -DDSO_DLFCN -DHA
VE_DLFCN_H -m64 -DL_ENDIAN -DTERMIO -g -O2 -fstack-protector --param=ssp-buffer-
size=4 -Wformat -Werror=format-security -D_FORTIFY_SOURCE=2 -Wl,-Bsymbolic-funct
ions -Wl,-z,relro -Wa,--noexecstack -Wall -DMD32_REG_T=int -DOPENSSL_IA32_SSE2 -
DOPENSSL_BN_ASM_MONT -DOPENSSL_BN_ASM_MONT5 -DOPENSSL_BN_ASM_GF2m -DSHA1_ASM -DS
HA256_ASM -DSHA512_ASM -DMD5_ASM -DAES_ASM -DVPAES_ASM -DBSAES_ASM -DWHIRLPOOL_A
SM -DGHASH_ASM
                  sign    verify    sign/s verify/s
rsa  512 bits 0.000072s 0.000006s  13955.2 168420.6
rsa 1024 bits 0.000252s 0.000016s   3974.6  62329.5
```

b) Openssl speed aes (screen shot shown below)

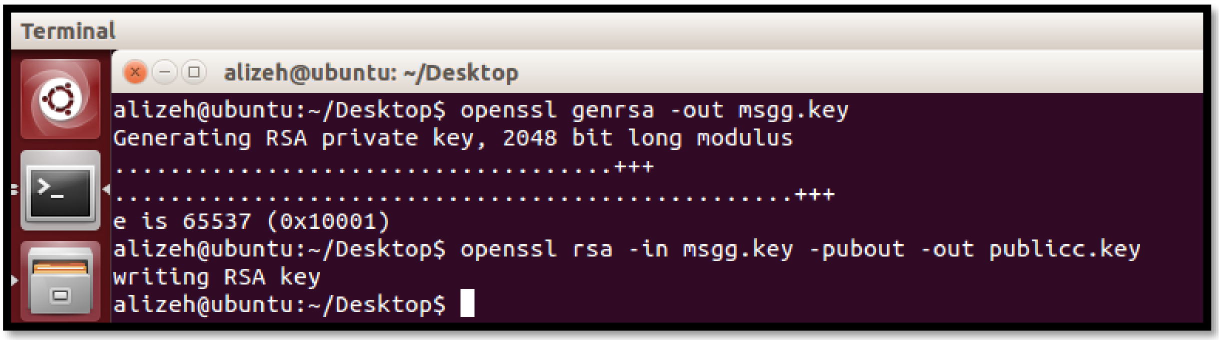


```
Doing aes-192 cbc for 3s on 256 size blocks: 821312 aes-192 cbc's in 2.82s
Doing aes-192 cbc for 3s on 1024 size blocks: 527031 aes-192 cbc's in 2.90s
Doing aes-192 cbc for 3s on 8192 size blocks: 63882 aes-192 cbc's in 2.91s
Doing aes-256 cbc for 3s on 16 size blocks: 12233677 aes-256 cbc's in 2.92s
Doing aes-256 cbc for 3s on 64 size blocks: 3116673 aes-256 cbc's in 2.91s
Doing aes-256 cbc for 3s on 256 size blocks: 890310 aes-256 cbc's in 2.93s
Doing aes-256 cbc for 3s on 1024 size blocks: 408910 aes-256 cbc's in 2.90s
Doing aes-256 cbc for 3s on 8192 size blocks: 57617 aes-256 cbc's in 2.91s
OpenSSL 1.0.1f 6 Jan 2014
built on: Fri Jun 20 18:54:02 UTC 2014
options:bn(64,64) rc4(16x,int) des(idx,cisc,16,int) aes(partial) blowfish(idx)
compiler: cc -fPIC -DOPENSSL_PIC -DOPENSSL_THREADS -D_REENTRANT -DDSO_DLFCN -DHA
VE_DLFCN_H -m64 -DL_ENDIAN -DTERMIO -g -O2 -fstack-protector --param=ssp-buffer-
size=4 -Wformat -Werror=format-security -D_FORTIFY_SOURCE=2 -Wl,-Bsymbolic-funct
ions -Wl,-z,relro -Wa,--noexecstack -Wall -DMD32_REG_T=int -DOPENSSL_IA32_SSE2 -
DOPENSSL_BN_ASM_MONT -DOPENSSL_BN_ASM_MONT5 -DOPENSSL_BN_ASM_GF2m -DSHA1_ASM -DS
HA256_ASM -DSHA512_ASM -DMD5_ASM -DAES_ASM -DVPAES_ASM -DBSAES_ASM -DWHIRLPOOL_A
SM -DGHASH_ASM
The 'numbers' are in 1000s of bytes per second processed.
type             16 bytes     64 bytes    256 bytes   1024 bytes   8192 bytes
aes-128 cbc      79845.95k   102507.02k    84510.84k   202303.81k   156302.33k
aes-192 cbc      46113.35k    83340.61k    74558.82k   186096.46k   179835.51k
aes-256 cbc      67033.85k    68545.39k    77788.18k   144387.53k   162198.78k
alizeh@ubuntu:~/Desktop$
```

My observation was that the speed for **aes** was faster than that of **rsa.**

Task 2

In task 2, OpenSSL will be used to generate digital signatures. Firstly, RSA public and private key pair was prepared as shown in the screen shot below:

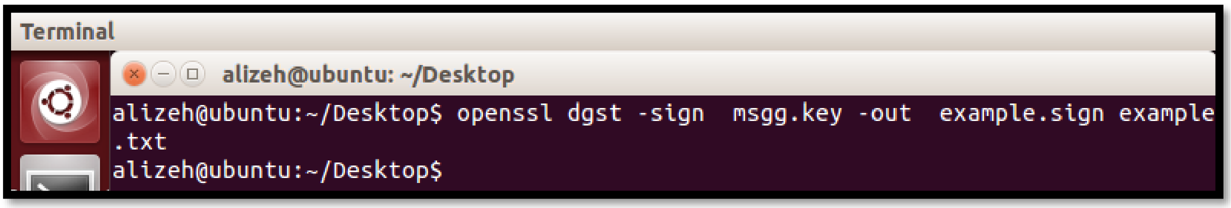


Terminal

alizeh@ubuntu: ~/Desktop

```
alizeh@ubuntu:~/Desktop$ openssl genrsa -out msgg.key
Generating RSA private key, 2048 bit long modulus
.....................................+++
..................................................+++
e is 65537 (0x10001)
alizeh@ubuntu:~/Desktop$ openssl rsa -in msgg.key -pubout -out publicc.key
writing RSA key
alizeh@ubuntu:~/Desktop$
```

1) Next, the file **example.txt** was signed and the output was saved in **example.sign**. The screen shot below shows the command used to perform this step.

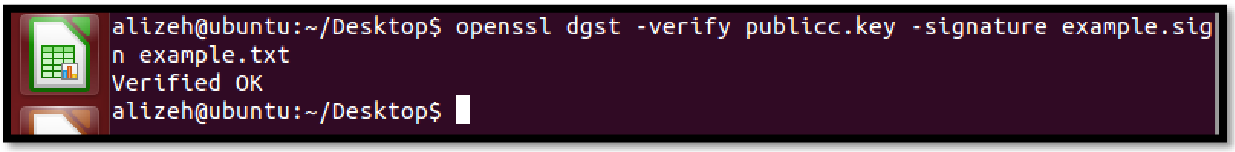


Terminal

alizeh@ubuntu: ~/Desktop

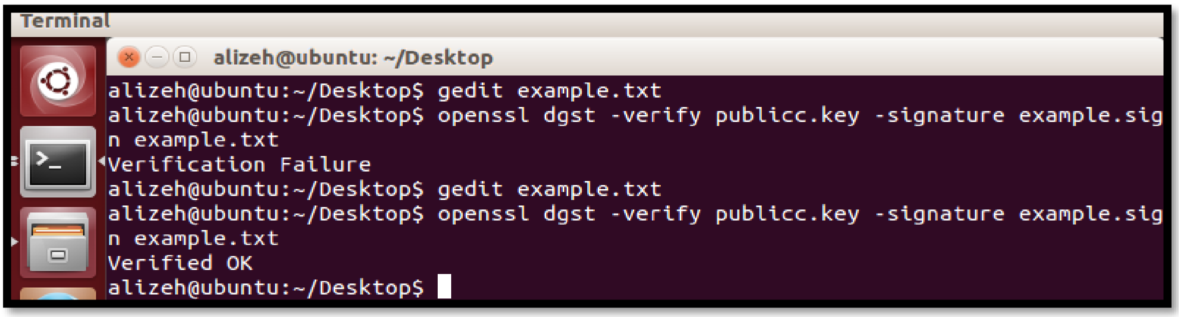
```
alizeh@ubuntu:~/Desktop$ openssl dgst -sign  msgg.key -out  example.sign example
.txt
alizeh@ubuntu:~/Desktop$
```

2) The command '**openssl dgst -verify publicc.key -signature example.sign example.txt**' was used to Verify the digital signature in example.sign.



```
alizeh@ubuntu:~/Desktop$ openssl dgst -verify publicc.key -signature example.sig
n example.txt
Verified OK
alizeh@ubuntu:~/Desktop$
```

3) Next, I slightly modified example.txt, and tried to verify the digital signature again. That gave did not verify and 'Verification Failure' appeared. This was because 'example.txt' file was modified after the signature which did not verify it. Moreover, when the file 'example.txt' was changed to the original file by eliminating the changes done, 'Verified OK' appeared. This proves that if any changes will be made after the signature, it will not be verified as shown in the screen shot below:



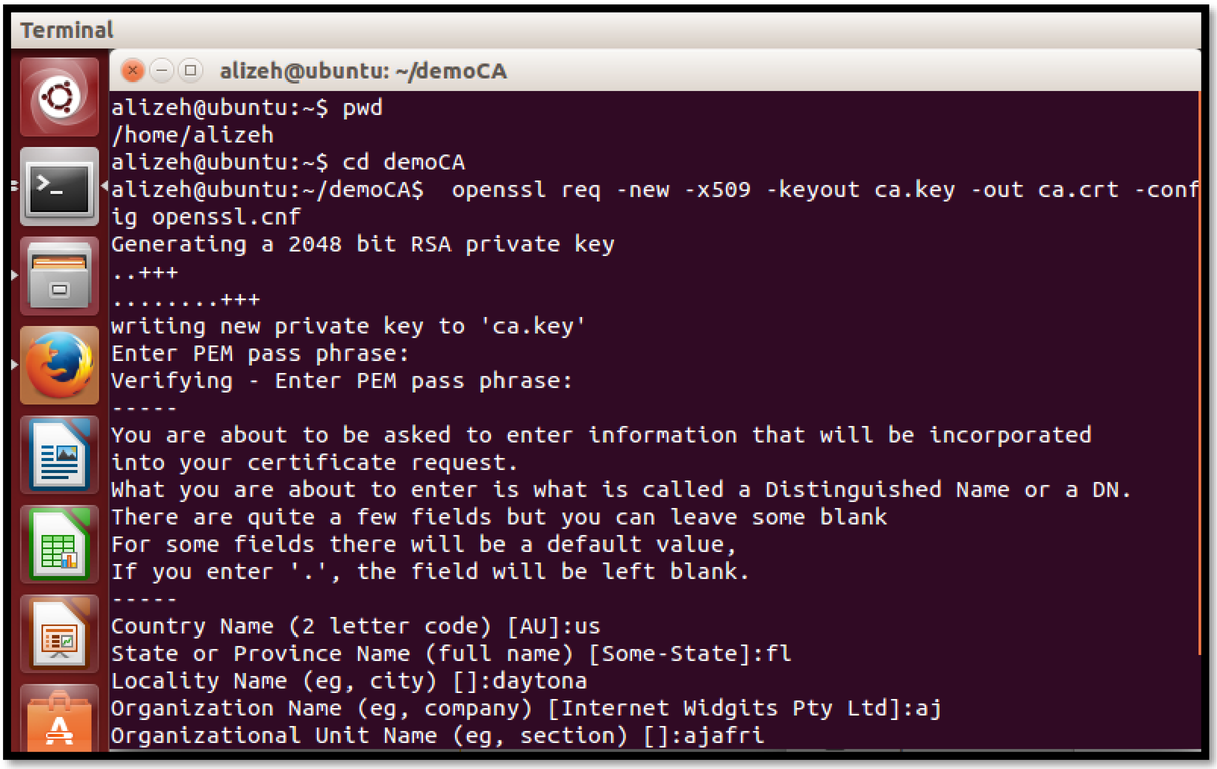
Terminal

alizeh@ubuntu: ~/Desktop

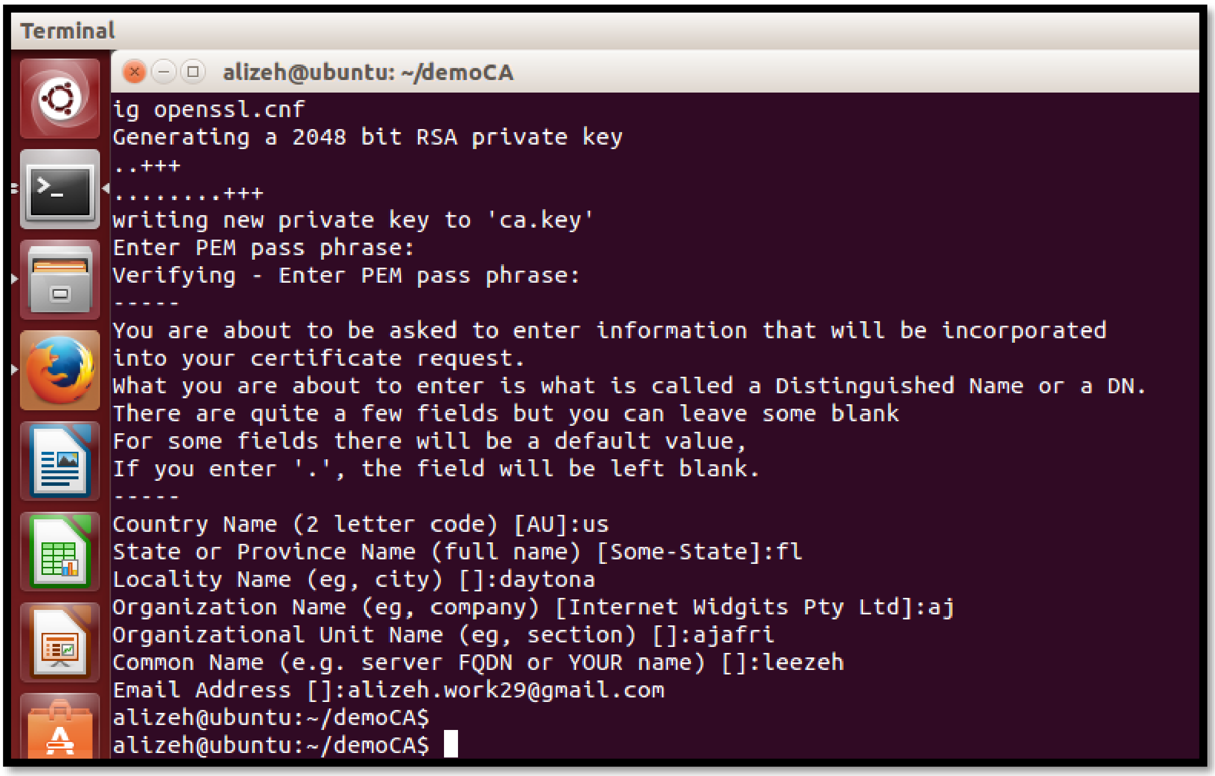
```
alizeh@ubuntu:~/Desktop$ gedit example.txt
alizeh@ubuntu:~/Desktop$ openssl dgst -verify publicc.key -signature example.sig
n example.txt
Verification Failure
alizeh@ubuntu:~/Desktop$ gedit example.txt
alizeh@ubuntu:~/Desktop$ openssl dgst -verify publicc.key -signature example.sig
n example.txt
Verified OK
alizeh@ubuntu:~/Desktop$
```

Task 3

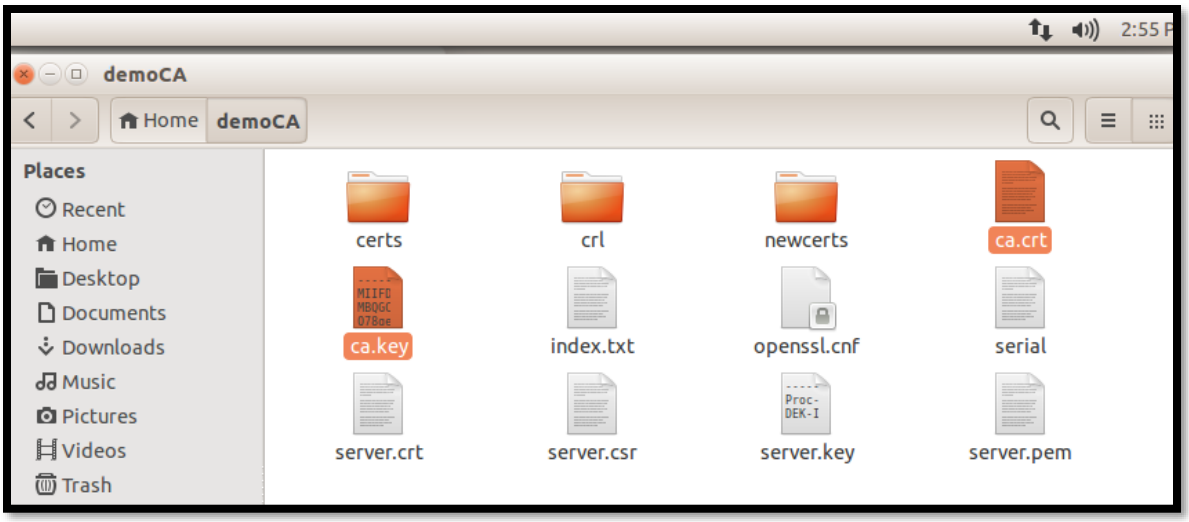
This task is for practice. Screen shots are shown below for the practice:

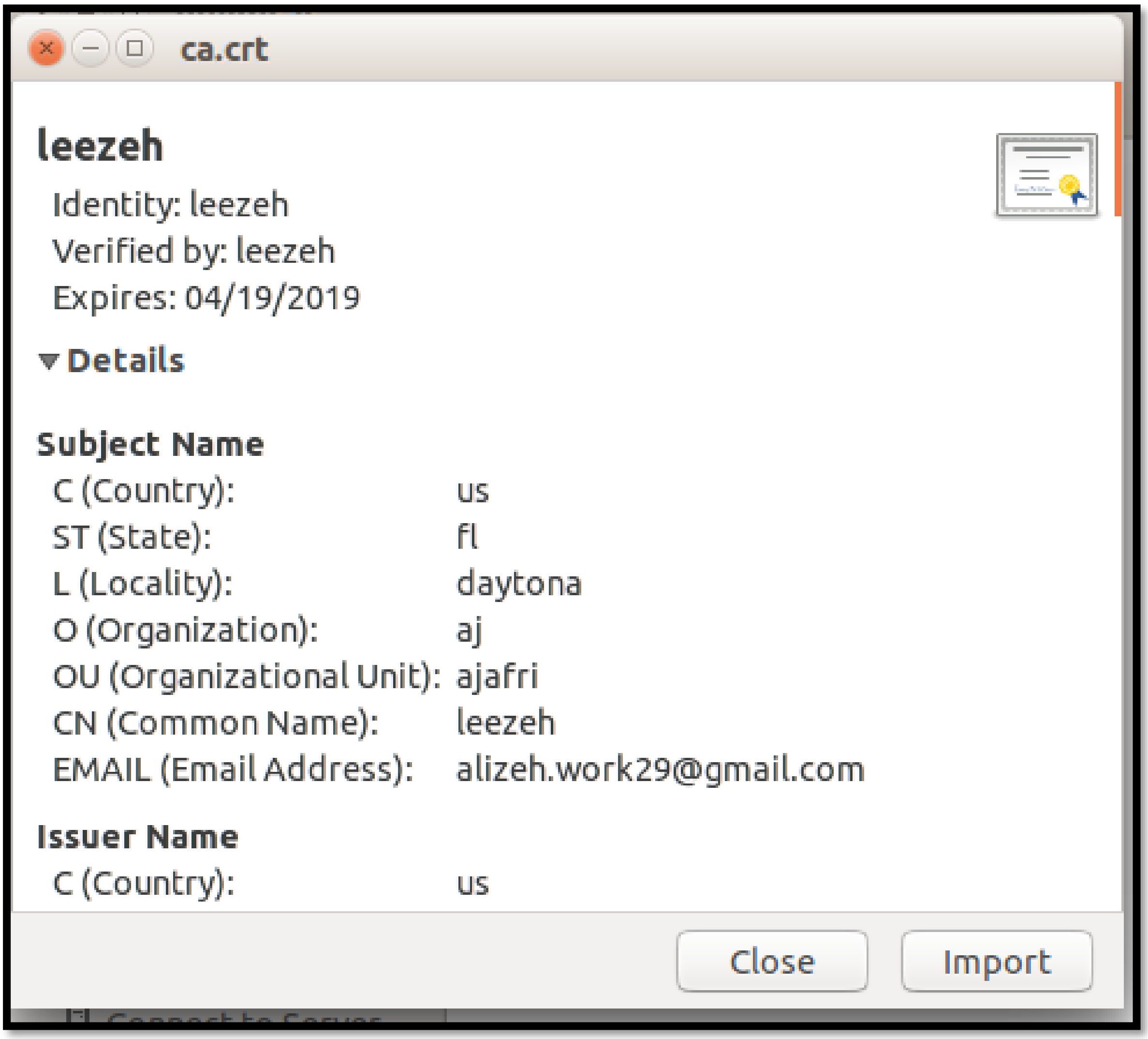


```
alizeh@ubuntu:~$ pwd
/home/alizeh
alizeh@ubuntu:~$ cd demoCA
alizeh@ubuntu:~/demoCA$  openssl req -new -x509 -keyout ca.key -out ca.crt -conf
ig openssl.cnf
Generating a 2048 bit RSA private key
..+++
........+++
writing new private key to 'ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass 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]:fl
Locality Name (eg, city) []:daytona
Organization Name (eg, company) [Internet Widgits Pty Ltd]:aj
Organizational Unit Name (eg, section) []:ajafri
```



```
ig 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]:fl
Locality Name (eg, city) []:daytona
Organization Name (eg, company) [Internet Widgits Pty Ltd]:aj
Organizational Unit Name (eg, section) []:ajafri
Common Name (e.g. server FQDN or YOUR name) []:leezeh
Email Address []:alizeh.work29@gmail.com
alizeh@ubuntu:~/demoCA$
alizeh@ubuntu:~/demoCA$ 
```





leezeh

Identity: leezeh
Verified by: leezeh
Expires: 04/19/2019

▼ **Details**

Subject Name

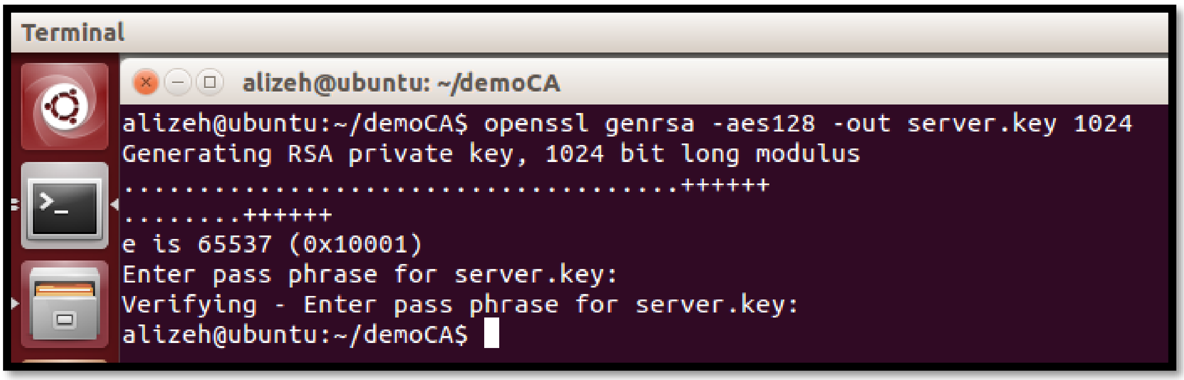
C (Country):	us
ST (State):	fl
L (Locality):	daytona
O (Organization):	aj
OU (Organizational Unit):	ajafri
CN (Common Name):	leezeh
EMAIL (Email Address):	alizeh.work29@gmail.com

Issuer Name

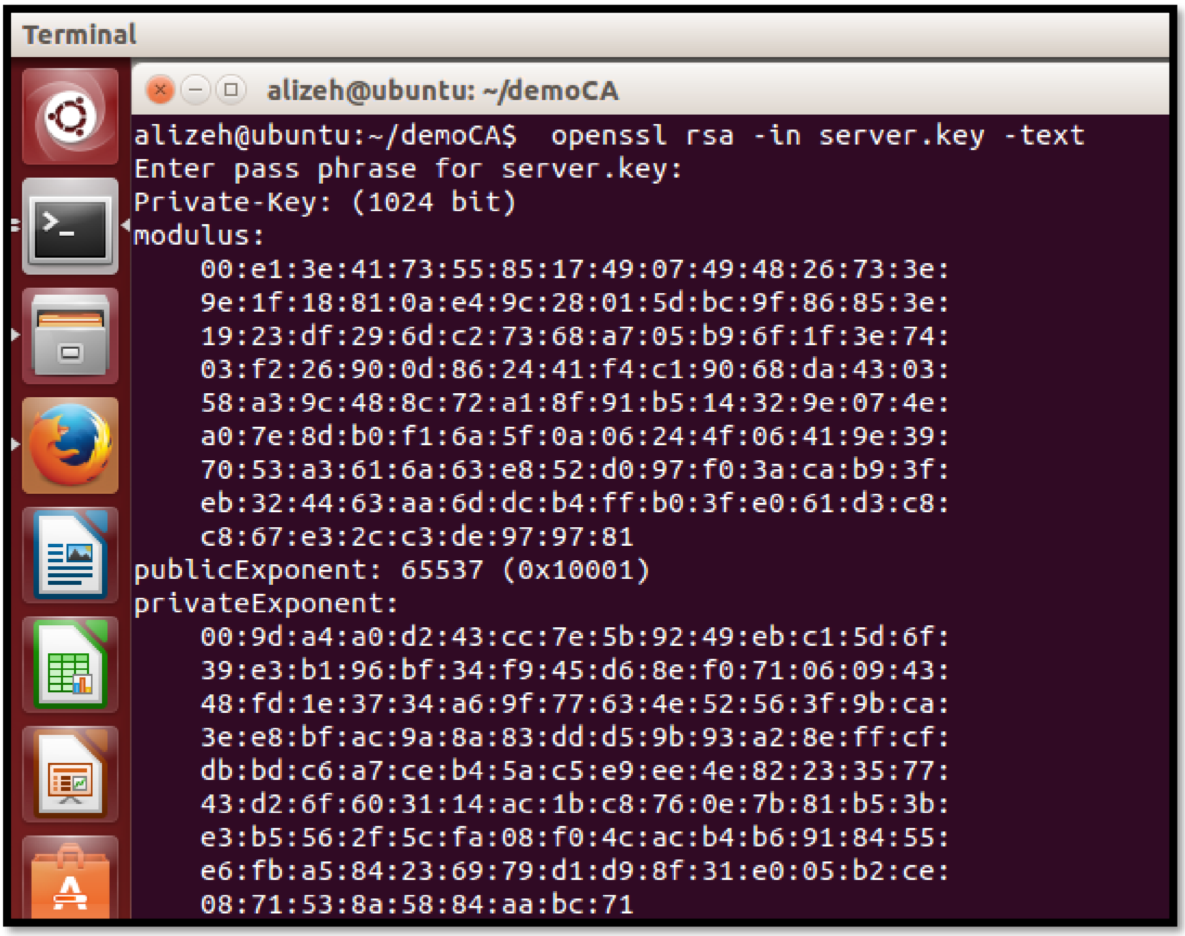
C (Country):	us

Task 4

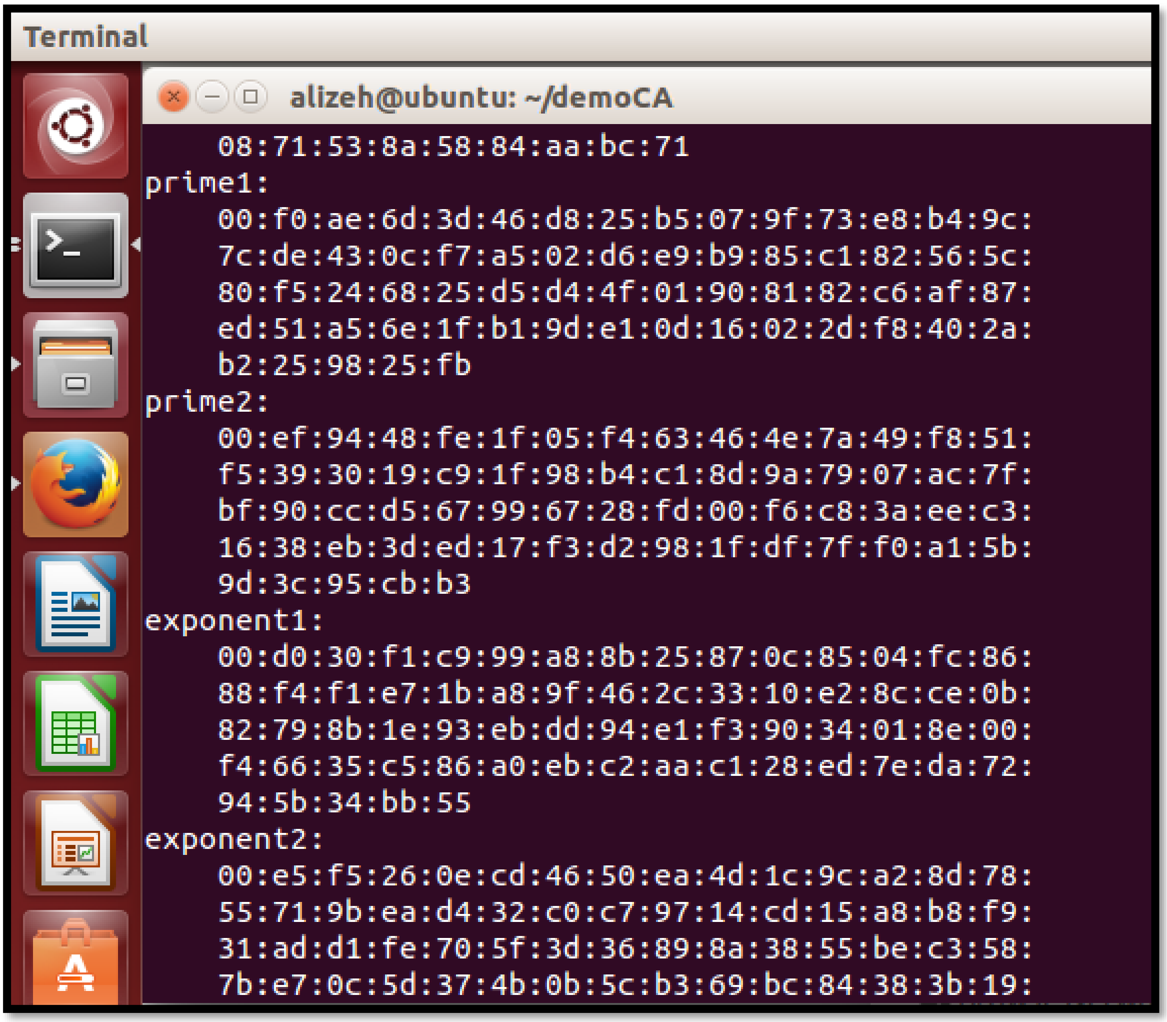
This task is also for the practice. Screen shot shown below:



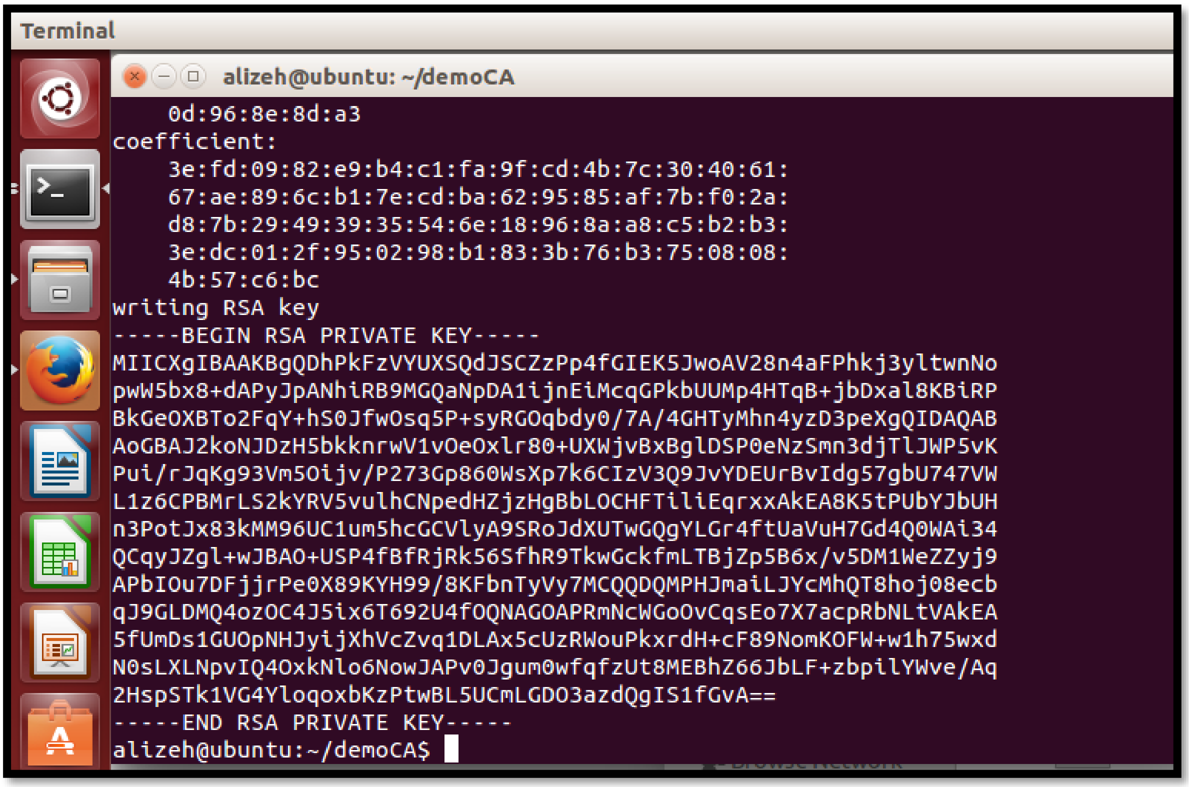
```
alizeh@ubuntu:~/demoCA$ openssl genrsa -aes128 -out server.key 1024
Generating RSA private key, 1024 bit long modulus
.....................................++++++
........++++++
e is 65537 (0x10001)
Enter pass phrase for server.key:
Verifying - Enter pass phrase for server.key:
alizeh@ubuntu:~/demoCA$ 
```



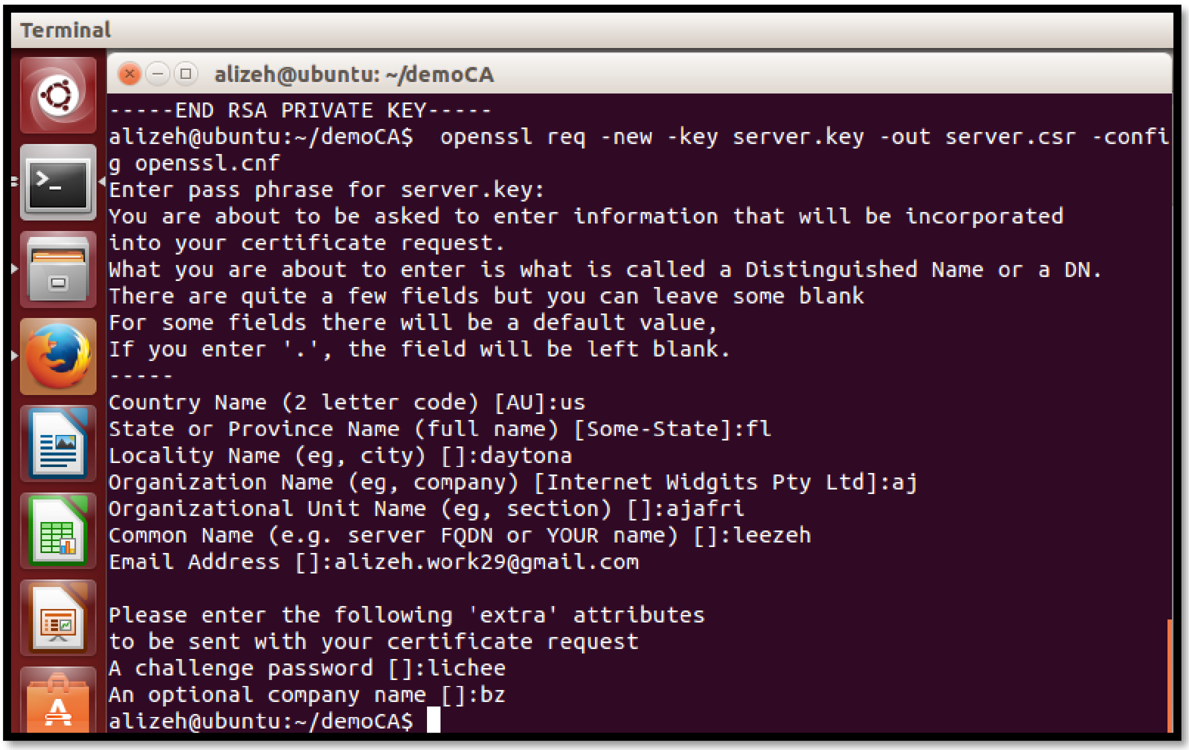
```
alizeh@ubuntu:~/demoCA$  openssl rsa -in server.key -text
Enter pass phrase for server.key:
Private-Key: (1024 bit)
modulus:
    00:e1:3e:41:73:55:85:17:49:07:49:48:26:73:3e:
    9e:1f:18:81:0a:e4:9c:28:01:5d:bc:9f:86:85:3e:
    19:23:df:29:6d:c2:73:68:a7:05:b9:6f:1f:3e:74:
    03:f2:26:90:0d:86:24:41:f4:c1:90:68:da:43:03:
    58:a3:9c:48:8c:72:a1:8f:91:b5:14:32:9e:07:4e:
    a0:7e:8d:b0:f1:6a:5f:0a:06:24:4f:06:41:9e:39:
    70:53:a3:61:6a:63:e8:52:d0:97:f0:3a:ca:b9:3f:
    eb:32:44:63:aa:6d:dc:b4:ff:b0:3f:e0:61:d3:c8:
    c8:67:e3:2c:c3:de:97:97:81
publicExponent: 65537 (0x10001)
privateExponent:
    00:9d:a4:a0:d2:43:cc:7e:5b:92:49:eb:c1:5d:6f:
    39:e3:b1:96:bf:34:f9:45:d6:8e:f0:71:06:09:43:
    48:fd:1e:37:34:a6:9f:77:63:4e:52:56:3f:9b:ca:
    3e:e8:bf:ac:9a:8a:83:dd:d5:9b:93:a2:8e:ff:cf:
    db:bd:c6:a7:ce:b4:5a:c5:e9:ee:4e:82:23:35:77:
    43:d2:6f:60:31:14:ac:1b:c8:76:0e:7b:81:b5:3b:
    e3:b5:56:2f:5c:fa:08:f0:4c:ac:b4:b6:91:84:55:
    e6:fb:a5:84:23:69:79:d1:d9:8f:31:e0:05:b2:ce:
    08:71:53:8a:58:84:aa:bc:71
```



```
    08:71:53:8a:58:84:aa:bc:71
prime1:
    00:f0:ae:6d:3d:46:d8:25:b5:07:9f:73:e8:b4:9c:
    7c:de:43:0c:f7:a5:02:d6:e9:b9:85:c1:82:56:5c:
    80:f5:24:68:25:d5:d4:4f:01:90:81:82:c6:af:87:
    ed:51:a5:6e:1f:b1:9d:e1:0d:16:02:2d:f8:40:2a:
    b2:25:98:25:fb
prime2:
    00:ef:94:48:fe:1f:05:f4:63:46:4e:7a:49:f8:51:
    f5:39:30:19:c9:1f:98:b4:c1:8d:9a:79:07:ac:7f:
    bf:90:cc:d5:67:99:67:28:fd:00:f6:c8:3a:ee:c3:
    16:38:eb:3d:ed:17:f3:d2:98:1f:df:7f:f0:a1:5b:
    9d:3c:95:cb:b3
exponent1:
    00:d0:30:f1:c9:99:a8:8b:25:87:0c:85:04:fc:86:
    88:f4:f1:e7:1b:a8:9f:46:2c:33:10:e2:8c:ce:0b:
    82:79:8b:1e:93:eb:dd:94:e1:f3:90:34:01:8e:00:
    f4:66:35:c5:86:a0:eb:c2:aa:c1:28:ed:7e:da:72:
    94:5b:34:bb:55
exponent2:
    00:e5:f5:26:0e:cd:46:50:ea:4d:1c:9c:a2:8d:78:
    55:71:9b:ea:d4:32:c0:c7:97:14:cd:15:a8:b8:f9:
    31:ad:d1:fe:70:5f:3d:36:89:8a:38:55:be:c3:58:
    7b:e7:0c:5d:37:4b:0b:5c:b3:69:bc:84:38:3b:19:
```

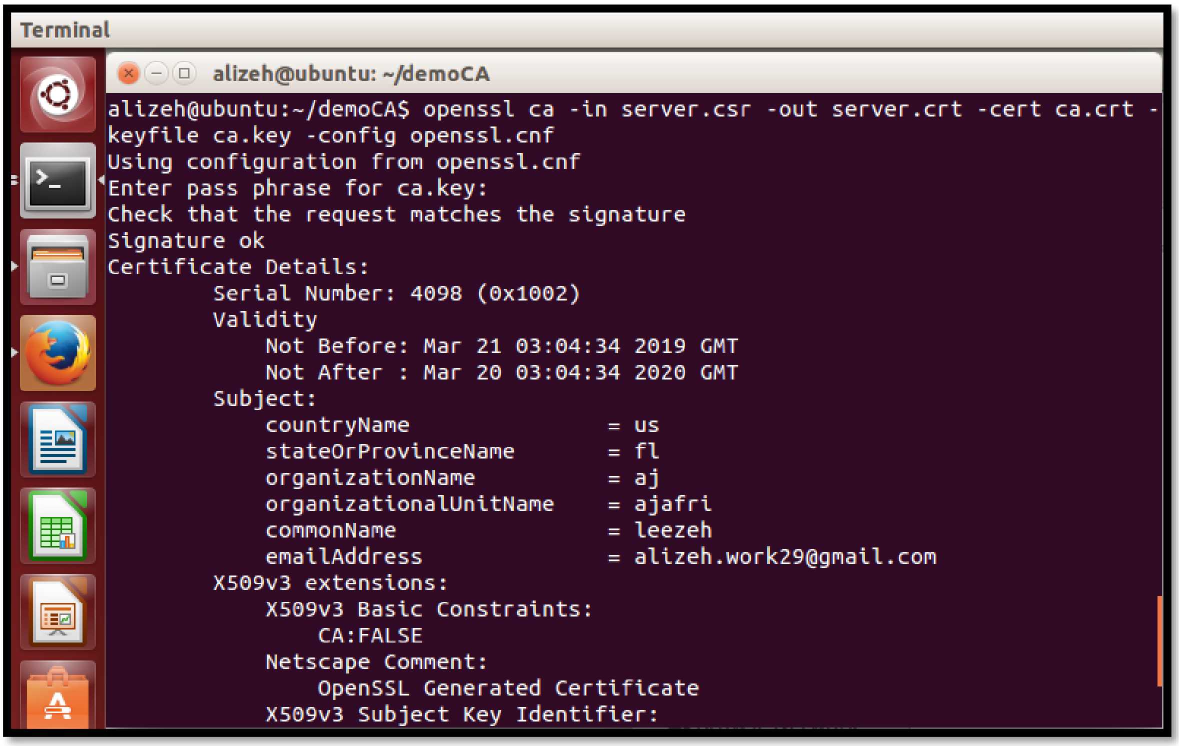


```
    0d:96:8e:8d:a3
coefficient:
    3e:fd:09:82:e9:b4:c1:fa:9f:cd:4b:7c:30:40:61:
    67:ae:89:6c:b1:7e:cd:ba:62:95:85:af:7b:f0:2a:
    d8:7b:29:49:39:35:54:6e:18:96:8a:a8:c5:b2:b3:
    3e:dc:01:2f:95:02:98:b1:83:3b:76:b3:75:08:08:
    4b:57:c6:bc
writing RSA key
-----BEGIN RSA PRIVATE KEY-----
MIICXgIBAAKBgQDhPkFzVYUXSQdJSCZzPp4fGIEK5JwoAV28n4aFPhkj3yltwnNo
pwW5bx8+dAPyJpANhiRB9MGQaNpDA1ijnEiMcqGPkbUUMp4HTqB+jbDxal8KBiRP
BkGeOXBTo2FqY+hS0JfwOsq5P+syRGOqbdy0/7A/4GHTyMhn4yzD3peXgQIDAQAB
AoGBAJ2koNJDzH5bkknrwV1vOeOxlr80+UXWjvBxBglDSP0eNzSmn3djTlJWP5vK
Pui/rJqKg93Vm5Oijv/P273Gp860WsXp7k6CIzV3Q9JvYDEUrBvIdg57gbU747VW
L1z6CPBMrLS2kYRV5vulhCNpedHZjzHgBbLOCHFTiliEqrxxAkEA8K5tPUbYJbUH
n3PotJx83kMM96UC1um5hcGCVlyA9SRoJdXUTwGQgYLGr4ftUaVuH7Gd4Q0WAi34
QCqyJZgl+wJBAO+USP4fBfRjRk56SfhR9TkwGckfmLTBjZp5B6x/v5DM1WeZZyj9
APbIOu7DFjjrPe0X89KYH99/8KFbnTyVy7MCQQDQMPHJmaiLJYcMhQT8hoj08ecb
qJ9GLDMQ4ozOC4J5ix6T692U4fOQNAGOAPRmNcWGoOvCqsEo7X7acpRbNLtVAkEA
5fUmDs1GUOpNHJyijXhVcZvq1DLAx5cUzRWouPkxrdH+cF89NomKOFW+w1h75wxd
N0sLXLNpvIQ4OxkNlo6NowJAPv0Jgum0wfqfzUt8MEBhZ66JbLF+zbpilYWve/Aq
2HspSTk1VG4YloqoxbKzPtwBL5UCmLGDO3azdQgIS1fGvA==
-----END RSA PRIVATE KEY-----
alizeh@ubuntu:~/demoCA$
```

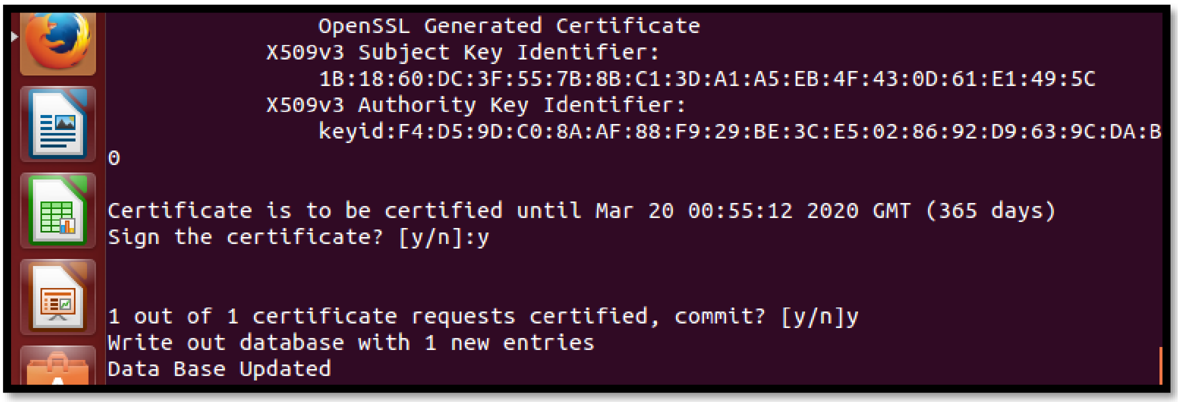


```
-----END RSA PRIVATE KEY-----
alizeh@ubuntu:~/demoCA$  openssl req -new -key server.key -out server.csr -confi
g openssl.cnf
Enter pass phrase for server.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what 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]:fl
Locality Name (eg, city) []:daytona
Organization Name (eg, company) [Internet Widgits Pty Ltd]:aj
Organizational Unit Name (eg, section) []:ajafri
Common Name (e.g. server FQDN or YOUR name) []:leezeh
Email Address []:alizeh.work29@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:lichee
An optional company name []:bz
alizeh@ubuntu:~/demoCA$
```



```
alizeh@ubuntu:~/demoCA$ 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: Mar 21 03:04:34 2019 GMT
            Not After : Mar 20 03:04:34 2020 GMT
        Subject:
            countryName               = us
            stateOrProvinceName       = fl
            organizationName          = aj
            organizationalUnitName    = ajafri
            commonName                = leezeh
            emailAddress              = alizeh.work29@gmail.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
```



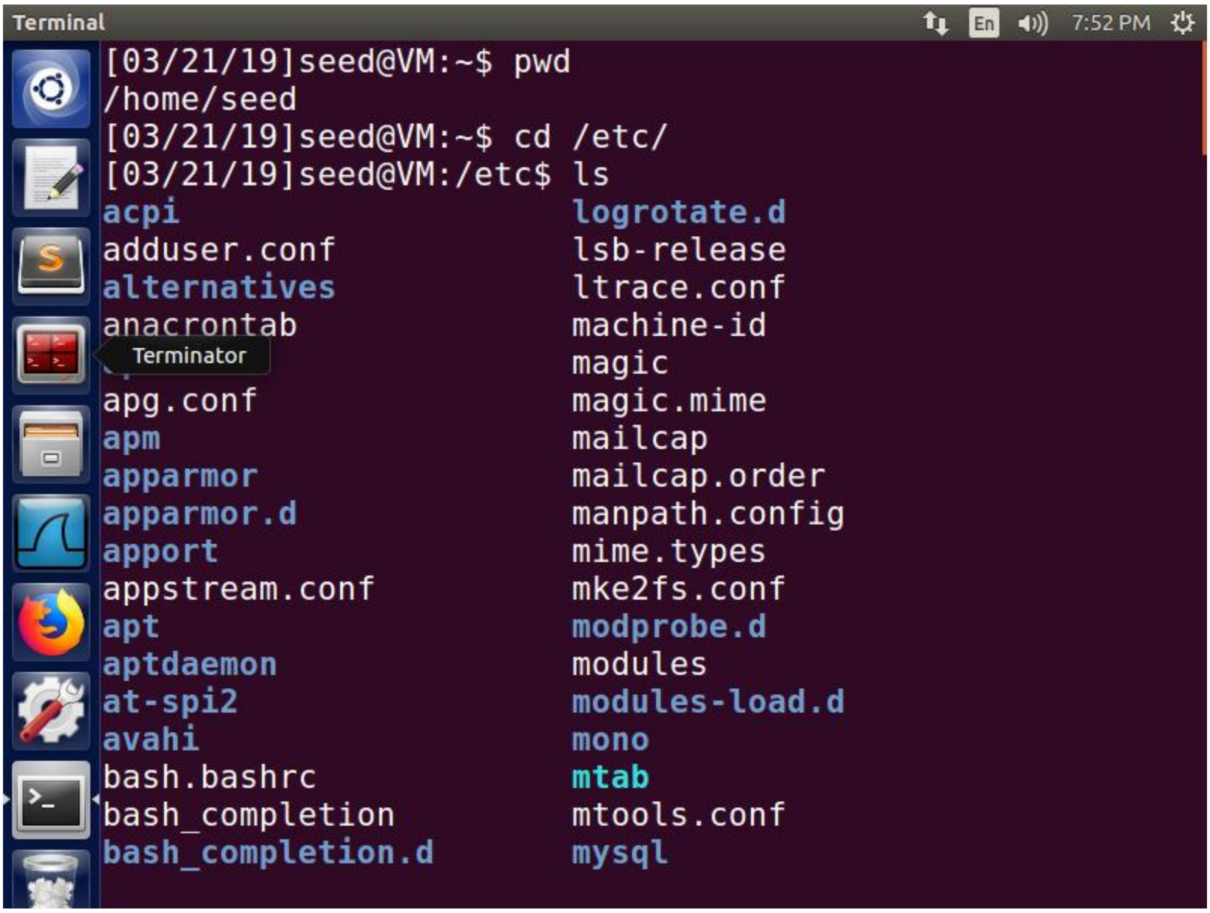
```
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                1B:18:60:DC:3F:55:7B:8B:C1:3D:A1:A5:EB:4F:43:0D:61:E1:49:5C
            X509v3 Authority Key Identifier:
                keyid:F4:D5:9D:C0:8A:AF:88:F9:29:BE:3C:E5:02:86:92:D9:63:9C:DA:B
0

Certificate is to be certified until Mar 20 00:55:12 2020 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Task 5

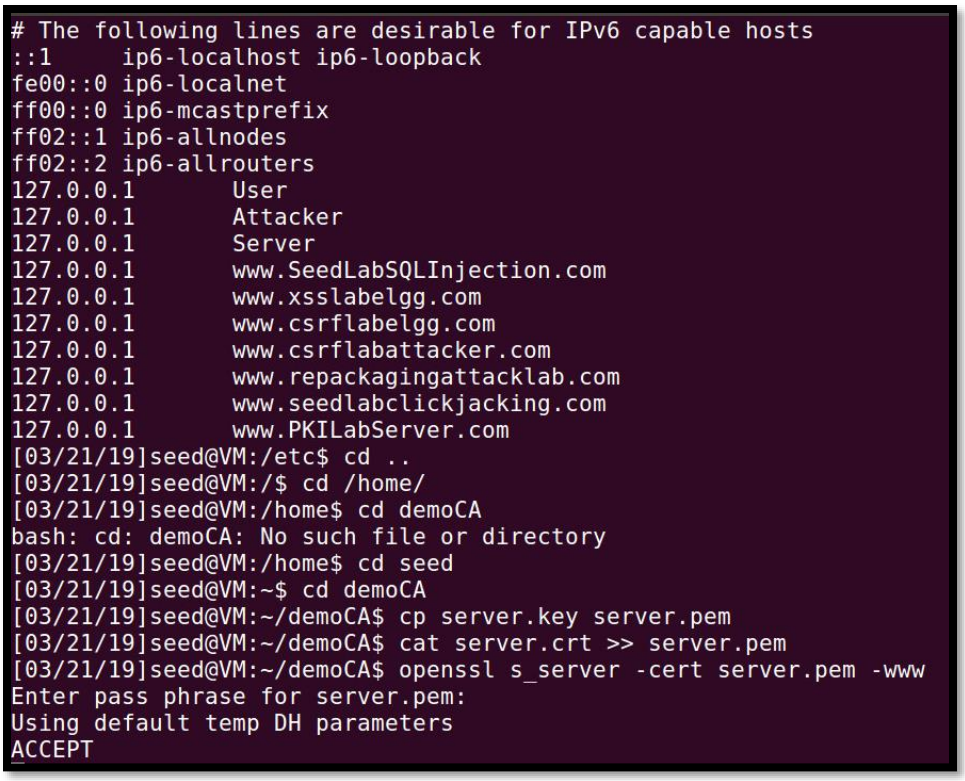
In this task, we will explore how public-key certificates are used by web sites to secure web browsing. First, domain name is need to be obtained. So, PKILabServer.com as domain name will be used. In order for our computer to get the domain name, /etc/hosts as shown below:



```
[03/21/19]seed@VM:~$ pwd
/home/seed
[03/21/19]seed@VM:~$ cd /etc/
[03/21/19]seed@VM:/etc$ ls
acpi                        logrotate.d
adduser.conf                lsb-release
alternatives                ltrace.conf
anacrontab                  machine-id
                            magic
apg.conf                    magic.mime
apm                         mailcap
apparmor                    mailcap.order
apparmor.d                  manpath.config
apport                      mime.types
appstream.conf              mke2fs.conf
apt                         modprobe.d
aptdaemon                   modules
at-spi2                     modules-load.d
avahi                       mono
bash.bashrc                 mtab
bash_completion             mtools.conf
bash_completion.d           mysql
```

Next, OpenSSL allows to start with a simple web server using the s server commands. Following are the commands which we used as shown below:

- cp server.key server.pem
- cat server.crt >> server.pem
- openssl s_server -cert server.pem -www



```
# 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       www.PKILabServer.com
[03/21/19]seed@VM:/etc$ cd ..
[03/21/19]seed@VM:/$ cd /home/
[03/21/19]seed@VM:/home$ cd demoCA
bash: cd: demoCA: No such file or directory
[03/21/19]seed@VM:/home$ cd seed
[03/21/19]seed@VM:~$ cd demoCA
[03/21/19]seed@VM:~/demoCA$ cp server.key server.pem
[03/21/19]seed@VM:~/demoCA$ cat server.crt >> server.pem
[03/21/19]seed@VM:~/demoCA$ openssl s_server -cert server.pem -www
Enter pass phrase for server.pem:
Using default temp DH parameters
ACCEPT
```

Next, the server can be accessed using the URL: https://PKILabServer.com:4433/. But, an error will occur from the browser. In Firefox, the certificate is not trusted as the issuer certificate is not known. Hence, we imported our own certificate. This was done by following these steps; Edit -> Preference -> Advanced -> View Certificates. Then the file 'ca.crt' was imported, the URL was run again and the following web page appeared:

If a byte in server.pem is modified using bless, then when the server is run it will show a message like not able to start.