भारतीय प्रौद्योगिकी संस्थान हैदराबाद

Indian Institute of Technology Hyderabad

Course:- Network Security(CS6903)

Report: Assignment 3

Project Members-

Sonu Kumar (CS20 MTECH 11011) Github Sonu228

Prabhat Kumar Kushwaha (NS 20 MTECH 11003)

prabhat0987

Subham Patel (NS 20 MTECH 11004) shubham-patel22

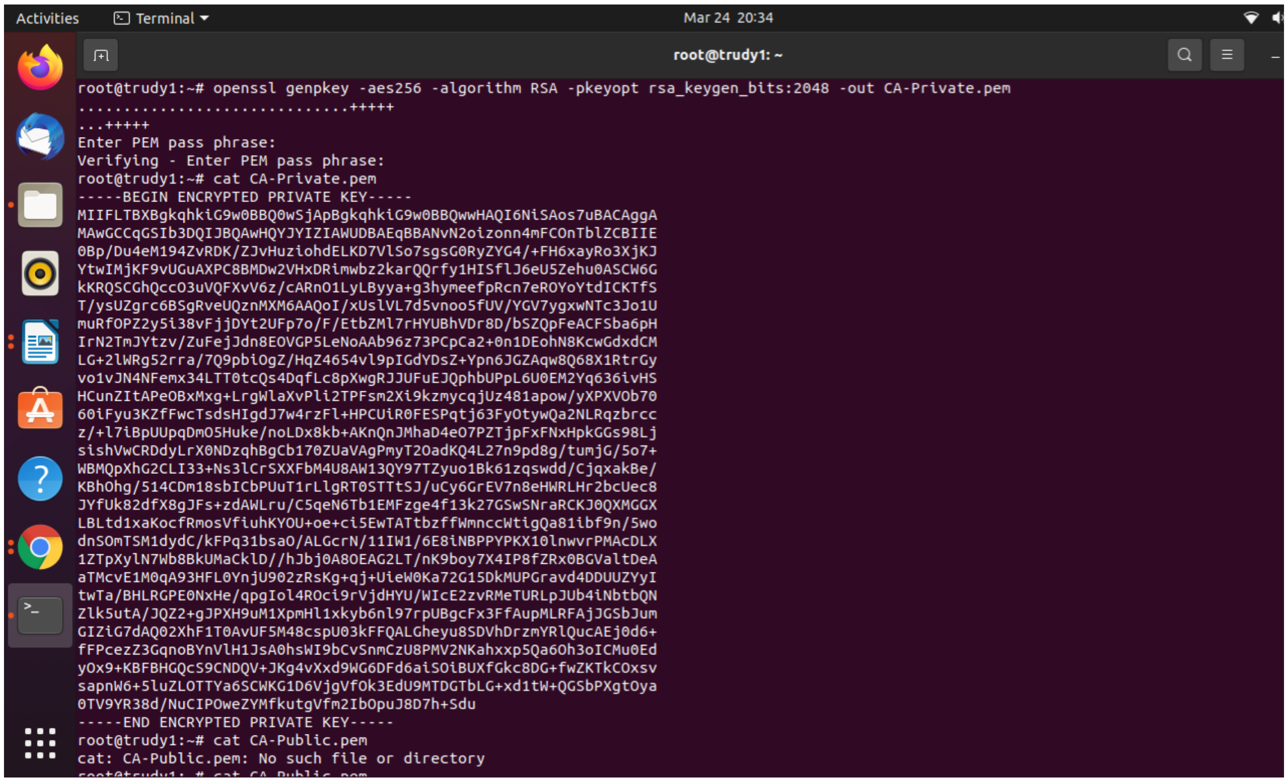
Certificate Authority, Trudy - Sonu Kumar
Bob - Prabhat Kumar Kushwaha
Alice - Subham Patel

Task 1:

In task 1 we used Trudy lxd container as CA
Bob and Alice container used to store Bob and Alice key , crt and csr.

CA Certificate generation:

Step1: RSA key generation using openssl command:
# openssl genpkey -aes256 -algorithm RSA -pkeyopt rsa_keygen_bits:2048 -out CA-Privat.pem

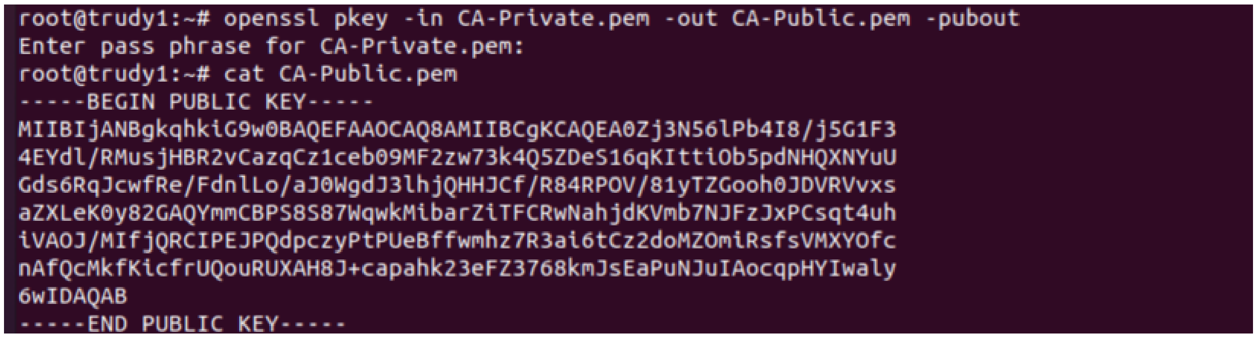


```
root@trudy1:~# openssl genpkey -aes256 -algorithm RSA -pkeyopt rsa_keygen_bits:2048 -out CA-Private.pem
..............................+++++
...+++++
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
root@trudy1:~# cat CA-Private.pem
-----BEGIN ENCRYPTED PRIVATE KEY-----
MIIFLTBXBgkqhkiG9w0BBQ0wSjApBgkqhkiG9w0BBQwwHAQI6NiSAos7uBACAggA
MAwGCCqGSIb3DQIJBQAwHQYJYIZIAWUDBAEqBBANvN2oizonn4mFCOnTblZCBIIE
0Bp/Du4eM194ZvRDK/ZJvHuziohdELKD7VlSo7sgsG0RyZYG4/+FH6xayRo3XjKJ
YtwIMjKF9vUGuAXPC8BMDw2VHxDRimwbz2karQQrfy1HISflJ6eU5Zehu0ASCW6G
kKRQSCGhQccO3uVQFXvV6z/cARnO1LyLByya+g3hymeefpRcn7eROYoYtdICKTfS
T/ysUZgrc6BSgRveUQznMXM6AAQoI/xUslVL7d5vnoo5fUV/YGV7ygxwNTc3Jo1U
muRfOPZ2y5i38vFjjDYt2UFp7o/F/EtbZMl7rHYUBhVDr8D/bSZQpFeACFSba6pH
IrN2TmJYtzv/ZuFejJdn8EOVGP5LeNoAAb96z73PCpCa2+0n1DEohN8KcwGdxdCM
LG+2lWRg52rra/7Q9pbiOgZ/HqZ4654vl9pIGdYDsZ+Ypn6JGZAqw8Q68X1RtrGy
vo1vJN4NFemx34LTT0tcQs4DqfLc8pXwgRJJUFuEJQphbUPpL6U0EM2Yq636ivHS
HCunZItAPeOBxMxg+LrgWlaXvPli2TPFsm2Xi9kzmycqjUz481apow/yXPXVOb70
60iFyu3KZfFwcTsdsHIgdJ7w4rzFl+HPCUiR0FESPqtj63FyOtywQa2NLRqzbrcc
z/+l7iBpUUpqDmO5Huke/noLDx8kb+AKnQnJMhaD4eO7PZTjpFxFNxHpkGGs98Lj
sishVwCRDdyLrX0NDzqhBgCb170ZUaVAgPmyT2OadKQ4L27n9pd8g/tumjG/5o7+
WBMQpXhG2CLI33+Ns3lCrSXXFbM4U8AW13QY97TZyuo1Bk61zqswdd/CjqxakBe/
KBhOhg/514CDm18sbICbPUuT1rLlgRT0STTtSJ/uCy6GrEV7n8eHWRLHr2bcUec8
JYfUk82dfX8gJFs+zdAWLru/C5qeN6Tb1EMFzge4f13k27GSwSNraRCKJ0QXMGGX
LBLtd1xaKocfRmosVfiuhKYOU+oe+ci5EwTATtbzffWmnccWtigQa81ibf9n/5wo
dnSOmTSM1dydC/kFPq31bsaO/ALGcrN/11IW1/6E8iNBPPYPKX10lnwvrPMAcDLX
1ZTpXylN7Wb8BkUMaCklD//hJbj0A8OEAG2LT/nK9boy7X4IP8fZRx0BGValtDeA
aTMcvE1M0qA93HFL0YnjU902zRsKg+qj+UieW0Ka72G15DkMUPGravd4DDUUZYyI
twTa/BHLRGPE0NxHe/qpgIol4ROci9rVjdHYU/WIcE2zvRMeTURLpJUb4iNbtbQN
Zlk5utA/JQZ2+gJPXH9uM1XpmHl1xkyb6nl97rpUBgcFx3FfAupMLRFAjJGSbJum
GIZiG7dAQ02XhF1T0AvUF5M48cspU03kFFQALGheyu8SDVhDrzmYRlQucAEj0d6+
fFPcezZ3GqnoBYnVlH1JsA0hsWI9bCvSnmCzU8PMV2NKahxxp5Qa6Oh3oICMu0Ed
yOx9+KBFBHGQcS9CNDQV+JKg4vXxd9WG6DFd6aiSOiBUXfGkc8DG+fwZKTkCOxsv
sapnW6+5luZLOTTYa6SCWKG1D6VjgVfOk3EdU9MTDGTbLG+xd1tW+QGSbPXgtOya
0TV9YR38d/NuCIPOweZYMfkutgVfm2IbOpuJ8D7h+Sdu
-----END ENCRYPTED PRIVATE KEY-----
root@trudy1:~# cat CA-Public.pem
cat: CA-Public.pem: No such file or directory
```

Step 2:

Public key Extraction using openssl command:

openssl pkey -in CA-Private.pem -out CA-Public.pem -pubout

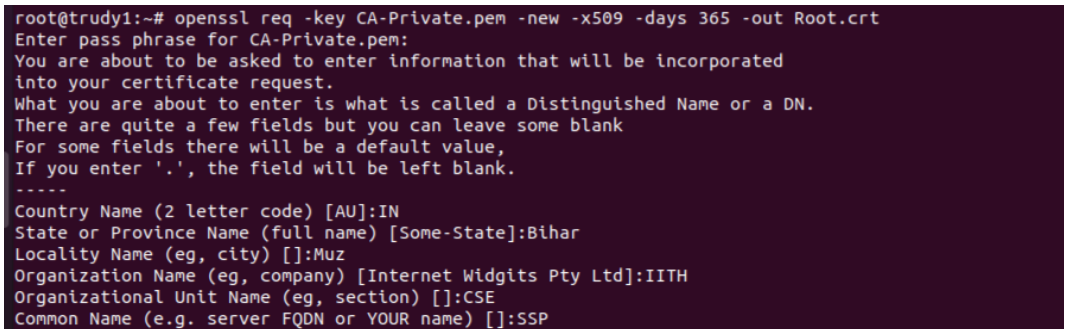


```
root@trudy1:~# openssl pkey -in CA-Private.pem -out CA-Public.pem -pubout
Enter pass phrase for CA-Private.pem:
root@trudy1:~# cat CA-Public.pem
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA0Zj3N56lPb4I8/j5G1F3
4EYdl/RMusjHBR2vCazqCz1ceb09MF2zw73k4Q5ZDeS16qKIttiOb5pdNHQXNYuU
Gds6RqJcwfRe/FdnlLo/aJ0WgdJ3lhjQHHJCf/R84RPOV/81yTZGooh0JDVRVvxs
aZXLeK0y82GAQYmmCBPS8S87WqwkMibarZiTFCRwNahjdKVmb7NJFzJxPCsqt4uh
iVAOJ/MIfjQRCIPEJPQdpczyPtPUeBffwmhz7R3ai6tCz2doMZOmiRsfsVMXYOfc
nAfQcMkfKicfrUQouRUXAH8J+capahk23eFZ3768kmJsEaPuNJuIAocqpHYIwaly
6wIDAQAB
-----END PUBLIC KEY-----
```

Step 3:

Generate self signed certificate using openssl command

openssl req -key CA-Private.pem -new -x509 -days 365 -out Root.crt

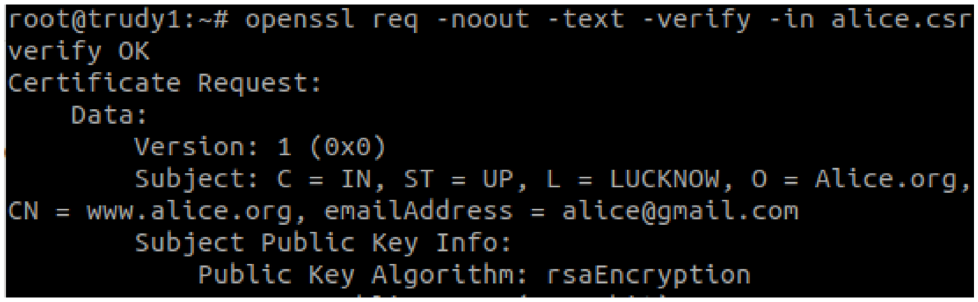


```
root@trudy1:~# openssl req -key CA-Private.pem -new -x509 -days 365 -out Root.crt
Enter pass phrase for CA-Private.pem:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
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]:IN
State or Province Name (full name) [Some-State]:Bihar
Locality Name (eg, city) []:Muz
Organization Name (eg, company) [Internet Widgits Pty Ltd]:IITH
Organizational Unit Name (eg, section) []:CSE
Common Name (e.g. server FQDN or YOUR name) []:SSP
```

● Verify Alice signed certificate

Using openssl command

# openssl req -noout -verify -in alice.csr

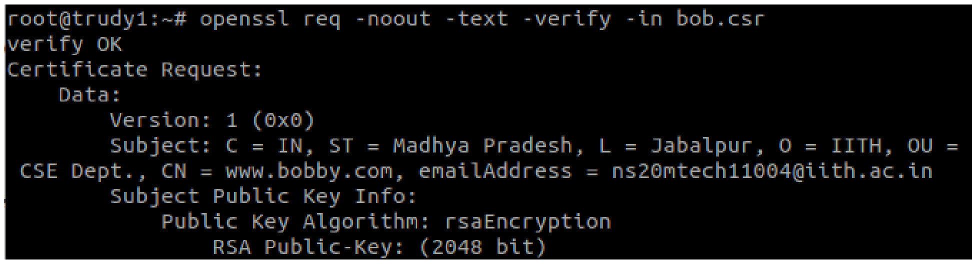


```
root@trudy1:~# openssl req -noout -text -verify -in alice.csr
verify OK
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: C = IN, ST = UP, L = LUCKNOW, O = Alice.org,
CN = www.alice.org, emailAddress = alice@gmail.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
```

●Verify Bob signed certificate

Using openssl command

#openssl req noout text -verify in bob.csr

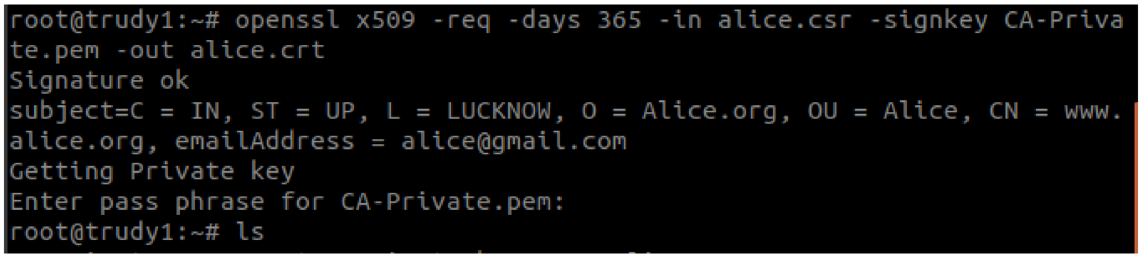


```
root@trudy1:~# openssl req -noout -text -verify -in bob.csr
verify OK
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: C = IN, ST = Madhya Pradesh, L = Jabalpur, O = IITH, OU =
 CSE Dept., CN = www.bobby.com, emailAddress = ns20mtech11004@iith.ac.in
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
```

●CA generate Signed certificate for Alice

Using openssl command

# openssl x509 -req -days -365 -in alice.csr -signkey CA-Private.pem -out alice.crt

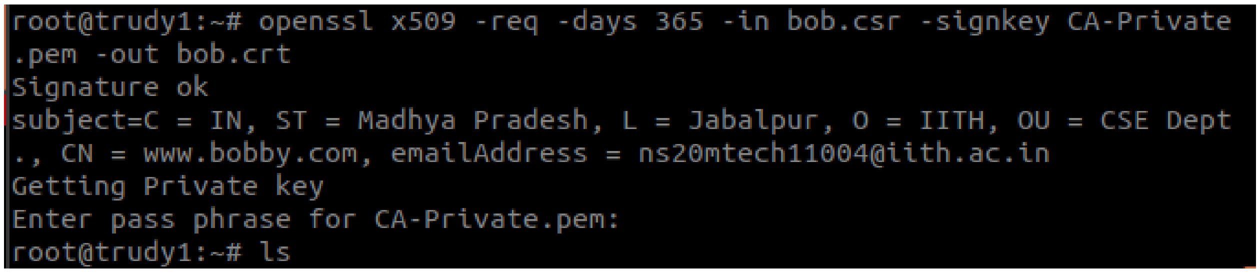


```
root@trudy1:~# openssl x509 -req -days 365 -in alice.csr -signkey CA-Priva
te.pem -out alice.crt
Signature ok
subject=C = IN, ST = UP, L = LUCKNOW, O = Alice.org, OU = Alice, CN = www.
alice.org, emailAddress = alice@gmail.com
Getting Private key
Enter pass phrase for CA-Private.pem:
root@trudy1:~# ls
```

●CA generate signed certificate for Bob

Using openssl command

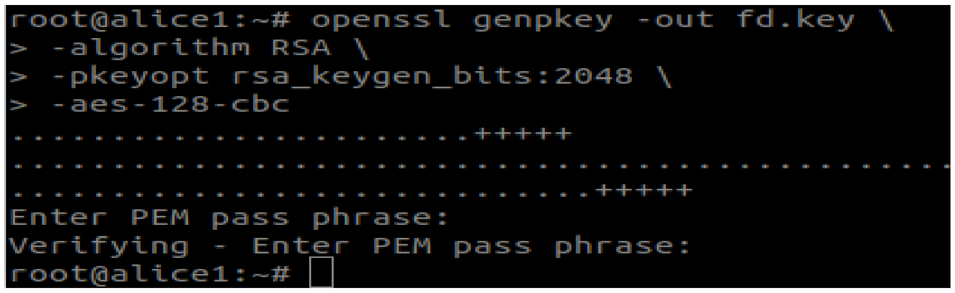
# openssl x509 -req -days -365 -in bob.csr -signkey CA-Private.pem -out bob.crt


```
root@trudy1:~# openssl x509 -req -days 365 -in bob.csr -signkey CA-Private
.pem -out bob.crt
Signature ok
subject=C = IN, ST = Madhya Pradesh, L = Jabalpur, O = IITH, OU = CSE Dept
., CN = www.bobby.com, emailAddress = ns20mtech11004@iith.ac.in
Getting Private key
Enter pass phrase for CA-Private.pem:
root@trudy1:~# ls
```

Alice certificate generation

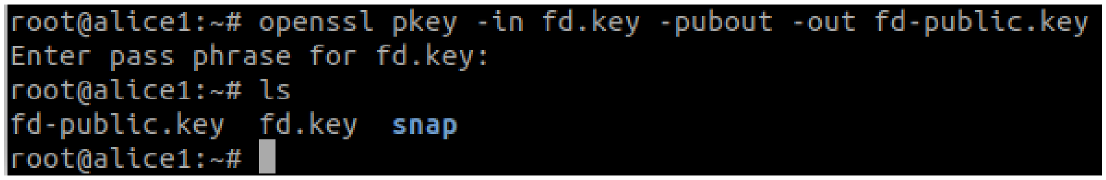
● Key generation using openssl command

openssl genpkey -out for.key -algorithm RSA -aes -128 -cbc


```
root@alice1:~# openssl genpkey -out fd.key \
> -algorithm RSA \
> -pkeyopt rsa_keygen_bits:2048 \
> -aes-128-cbc
.......................+++++
................................................
..............................+++++
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
root@alice1:~# 
```

● Public key Extraction using openssl command

openssl pkey -in fd.key -pubout -out fd-publc.key

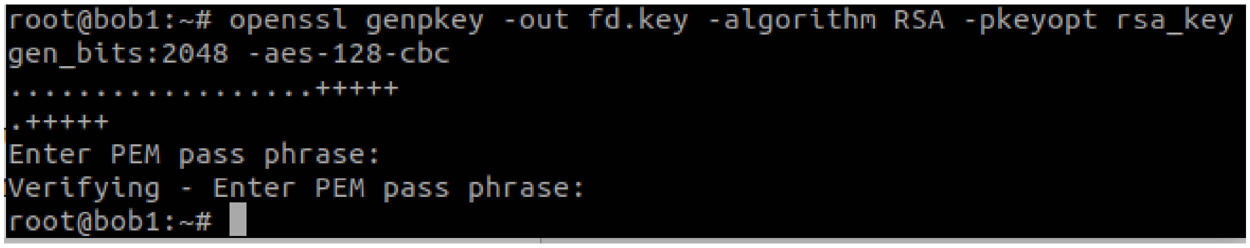

```
root@alice1:~# openssl pkey -in fd.key -pubout -out fd-public.key
Enter pass phrase for fd.key:
root@alice1:~# ls
fd-public.key  fd.key  snap
root@alice1:~# 
```

● generate csr certificate

Bob certificate generation

● Key generation using openssl command

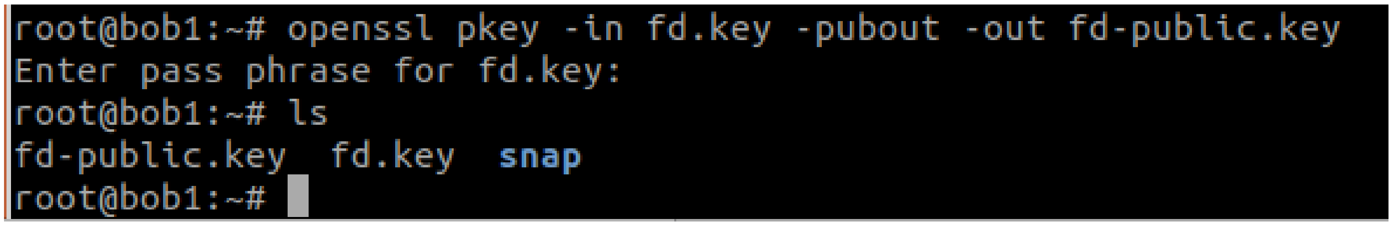
openssl genpkey -out for.key -algorithm RSA -pkeyout rsa_key gen_bit:2048 -aes -128 -cbc



```
root@bob1:~# openssl genpkey -out fd.key -algorithm RSA -pkeyopt rsa_key
gen_bits:2048 -aes-128-cbc
..................+++++
.+++++
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
root@bob1:~# 
```

●Public key Extraction using openssl command

#openssl pkey -in fd.key -pubout -out fd-publc.key

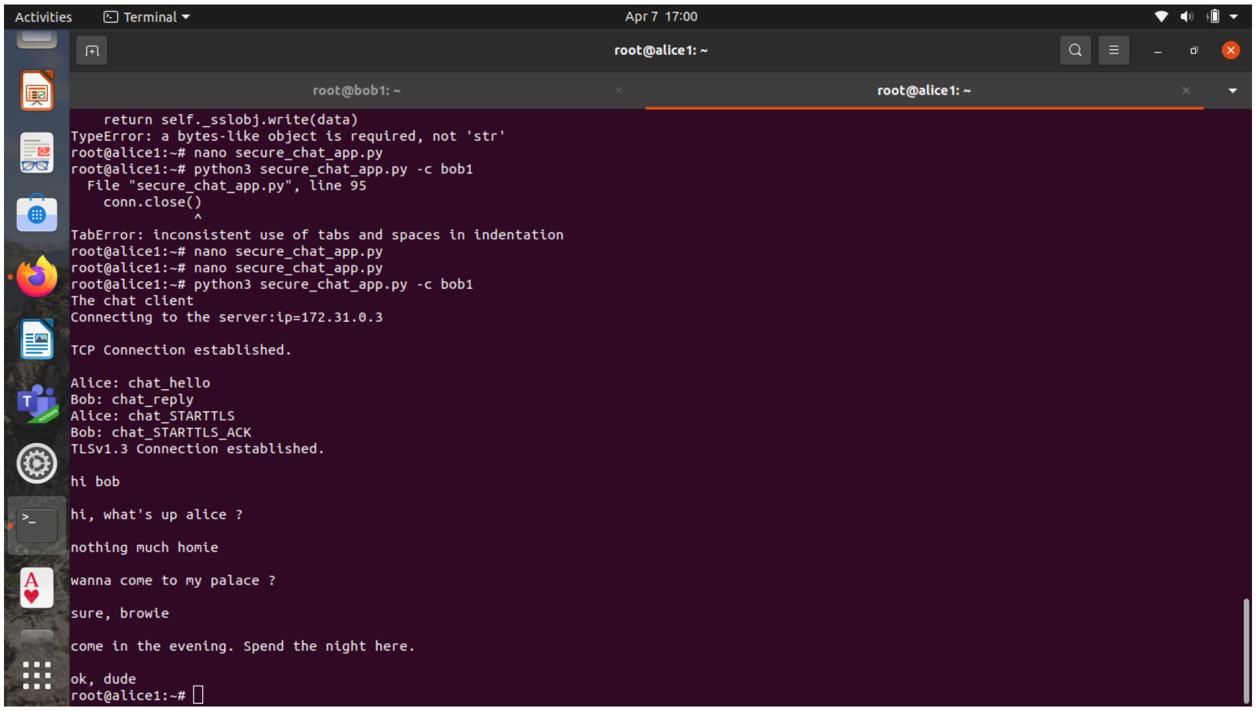


```
root@bob1:~# openssl pkey -in fd.key -pubout -out fd-public.key
Enter pass phrase for fd.key:
root@bob1:~# ls
fd-public.key  fd.key  snap
root@bob1:~# 
```

Task 2:

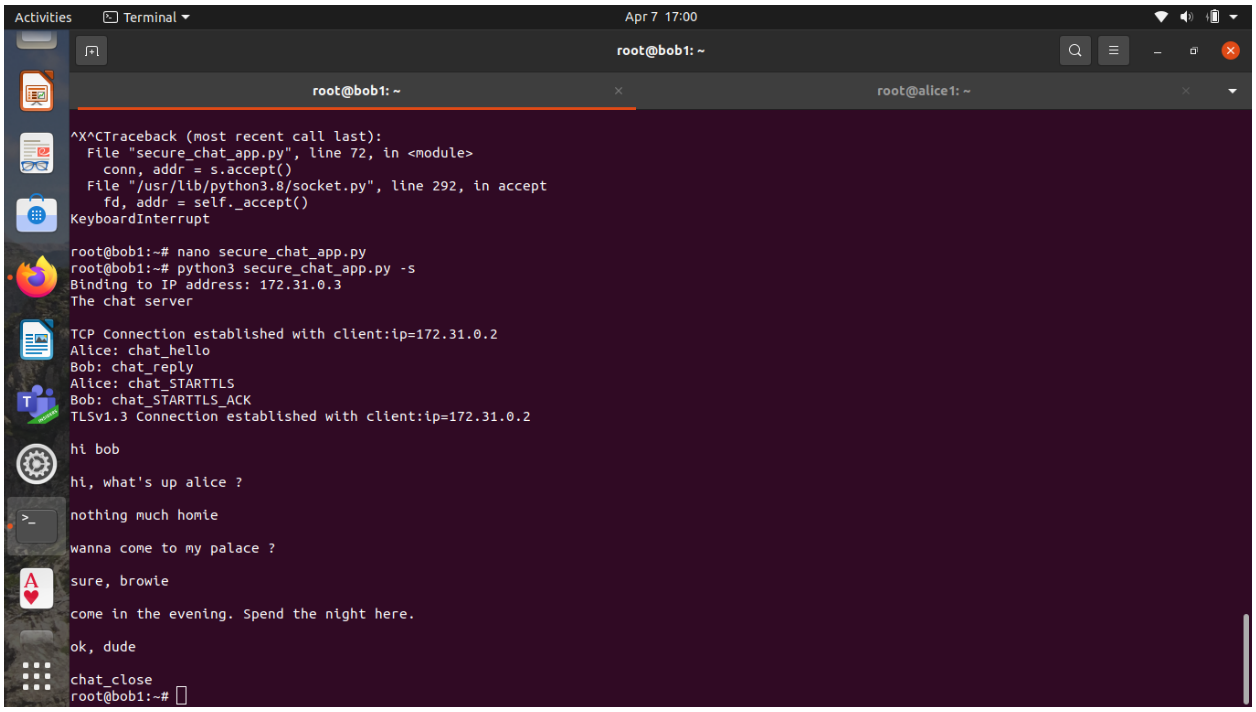
We have developed the code in python.

Alice Side(Client side)



```
    return self._sslobj.write(data)
TypeError: a bytes-like object is required, not 'str'
root@alice1:~# nano secure_chat_app.py
root@alice1:~# python3 secure_chat_app.py -c bob1
  File "secure_chat_app.py", line 95
    conn.close()
            ^
TabError: inconsistent use of tabs and spaces in indentation
root@alice1:~# nano secure_chat_app.py
root@alice1:~# nano secure_chat_app.py
root@alice1:~# python3 secure_chat_app.py -c bob1
The chat client
Connecting to the server:ip=172.31.0.3

TCP Connection established.

Alice: chat_hello
Bob: chat_reply
Alice: chat_STARTTLS
Bob: chat_STARTTLS_ACK
TLSv1.3 Connection established.

hi bob

hi, what's up alice ?

nothing much homie

wanna come to my palace ?

sure, browie

come in the evening. Spend the night here.

ok, dude
root@alice1:~# 
```

Bob Side(Server Side)



```
^X^CTraceback (most recent call last):
  File "secure_chat_app.py", line 72, in <module>
    conn, addr = s.accept()
  File "/usr/lib/python3.8/socket.py", line 292, in accept
    fd, addr = self._accept()
KeyboardInterrupt

root@bob1:~# nano secure_chat_app.py
root@bob1:~# python3 secure_chat_app.py -s
Binding to IP address: 172.31.0.3
The chat server

TCP Connection established with client:ip=172.31.0.2
Alice: chat_hello
Bob: chat_reply
Alice: chat_STARTTLS
Bob: chat_STARTTLS_ACK
TLSv1.3 Connection established with client:ip=172.31.0.2

hi bob

hi, what's up alice ?

nothing much homie

wanna come to my palace ?

sure, browie

come in the evening. Spend the night here.

ok, dude

chat_close
root@bob1:~# 
```

Task 3

The approach we used is as follows :
• to DNS poisoning. Use Alice and Bob's msg will start going to Trudy.
• Bob will start secure chat as server.
•Trudy use its interceptor program will become client for Bob
And also become server for Alice.
• When Alice starts secure chat as client it will get connected to Trudy.
• Whatever buffer Trudy receives from Alice copies to Bob's buffer and vice versa.
•Except the chat_STARTTLS msg from Alice which he will not pass to Bob and just reply to Alice chat_STARTTLS_NOT_SUPPORTED.

TASK 4
If you have implemented Task 3 then it is similar.

To DNS poisoning. Use Alice and Bob's msg will start going to Trudy.
•. Bob will start secure chat as server.
•. Trudy through its interceptor program will become client for Bob
And also become server for Alice.
•. When Alice starts secure chat as client it will get connected to Trudy.
•. Whatever buffer Trudy receives from Alice, he can send the same or modify it before passing it to Bob and vice versa.

•. When Alice starts the TLS connection with Bob, it actually reaches Trudy. So Trudy will establish one TLS connection with Alice as server. Trudy will establish another TLS connection with Bob as client.

•Trudy becomes server (use fakebob.crt) for alice and becomes client (fakealice.crt) for bob. So 2 TLS connection method in a single file.