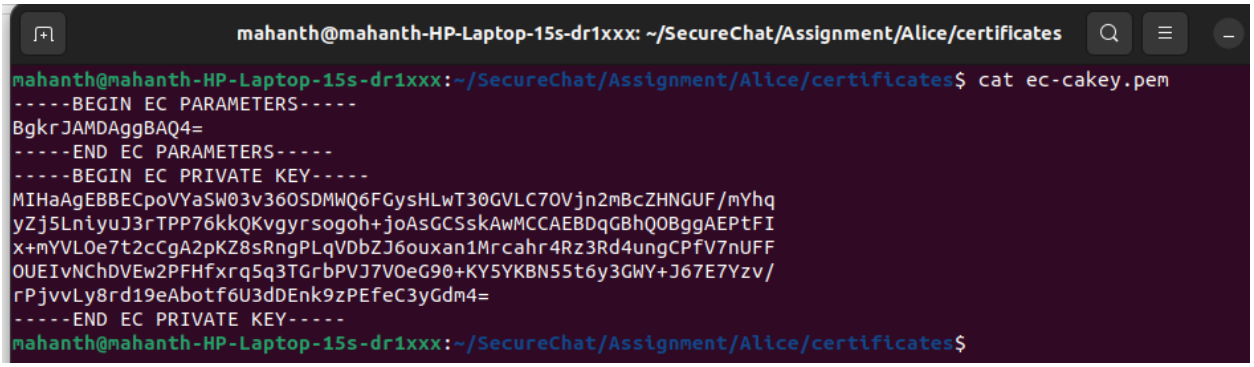
TASK 1 : Certificates

Commands for Creating a root CA certificate (V3 X.509 certificate, self-signed using 512-bit ECC Private Key of the root)

We have generated the subject name: NTS Root R1, V3 X.509 certificate, self-signed using 512-bit ECC Private Key of the root .

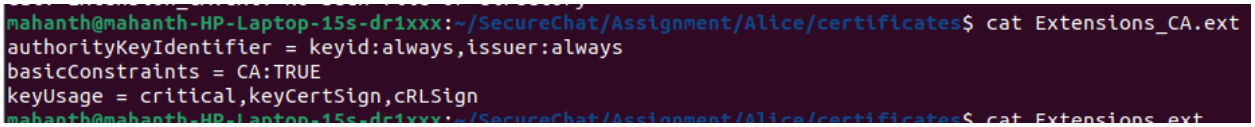
```
Command: openssl ecparam -out ec-cakey.pem -name brainpoolP512t1 -genkey
```




```
Command: openssl req -new -key ec-cakey.pem -out rootCA.csr
```

```
Command: openssl x509 -req -in rootCA.csr -signkey ec-cakey.pem -out
rootCA.crt -days 3650 -sha256 -extfile Extensions_CA.ext
```

The extensions had the CA true indicating that it can sign the other certificates.



```
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$ cat rootCA.crt
-----BEGIN CERTIFICATE-----
MIIDszCCAxegAwIBAgIUdTFWaHQ7qrKUrAf3Uj3+6iFS3qkwCgYIKoZIzj0EAwIw
gYwxCzAJBgNVBAYTAklOMQswCQYDVQQIDAJUUzEMMAoGA1UEBwwDSFlEMRQwEgYD
VQQKDAtOVFMgUm9vdCBSMTEMMAoGA1UECwwDTlRTMRQwEgYDVQQDDAtOVFMgUm9v
dCBSMTEoMCYGCSqGSIb3DQEJARYZY3MyMm10ZWNoMTEwMTVAaWl0aC5hYy5pbjAe
Fw0yMzAzMjUxMjAzMjNaFw0zMzAzMjIxMjAzMjNaMIGMMQswCQYDVQQGEwJJTjEL
MAkGA1UECAwCVFMxDDAKBgNVBAcMA0hZRDEUMBIGA1UECgwLTlRTIFJvb3QgUjEx
DDAKBgNVBAsMA05UUzEUMBIGA1UEAwwLTlRTIFJvb3QgUjExKDAmBgkqhkiG9w0B
CQEWGWNzMjJtdGVjaDExMDE1QGlpdGguYWMuaW4wgZswFAYHKoIzj0CAQYJKyQD
AwIIAQEOA4GCAAQ+0UjH6ZhUs57u3ZwKADakpnyxGeA8upUNtknqi7FqfUytxqGv
hHPdF3i6eAI99XudQUU5QQi80KENUTDY8Ud/GurmrdMats9UntU54b3T4pjlgoE3
nm3rLcZZj4nrsTtjO/+s+O+8vLyt3X14Bui1/pTd0MSeT3M8R94LfIZ2bqOCARAw
ggEMMIHMBgNVHSMEgcQwgcGAFGLxGw+2zrlHqbJVxDtK9rOXDh4IoYGSpIGPMIGM
MQswCQYDVQQGEwJJTjELMAkGA1UECAwCVFMxDDAKBgNVBAcMA0hZRDEUMBIGA1UE
CgwLTlRTIFJvb3QgUjExDDAKBgNVBAsMA05UUzEUMBIGA1UEAwwLTlRTIFJvb3Qg
UjExKDAmBgkqhkiG9w0BCQEWGWNzMjJtdGVjaDExMDE1QGlpdGguYWMuaW6CFHUx
Vmh0O6qylKwH91I9/uohUt6pMAwGA1UdEwQFMAMBAf8wDgYDVR0PAQH/BAQDAgEG
MB0GA1UdDgQWBBRi8RsPts65R6myVcQ7SvazlW4eCDAKBggqhkjOPQQDAgOBiQAw
gYUCQQCjNNHvpZu3jjJWxE2pcb8PfrYSYVMYAQlJWy1Lr9D/3hpPFYhO1CkH7yfn
RwcViLIBOM+/jO4YnVy5We63R6ZYAkB+RMhkYkne3RgqyUm6vO5mCXupoDJZUu/K
eoYJwOULSBgS5h7QAQLZTwULH8LYG88q2lm4UxYh4gCBrVU5hjzS
-----END CERTIFICATE-----
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$
```

Commands to generate the intermediate certificate signed by rootCA

```
Command: openssl req -newkey rsa:2048 -nodes -keyout intermediate_key.pem
-days 365 -out inter.csr
```

```
Command: openssl x509 -req -CA rootCA.crt -days 365 -in inter.csr -CAkey
ec-cakey.pem -CAcreateserial -out inter.crt -extfile Extensions_CA.ext
```

```
-----END CERTIFICATE-----
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$ cat inter.crt
-----BEGIN CERTIFICATE-----
MIIENDCCA5igAwIBAgIUCELwgnWa1cYJLHXBkV66SNk4d6AwCgYIKoZIzj0EAwIw
gYwxCzAJBgNVBAYTAklOMQswCQYDVQQIDAJUUzEMMAoGA1UEBwwDSFlEMRQwEgYD
VQQKDAtOVFMgUm9vdCBSMTEMMAoGA1UECwwDTlRTMRQwEgYDVQQDDAtOVFMgUm9v
dCBSMTEoMCYGCSqGSIb3DQEJARYZY3MyMm10ZWNoMTEwMTVAaWl0aC5hYy5pbjAe
Fw0yMzAzMjUxMjI3MjhaFw0yNDAzMjQxMjI3MjhaMIGFMQswCQYDVQQGEwJJTjEL
MAkGA1UECAwCVFMxDDAKBgNVBAcMA0hZRDEMMAoGA1UECgwDTlRTMQ4wDAYDVQQL
DAVJTlRFUjETMBEGA1UEAwwKTlRTIENBIDFSMzEoMCYGCSqGSIb3DQEJARYZY3My
Mm10ZWNoMTEwMTVAaWl0aC5hYy5pbjCCASIwDQYJKoZIhvcNAQEBBQADggEPADCC
AQoCggEBAPdvCG0HCEo4y2JvwWeb5pKQiXxWuyFmfjI/8rw+02f9qikb/hY2X6lO
RxgIh93bSIfbHeTySgHnzXUlyHRtB0ULub/iRZWf1bi6By2ZyUMF+WMWWPWWU3ym
cC+tNrjD9JWwoxB9WzYJ7mcUjPFHCtF6O80OBIV+6j8zQr8dbDBulmDLLb0DrHyU
abCNXTq6Ur7SzwJesNICHJ0oSrSp3oU/h9lO+TDagXCEQ9V6ROS7Umlm/FdjW036
9HVkXZ44RcOHJQvoRsEaAdkH1fXjTBLjI2ulxlN3QQIaQy1VM3vVV8Lm3+G7NP68
vxsldFyUMWB2+RUqws4RQu3XVvLlBrMCAwEAAaOCARAwggEMMIHMBgNVHSMEgcQw
gcGAFGLxGw+2zrlHqbJVxDtK9rOXDh4IoYGSpIGPMIGMMQswCQYDVQQGEwJJTjEL
MAkGA1UECAwCVFMxDDAKBgNVBAcMA0hZRDEUMBIGA1UECgwLTlRTIFJvb3QgUjEx
DDAKBgNVBAsMA05UUzEUMBIGA1UEAwwLTlRTIFJvb3QgUjExKDAmBgkqhkiG9w0B
CQEWGWNzMjJtdGVjaDExMDE1QGlpdGguYWMuaW6CFHUxVmh0O6qylKwH91I9/uoh
Ut6pMAwGA1UdEwQFMAMBAf8wDgYDVR0PAQH/BAQDAgEGMB0GA1UdDgQWBBQuB8Qn
QE98LndGZ8vm0hF6ialYzTAKBggqhkjOPQQDAgOBiQAwgYUCQQCdxgP17IBtAQ5k
aY7DeiGCbLFuON7ekgsS++Q8xXsXe830xuRR73t1UMTZ65InlsqH1RSQr3FaklGl
nKYkb6oQAkAv31ruT/n1RQ0fW+K0n22yIIhHB+6EOCSttеUKdPrmMr8ABJmN7ckM
llCFRVyj9Bf7Dk54abDeyzV0I2jwyijZ
-----END CERTIFICATE-----
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$
```

Commands to generate Alice certificates

We have changed the key size to 2048 bits since there was an error with a 1024 bits key saying that it is small.

```
Command: openssl req -newkey rsa:2048 -nodes -keyout alice.key -days 365
-out alice.csr -subj="/C=US/O=NTS/OU=CA/CN=Alice1.com"
```

```
Command: openssl x509 -req -CA inter_2.crt -days 365 -in alice.csr -CAkey
intermediate_key.pem -CAcreateserial -out alice_2.crt -extfile
Extensions.ext
```

```
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$ cat Extensions.ext
authorityKeyIdentifier = keyid,issuer
basicConstraints = CA:FALSE
keyUsage = digitalSignature, nonRepudiation,keyEncipherment,dataEncipherment
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$
```

```
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$ cat alice.crt
-----BEGIN CERTIFICATE-----
MIIDqzCCApOgAwIBAgIUaHSTS0a6X8gKaXFB+WQvLvcq4GswDQYJKoZIhvcNAQEL
BQAwgYUxCzAJBgNVBAYTAklOMQswCQYDVQQIDAJUUzEMMAoGA1UEBwwDSFlEMQww
CgYDVQQKDANOVFMxDjAMBgNVBAsMBUlOVEVSMRMwEQYDVQQDDApOVFMgQ0EgMVIz
MSgwJgYJKoZIhvcNAQkBFhljczIybXRlY2gxMTAxNUBpaXRoLmFjLmluMB4XDTIz
MDMyNTEyMjg1M1oXDTI0MDMyNDEyMjg1M1owPTELMAkGA1UEBhMCVVMxDDAKBgNV
BAoMA05UUzELMAkGA1UECwwCQ0ExEzARBgNVBAMMCkFsaWNlMS5jb20wggEiMA0G
CSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCX5lPHCrSs7cUAGKCT+zgJz4ZTIKVn
ildqh/nQJFyywJi0EdR+w+Z87WB4RaEVi7xF6IRVeRtCvXVuDErqYN7YOW0DlMkZ
4otffDUHbtzZKQdSm/sLpX9op0DmOknOpOIbrtqpULUAewUPxlQK7JA7ZVrJMacs
Akxs+esG0/gvJcPeFAHHDmKmd2t+KJHiK5XzxueXsoduxG1RT+vqLbQUiQ+6USls
4XHg+NTT0rlEylyd054oQSl2mv4npHiV0TtYKeR9aLVig4E7UDfLBuBvGMfZW+5B
vvo3R4SE0l5GsztSqdwXeihGDnah0x1QtKth5ciHUw/mXObRsJQh7MuDAgMBAAGj
WjBYMB8GA1UdIwQYMBaAFC4HxCdAT3wud0Zny+bSEXqJqVjNMAkGA1UdEwQCMAAw
CwYDVR0PBAQDAgTwMB0GA1UdDgQWBBTaFdPX9rGbQfzU79j17pAwlbKldDANBgkq
hkiG9w0BAQsFAAOCAQEAreG9bPiF9T8PsQ/TfipePocXdoV/f940T9jwJNXlYd5g
ojD/Yx4Fc4ulZYFRvOKgI270dCw82I51i9MCXIgrPizNPhKIDwtC2Z4+QZ9Yrphr
vg0PoB1jV+o2DHRdvkMdmUxRY6q+iUienLOI1V0wfOv+D6KeiBbeXi89HjOK8ZZ1
EHLp26DAQymsgOrC1uXIaYQVx+/Qr2UI1Mjw6Bq/bJIccmV6pxTo3VegkY5YuG8x
IVBTFwDm15Lj5lt8OG0BIFzjo6L03TKiyrlwL5nHFgemYg5iI2mZKlI6jZiU5Y3o
3nE3iV5KPQAxuui845nfBAggwWEpfJxvB0wBzYb71g==
-----END CERTIFICATE-----
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$
```

Commands to generate bob certificate

```
Command: openssl ecparam -genkey -name prime256v1 -out bob.key
```

```
Command: openssl req -new -key bob.key -out Bob.csr
-subj="/C=US/O=NTS/OU=CA/CN=Bob1.com"
```

```
Command: openssl x509 -req -CA inter_2.crt -days 365 -in Bob.csr -CAkey
```

```
intermediate_key.pem -CAcreateserial -out bob.crt -extfile Extensions.ext
```

```
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$ cat bob.crt
-----BEGIN CERTIFICATE-----
MIIC3jCCAcagAwIBAgIUJ08njewbyie+ut0pvT0TyLmu6LUwDQYJKoZIhvcNAQEL
BQAwgYUxCzAJBgNVBAYTAklOMQswCQYDVQQIDAJUUzEMMAoGA1UEBwwDSFlEMQww
CgYDVQQKDANOVFMxDjAMBgNVBAsMBUlOVEVSMRMwEQYDVQQDDApOVFMgQ0EgMVIz
MSgwJgYJKoZIhvcNAQkBFhljczIybXRlY2gxMTAxNUBpaXRoLmFjLmluMB4XDTIz
MDMyNTEyMjk0M1oXDTI0MDMyNDEyMjk0M1owOzELMAkGA1UEBhMCVVMxDDAKBgNV
BAoMA05UUzELMAkGA1UECwwCQ0ExETAPBgNVBAMMCEJvYjEuY29tMFkwEwYHKoZI
zj0CAQYIKoZIzj0DAQcDQgAEV5mgRZ//TwlIRcyKR9b5XdqaCi7UQ42kNTi2hSKU
jNJRzSGCEQWP5KWWzVOLhaTvMS3XdnyAa/KxPUCNP+qY5aNaMFgwHwYDVR0jBBgw
FoAULgfEJ0BPfC53RmfL5tIReompWM0wCQYDVR0TBAIwADALBgNVHQ8EBAMCBPAw
HQYDVR0OBBYEFKY0hdtDN2jdgjBA5HbD8zvMtOPxMA0GCSqGSIb3DQEBCwUAA4IB
AQAfi/Qq4uOk8LAuiUa3V6Gyae4UuvEJ33OhYb0cMjg9gQ9W3skz3zutWuw72e5u
1pLeeY/hRGVQaus6cPdiSzlFY5PcrWU/f4pZleDo7/GrPrltFPzsIi+qNqdP1kk4
NrCBdEEjodSRSBfxeay5zUkrH3CrvDIg0bqci/s9NNJ8CNNMWld2qdZ7Em8OWUoM
oIY8XGv5acDQ0NP7LYk/LJHU9BIJcKQM+Q5MKpTxzCagZT1Vlw09VP6bX96/KjDO
S8NuAhlsQ83Z/MFwL8uEiCqY7QD0f/+nFa/Mt3dntqIU0iVS5DRAp/RYmxta67tL
8p4tkV3QOWXUZobAoUu9vHBa
-----END CERTIFICATE-----
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$
```

VERIFICATION OF ALL THE CERTIFICATES:

Using a shell script we have combined both the intermediate and root certificates into pem file which we named it as CAfile.pem. Which is used later for verification purposes.

Shell script to combine the certificates

```
for i in inter_2.crt rootCA_2.crt; do
    openssl x509 -in $i -text >> CAfile.pem
done
```

```
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$ openssl verify -verbose -CAfile rootCA.crt inter.crt
inter.crt: OK
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/certificates$
```

```
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/new_certs$ openssl verify -verbose -CAfile CAfile.pem alice_2.crt
alice_2.crt: OK
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/new_certs$
```

```
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/new_certs$ openssl verify -verbose -CAfile CAfile.pem bob_2.crt
bob_2.crt: OK
mahanth@mahanth-HP-Laptop-15s-dr1xxx:~/SecureChat/Assignment/Alice/new_certs$
```

TASK 2 : Secure Chat

sec_server_client.cpp :

Server starts the app using command:

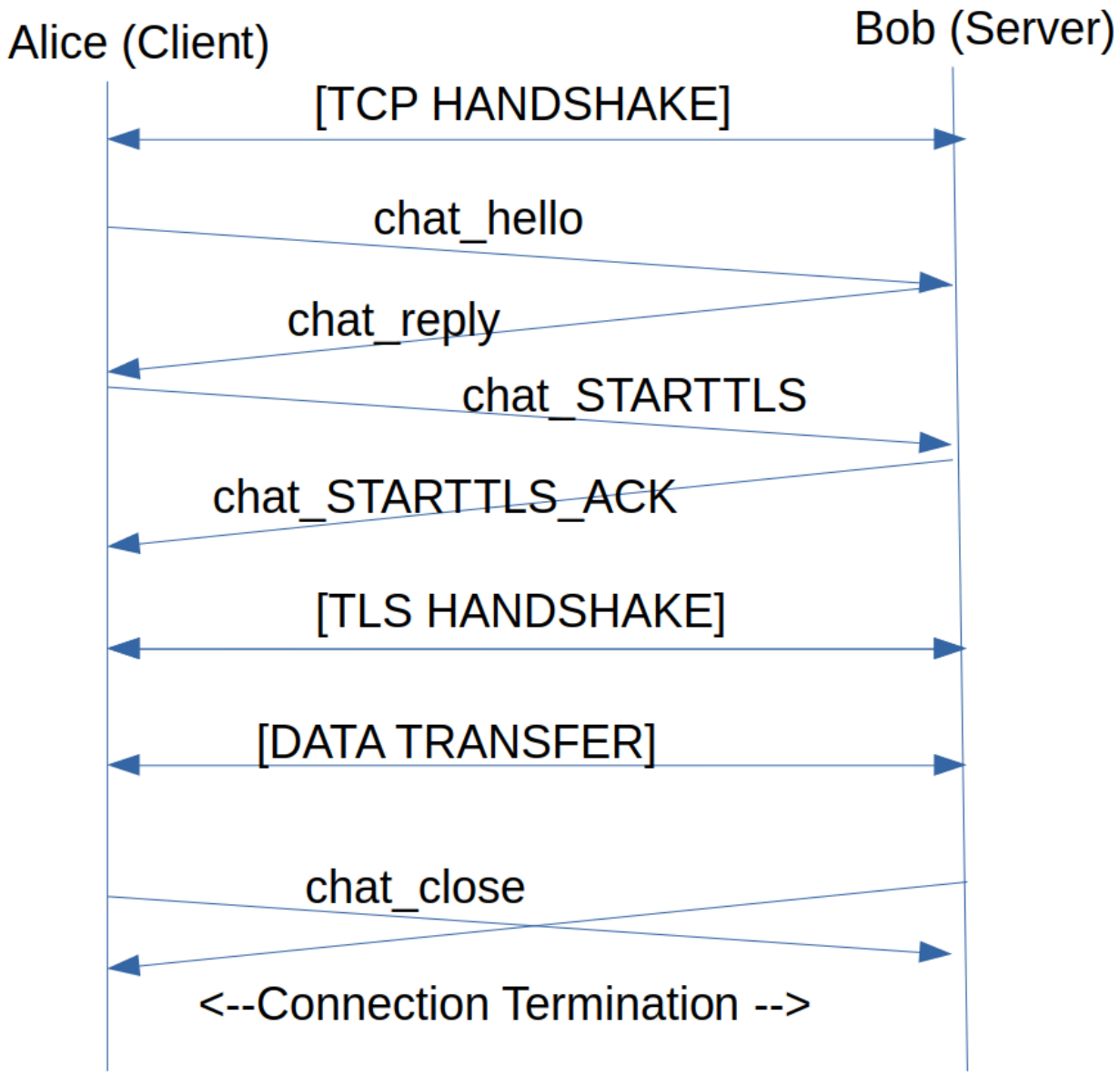
```
sec_server_client.cpp -s
```

Client starts the app using command :

```
sec_server_client.cpp -c bob1
```

This c++ program consists of 2 functions : runServer() and runClient(<server hostname>), which can be triggered by command line arguments -s and -c b server and client respectively.

The Communication Flow chart for this Task 2 has been shown below :



Functions :

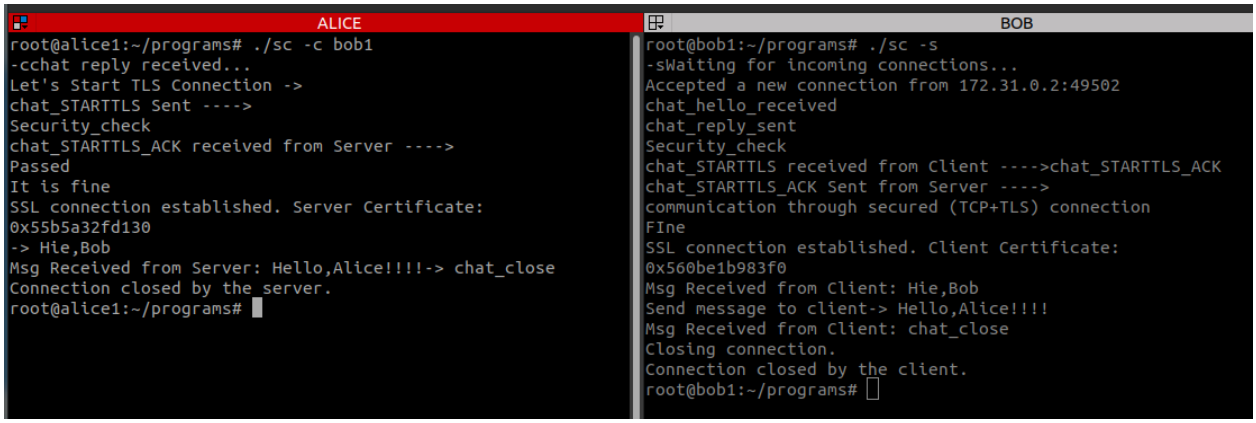
1. **runServer() :**
    - The socket is initially generated on the server side, attached to its port, and listens to see if any clients are connecting after the client and server certificates have been loaded.
    - The chat reply is sent to the client whenever a client connects to the port and the server receives the chat hello.
    - The communication will continue in an unsecured way unless the Server receives a chat STARTTLS message from the Client and responds with chat STARTTLS ACK to indicate that it supports Secure Communication.
    - As long as neither the Client nor the Server sends the other party a chat close

message, the Client and Server continue exchanging messages. If the Server receives a bye from the terminal, it sends the chat close message to the Client.
- With the chat close command, the client and server connection is broken.

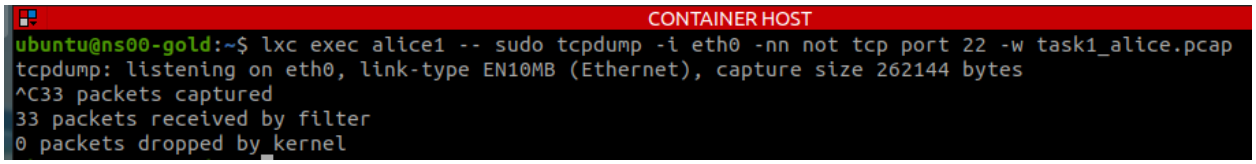
2. **runClient(<server hostname>) :**

- We determine the server IP address using gethostbyname(), and after that, we call connect() to establish a connection between the Client and the Server.
- The Client sends a chat hello to the Server upon a successful connection, and the Client then waits for the Server to respond.
- As a reply to chat hello, the Server sends the chat reply message to the Client.
- To begin Secured Communication with the Server, the Client sends chat STARTTLS and waits for a response from the Server.
- If the Server is TLS-capable, it will get a chat STARTTLS ACK, which will trigger the exchange of encrypted data.
- The data will be transmitted with encryption unless it receives the message chat STARTTLS NOT SUPPORTED, which triggers an unencrypted transmission.
- Unless one of the two parties gives the other a chat close message, Client and Server remain exchanging messages.

Alice1 and Bob1 Container Call Flow Terminal :



PCAP Evidences :
Taking pcap traces on Alice1 container :



TCP Handshake :

	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.31.0.2	172.31.0.3	TCP	74	56034 → 8084 [SYN] Seq=0
2	0.000146	172.31.0.3	172.31.0.2	TCP	74	8084 → 56034 [SYN, ACK] S
3	0.000181	172.31.0.2	172.31.0.3	TCP	66	56034 → 8084 [ACK] Seq=1

CHAT Hello :

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.31.0.2	172.31.0.3	TCP	74	56034 → 8084 [SYN] Seq=0 W
2	0.000146	172.31.0.3	172.31.0.2	TCP	74	8084 → 56034 [SYN, ACK] Se
3	0.000181	172.31.0.2	172.31.0.3	TCP	66	56034 → 8084 [ACK] Seq=1 A
4	0.000655	172.31.0.2	172.31.0.3	TCP	76	56034 → 8084 [PSH, ACK] Se
5	0.000694	172.31.0.3	172.31.0.2	TCP	66	8084 → 56034 [ACK] Seq=1 A
6	0.000790	172.31.0.3	172.31.0.2	TCP	76	8084 → 56034 [PSH, ACK] Se

```
▸ Frame 4: 76 bytes on wire (608 bits), 76 bytes captured (608 bits)
▸ Ethernet II, Src: Xensourc_6c:6d:b0 (00:16:3e:6c:6d:b0), Dst: Xensourc_ec:4b:d4 (00:16
▸ Internet Protocol Version 4, Src: 172.31.0.2, Dst: 172.31.0.3
▸ Transmission Control Protocol, Src Port: 56034, Dst Port: 8084, Seq: 1, Ack: 1, Len: 1

0000  00 16 3e ec 4b d4 00 16  3e 6c 6d b0 08 00 45 00   ··>·K··· >lm···E·
0010  00 3e 5b 74 40 00 40 06  87 02 ac 1f 00 02 ac 1f   ·>[t@·@· ········
0020  00 03 da e2 1f 94 92 56  37 a7 18 9e 04 0c 80 18   ·······V 7·······
0030  01 f6 58 74 00 00 01 01  08 0a f6 c2 bb f1 b4 cb   ··Xt···· ········
0040  7f 98 63 68 61 74 5f 68  65 6c 6c 6f              ··chat_h ello
```

CHAT Reply :

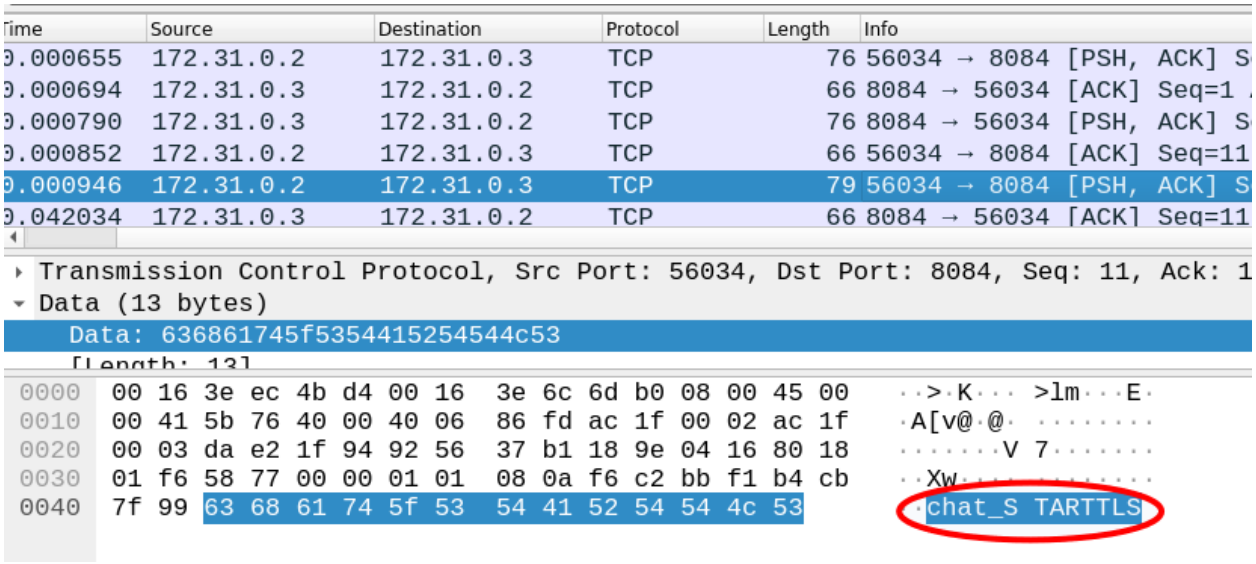
Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.31.0.2	172.31.0.3	TCP	74	56034 → 8084 [SYN]
2	0.000146	172.31.0.3	172.31.0.2	TCP	74	8084 → 56034 [SYN,
3	0.000181	172.31.0.2	172.31.0.3	TCP	66	56034 → 8084 [ACK]
4	0.000655	172.31.0.2	172.31.0.3	TCP	76	56034 → 8084 [PSH,
5	0.000694	172.31.0.3	172.31.0.2	TCP	66	8084 → 56034 [ACK]
6	0.000790	172.31.0.3	172.31.0.2	TCP	76	8084 → 56034 [PSH,

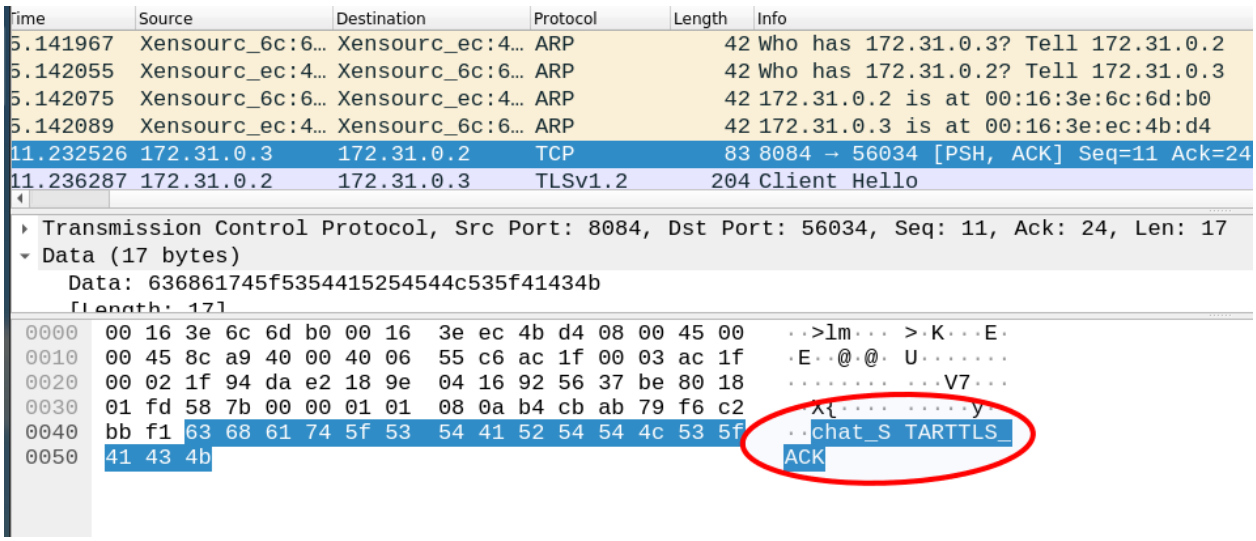
```
▸ Transmission Control Protocol, Src Port: 8084, Dst Port: 56034, Seq: 1, Ack: 11
▾ Data (10 bytes)
    Data: 636861745f7265706c79
    [Length: 10]

0000  00 16 3e 6c 6d b0 00 16  3e ec 4b d4 08 00 45 00   ··>lm··· >·K···E·
0010  00 3e 8c a7 40 00 40 06  55 cf ac 1f 00 03 ac 1f   ·>··@·@· U·······
0020  00 02 1f 94 da e2 18 9e  04 0c 92 56 37 b1 80 18   ········ ···V7···
0030  01 fd 58 74 00 00 01 01  08 0a b4 cb 7f 99 f6 c2   ··Xt···· ········
0040  bb f1 63 68 61 74 5f 72  65 70 6c 79              ··chat_r eply
```

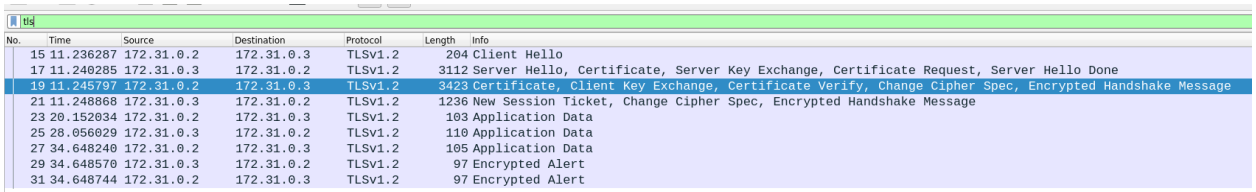
chat_STARTTLS :



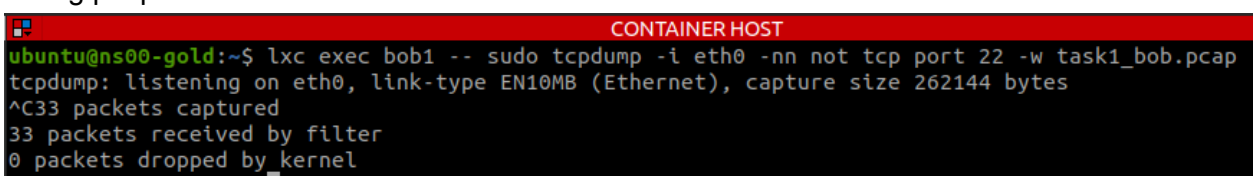
Receiving Ack on chat_STARTTLS



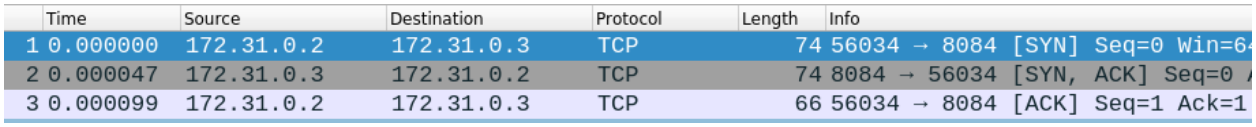
TLS Handshake and Data Transfer :



Taking pcap traces on Bob1 container :

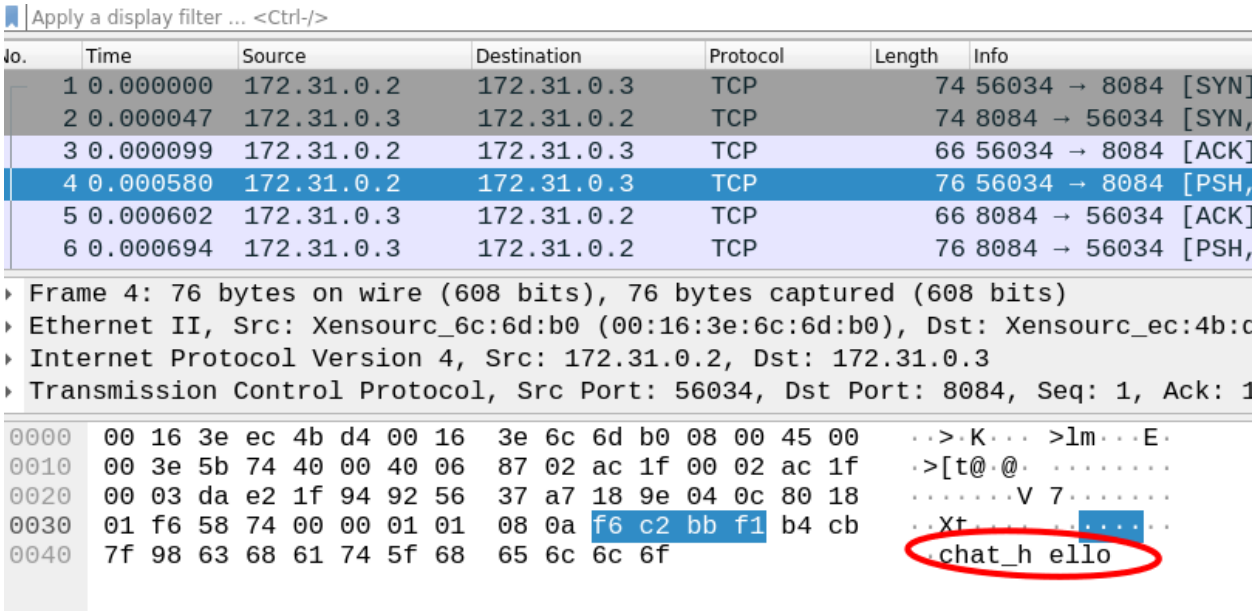


TCP Handshake :



	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.31.0.2	172.31.0.3	TCP	74	56034 → 8084 [SYN] Seq=0 Win=6
2	0.000047	172.31.0.3	172.31.0.2	TCP	74	8084 → 56034 [SYN, ACK] Seq=0
3	0.000099	172.31.0.2	172.31.0.3	TCP	66	56034 → 8084 [ACK] Seq=1 Ack=1

Got Chat hello from Alice :



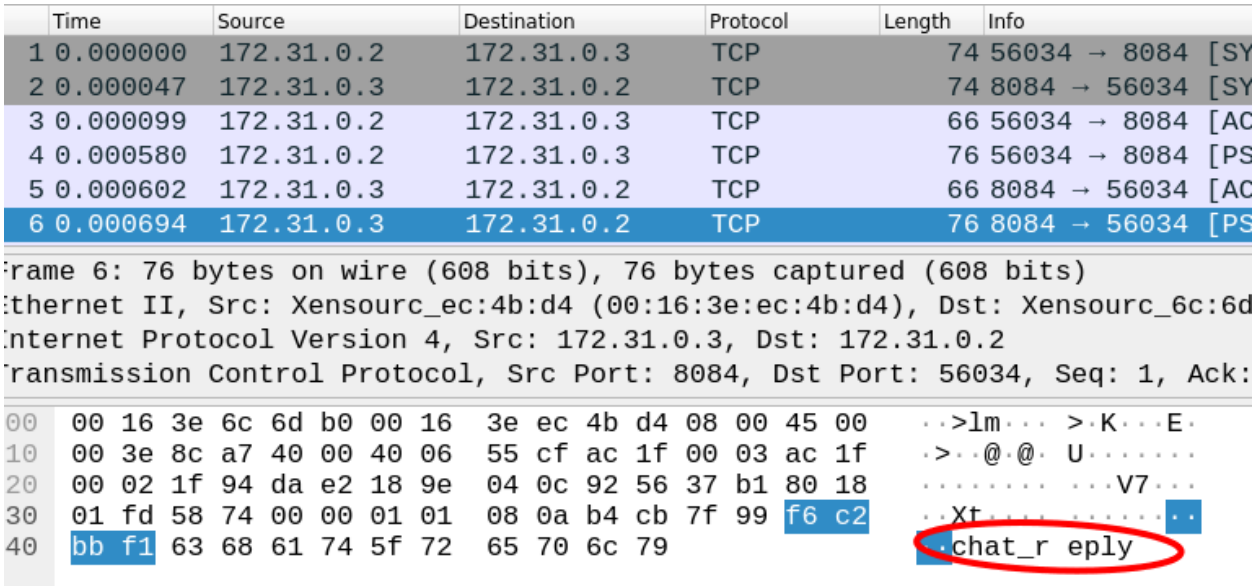
Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.31.0.2	172.31.0.3	TCP	74	56034 → 8084 [SYN]
2	0.000047	172.31.0.3	172.31.0.2	TCP	74	8084 → 56034 [SYN,
3	0.000099	172.31.0.2	172.31.0.3	TCP	66	56034 → 8084 [ACK]
4	0.000580	172.31.0.2	172.31.0.3	TCP	76	56034 → 8084 [PSH,
5	0.000602	172.31.0.3	172.31.0.2	TCP	66	8084 → 56034 [ACK]
6	0.000694	172.31.0.3	172.31.0.2	TCP	76	8084 → 56034 [PSH,

```
▸ Frame 4: 76 bytes on wire (608 bits), 76 bytes captured (608 bits)
▸ Ethernet II, Src: Xensourc_6c:6d:b0 (00:16:3e:6c:6d:b0), Dst: Xensourc_ec:4b:d
▸ Internet Protocol Version 4, Src: 172.31.0.2, Dst: 172.31.0.3
▸ Transmission Control Protocol, Src Port: 56034, Dst Port: 8084, Seq: 1, Ack: 1
```

```
0000  00 16 3e ec 4b d4 00 16  3e 6c 6d b0 08 00 45 00   ··>·K··· >lm···E·
0010  00 3e 5b 74 40 00 40 06  87 02 ac 1f 00 02 ac 1f   ·>[t@·@· ········
0020  00 03 da e2 1f 94 92 56  37 a7 18 9e 04 0c 80 18   ·······V 7·······
0030  01 f6 58 74 00 00 01 01  08 0a f6 c2 bb f1 b4 cb   ··Xt···· ········
0040  7f 98 63 68 61 74 5f 68  65 6c 6c 6f              ··chat_h ello
```

Sending chat reply to alice :

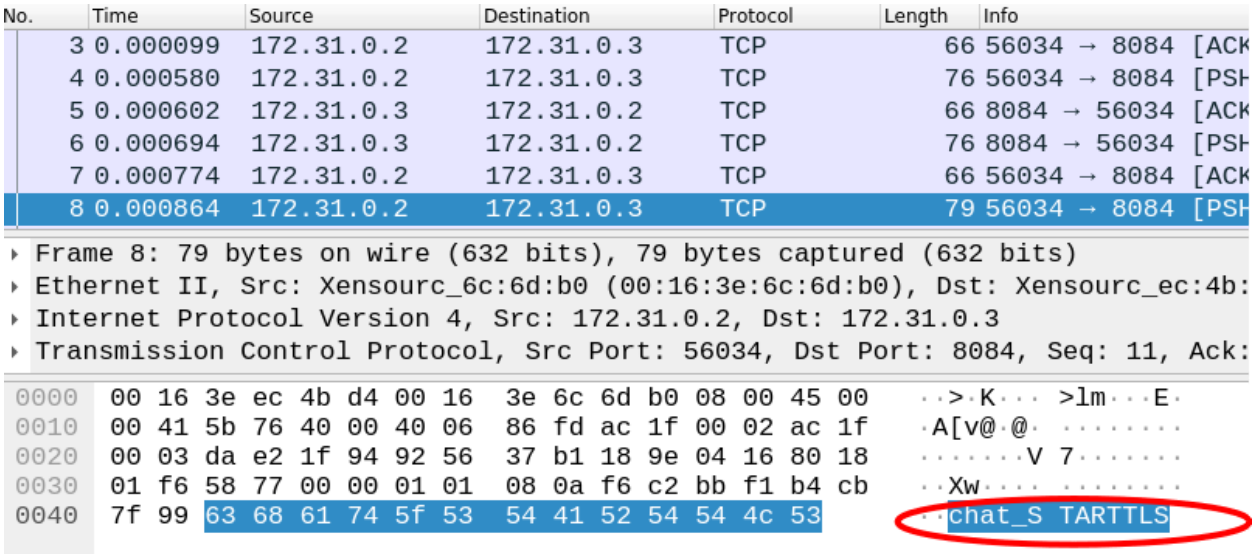


	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.31.0.2	172.31.0.3	TCP	74	56034 → 8084 [SY
2	0.000047	172.31.0.3	172.31.0.2	TCP	74	8084 → 56034 [SY
3	0.000099	172.31.0.2	172.31.0.3	TCP	66	56034 → 8084 [AC
4	0.000580	172.31.0.2	172.31.0.3	TCP	76	56034 → 8084 [PS
5	0.000602	172.31.0.3	172.31.0.2	TCP	66	8084 → 56034 [AC
6	0.000694	172.31.0.3	172.31.0.2	TCP	76	8084 → 56034 [PS

```
Frame 6: 76 bytes on wire (608 bits), 76 bytes captured (608 bits)
Ethernet II, Src: Xensourc_ec:4b:d4 (00:16:3e:ec:4b:d4), Dst: Xensourc_6c:6d
Internet Protocol Version 4, Src: 172.31.0.3, Dst: 172.31.0.2
Transmission Control Protocol, Src Port: 8084, Dst Port: 56034, Seq: 1, Ack:
```

```
00  00 16 3e 6c 6d b0 00 16  3e ec 4b d4 08 00 45 00   ··>lm··· >·K···E·
10  00 3e 8c a7 40 00 40 06  55 cf ac 1f 00 03 ac 1f   ·>··@·@· U·······
20  00 02 1f 94 da e2 18 9e  04 0c 92 56 37 b1 80 18   ········ ···V7···
30  01 fd 58 74 00 00 01 01  08 0a b4 cb 7f 99 f6 c2   ··Xt···· ········
40  bb f1 63 68 61 74 5f 72  65 70 6c 79              ··chat_r eply
```

Got a TLS request from Alice :

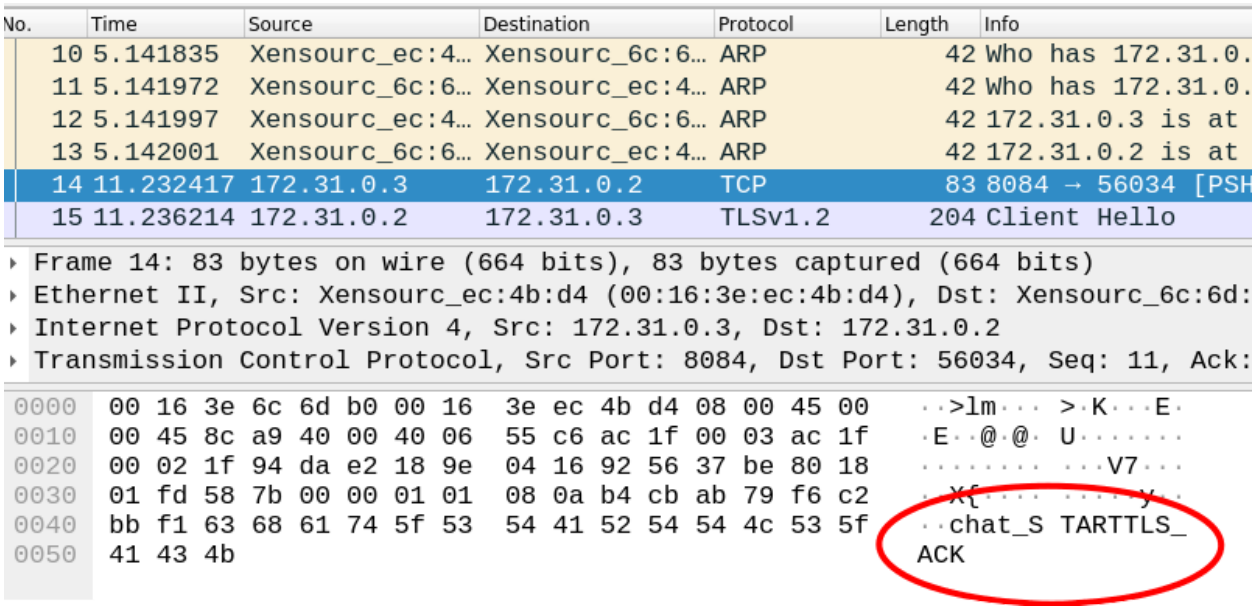


No.	Time	Source	Destination	Protocol	Length	Info
3	0.000099	172.31.0.2	172.31.0.3	TCP	66	56034 → 8084 [ACK
4	0.000580	172.31.0.2	172.31.0.3	TCP	76	56034 → 8084 [PSH
5	0.000602	172.31.0.3	172.31.0.2	TCP	66	8084 → 56034 [ACK
6	0.000694	172.31.0.3	172.31.0.2	TCP	76	8084 → 56034 [PSH
7	0.000774	172.31.0.2	172.31.0.3	TCP	66	56034 → 8084 [ACK
8	0.000864	172.31.0.2	172.31.0.3	TCP	79	56034 → 8084 [PSH

```
▸ Frame 8: 79 bytes on wire (632 bits), 79 bytes captured (632 bits)
▸ Ethernet II, Src: Xensourc_6c:6d:b0 (00:16:3e:6c:6d:b0), Dst: Xensourc_ec:4b:
▸ Internet Protocol Version 4, Src: 172.31.0.2, Dst: 172.31.0.3
▸ Transmission Control Protocol, Src Port: 56034, Dst Port: 8084, Seq: 11, Ack:
```

```
0000  00 16 3e ec 4b d4 00 16  3e 6c 6d b0 08 00 45 00   ··>·K··· >lm···E·
0010  00 41 5b 76 40 00 40 06  86 fd ac 1f 00 02 ac 1f   ·A[v@·@· ········
0020  00 03 da e2 1f 94 92 56  37 b1 18 9e 04 16 80 18   ·······V 7·······
0030  01 f6 58 77 00 00 01 01  08 0a f6 c2 bb f1 b4 cb   ··Xw···· ········
0040  7f 99 63 68 61 74 5f 53  54 41 52 54 54 4c 53      ··chat_S TARTTLS
```

Sending ACK to the TLS :

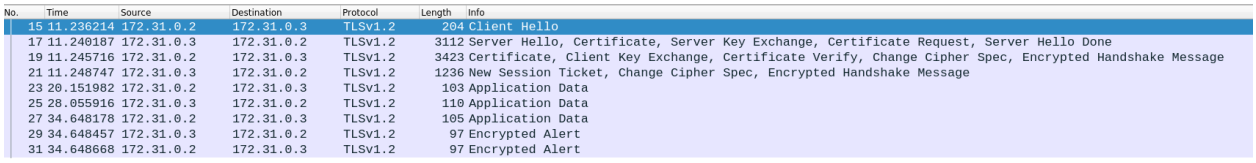


No.	Time	Source	Destination	Protocol	Length	Info
10	5.141835	Xensourc_ec:4…	Xensourc_6c:6…	ARP	42	Who has 172.31.0.
11	5.141972	Xensourc_6c:6…	Xensourc_ec:4…	ARP	42	Who has 172.31.0.
12	5.141997	Xensourc_ec:4…	Xensourc_6c:6…	ARP	42	172.31.0.3 is at
13	5.142001	Xensourc_6c:6…	Xensourc_ec:4…	ARP	42	172.31.0.2 is at
14	11.232417	172.31.0.3	172.31.0.2	TCP	83	8084 → 56034 [PSH
15	11.236214	172.31.0.2	172.31.0.3	TLSv1.2	204	Client Hello

```
▸ Frame 14: 83 bytes on wire (664 bits), 83 bytes captured (664 bits)
▸ Ethernet II, Src: Xensourc_ec:4b:d4 (00:16:3e:ec:4b:d4), Dst: Xensourc_6c:6d:
▸ Internet Protocol Version 4, Src: 172.31.0.3, Dst: 172.31.0.2
▸ Transmission Control Protocol, Src Port: 8084, Dst Port: 56034, Seq: 11, Ack:
```

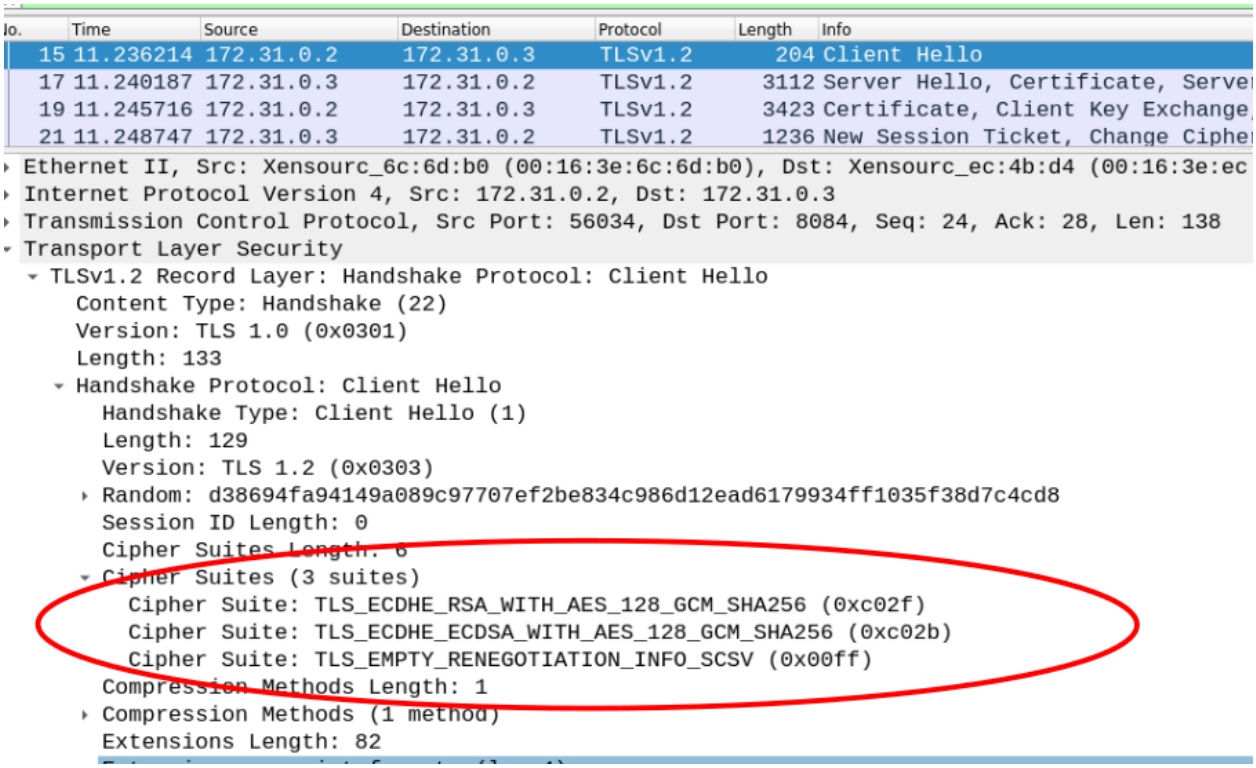
```
0000  00 16 3e 6c 6d b0 00 16  3e ec 4b d4 08 00 45 00   ··>lm··· >·K···E·
0010  00 45 8c a9 40 00 40 06  55 c6 ac 1f 00 03 ac 1f   ·E··@·@· U·······
0020  00 02 1f 94 da e2 18 9e  04 16 92 56 37 be 80 18   ········ ···V7···
0030  01 fd 58 7b 00 00 01 01  08 0a b4 cb ab 79 f6 c2   ··X{···· ·····y··
0040  bb f1 63 68 61 74 5f 53  54 41 52 54 54 4c 53 5f   ··chat_S TARTTLS_
0050  41 43 4b                                          ACK
```

TLS Handshake and Data Transfer on Bob's side :



No.	Time	Source	Destination	Protocol	Length	Info
15	11.236214	172.31.0.2	172.31.0.3	TLSv1.2	204	Client Hello
17	11.240187	172.31.0.3	172.31.0.2	TLSv1.2	3112	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
19	11.245716	172.31.0.2	172.31.0.3	TLSv1.2	3423	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
21	11.248747	172.31.0.3	172.31.0.2	TLSv1.2	1236	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
23	20.151982	172.31.0.2	172.31.0.3	TLSv1.2	103	Application Data
25	28.055916	172.31.0.3	172.31.0.2	TLSv1.2	110	Application Data
27	34.648178	172.31.0.2	172.31.0.3	TLSv1.2	105	Application Data
29	34.648457	172.31.0.3	172.31.0.2	TLSv1.2	97	Encrypted Alert
31	34.648668	172.31.0.2	172.31.0.3	TLSv1.2	97	Encrypted Alert

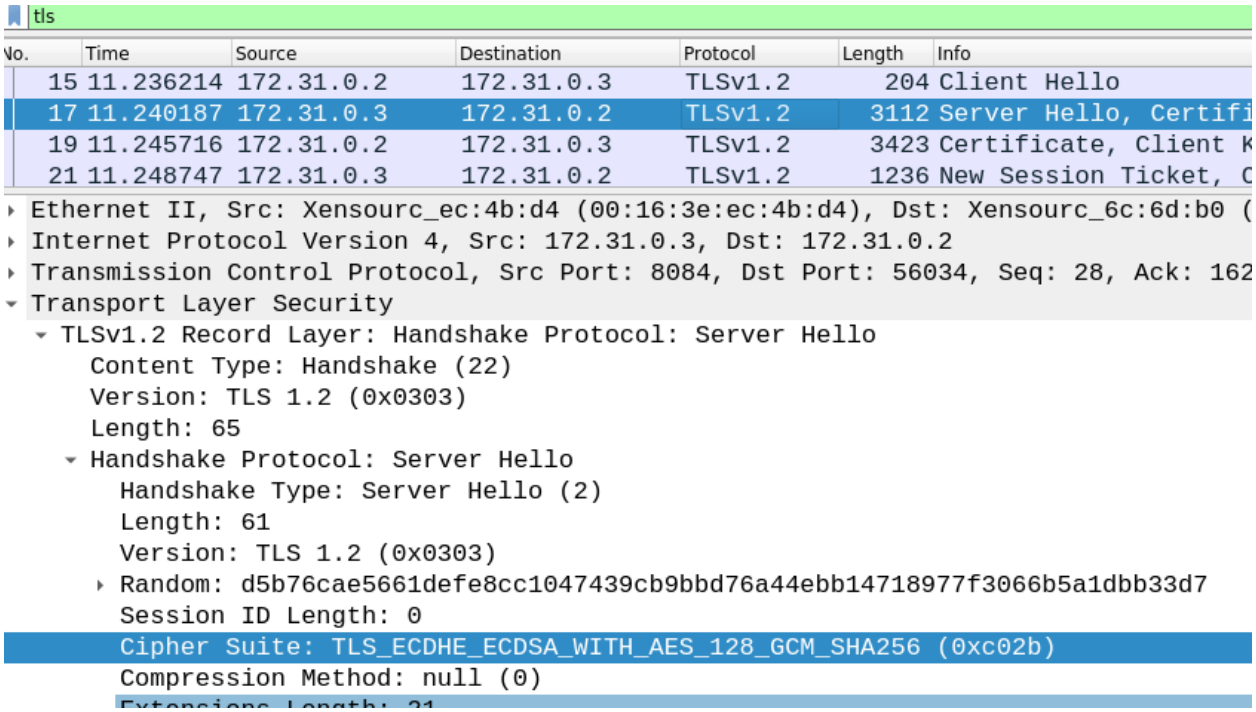
Cipher Suites that were set in our code that supports the PFS are shown in the client hello as follows :



This was set by using the following line of code in client() function :

```
(SSL_CTX_set_cipher_list(ctx, "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256")
```

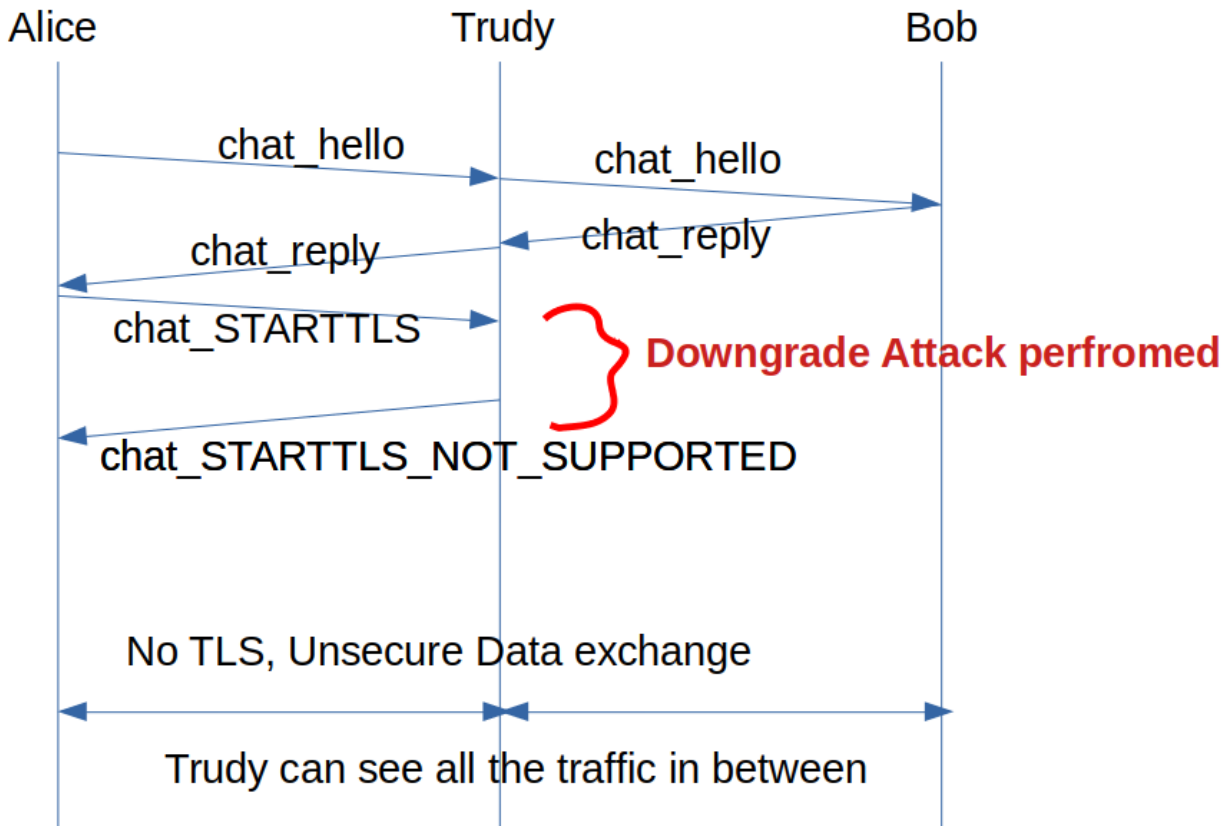
Server selects one of these(preconfigured in server as well) and selects the following Cipher suites among the 3 which the Client offered :



TASK 3 : Downgrade Attack

We can see from "**sec_server_client.cpp**" that anytime the message "**chat_STARTTLS_NOT_SUPPORTED**" is received, we set the secure variable to "FALSE." For example, it is presumed that Bob does not want to conduct a secure chat conversation if Alice receives a "**chat_STARTTLS_NOT_SUPPORTED**" message after sending "**chat_STARTTLS**" to Bob.

Downgrade Attack is done as follows :



- We conducted the attack using the method downGrade() in the "**sec_server_client.cpp**".
- The socket is initially generated on the client side and is set to listen for connections using the listen address and port=8084 parameters.
- Hence, the false Trudy server receives the connection message sent by the client when it tries to connect to the server.
- All is well until Alice tries to establish a TLS connection and sends the command "chat STARTTLS" to the server. At that point, our function interjects and replies with the message "chat STARTTLS NOT SUPPORTED," eventually disabling all TLS connections between client and server.

We perform the downgrade attack by using command :

```
g++ downgrade.c++ -lssl -lcrypto -o dg
```

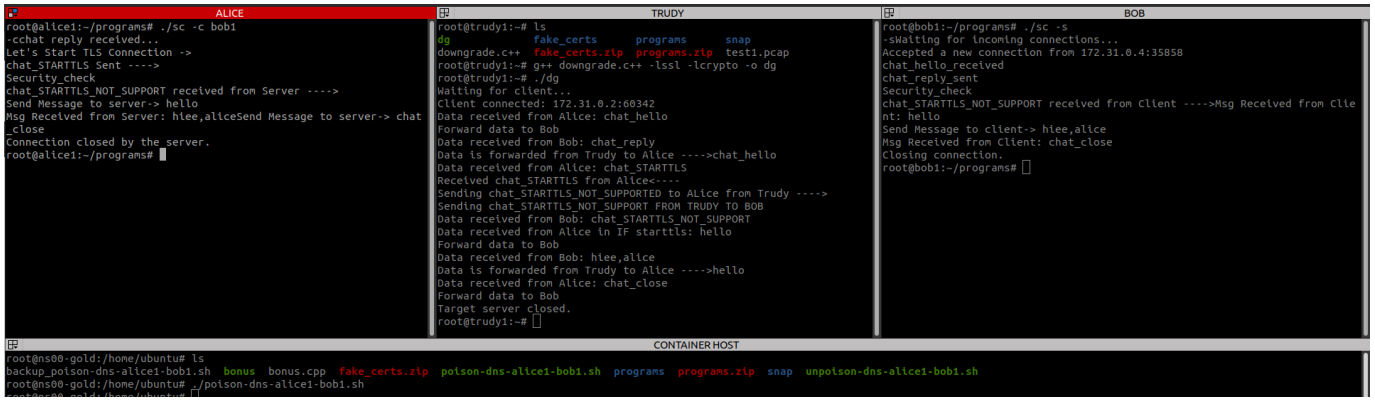
We first run the DNS Poisoning script in Container Host :

```
./poison-dns-alice1-bob1.sh
```

By running this script, Trudy has become the MITM, because the DNS resolvers at Alice and Bob side i.e “/etc/hosts” were changed by this script.

As a result, Alice and Bob's DNS have been contaminated, and their IP addresses have been changed. Then, in the container Trudy, we run the file "sec_server_client.cpp" to create a fake client and server socket.

Entire call flow can be seen in the command line below :



In fact, we are operating in an unsafe manner as we run "sec_server_client.cpp" to create a client-server communication between Alice and Bob.

The link between Alice and Bob is split as Alice and Trudy and between Trudy and Bob.

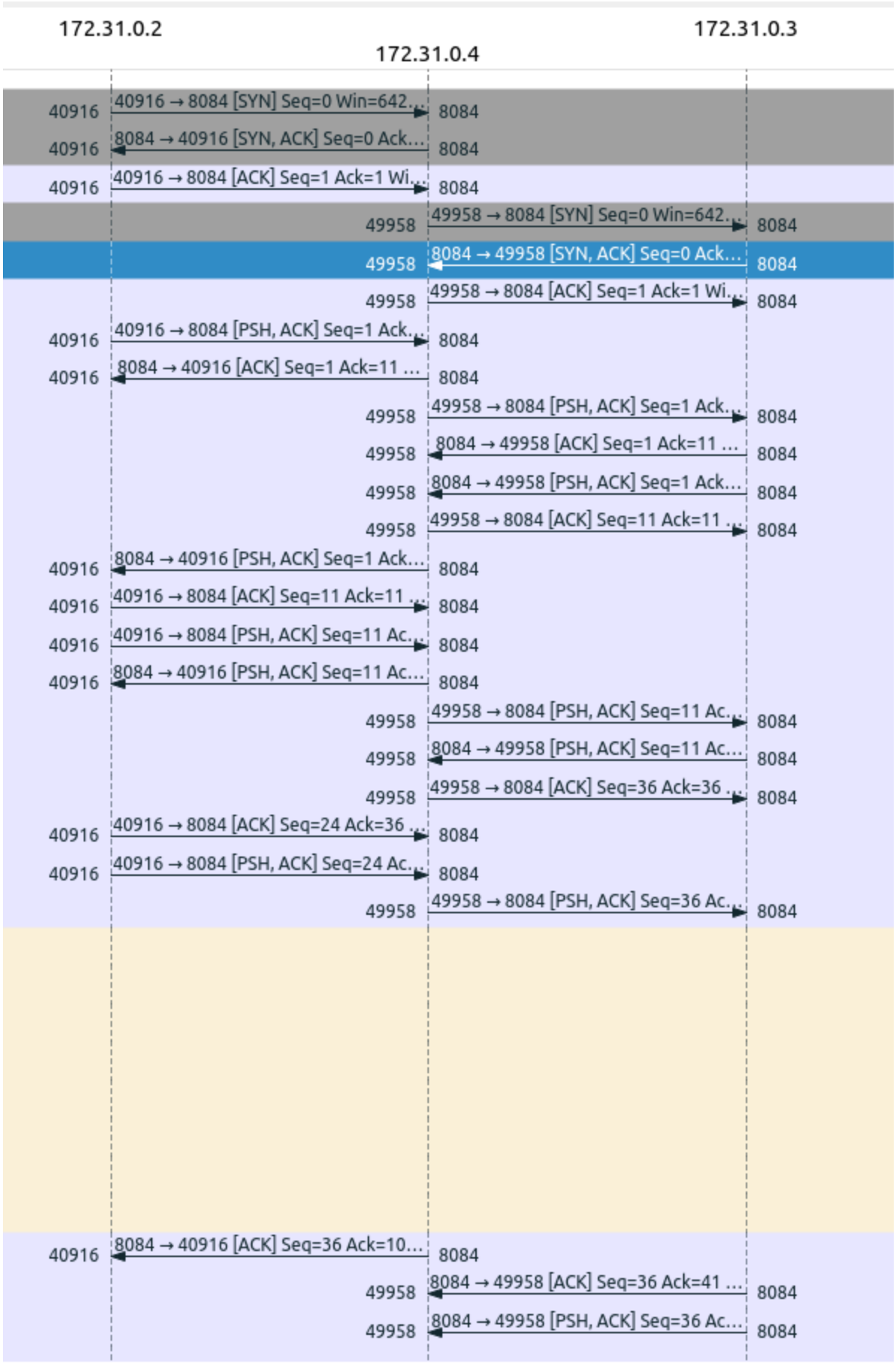
When Alice sends Trudy a "chat hello" message, she simply forwards it to Bob. Likewise, when Bob sends Trudy a "chat reply," she simply transfers the message to Alice.

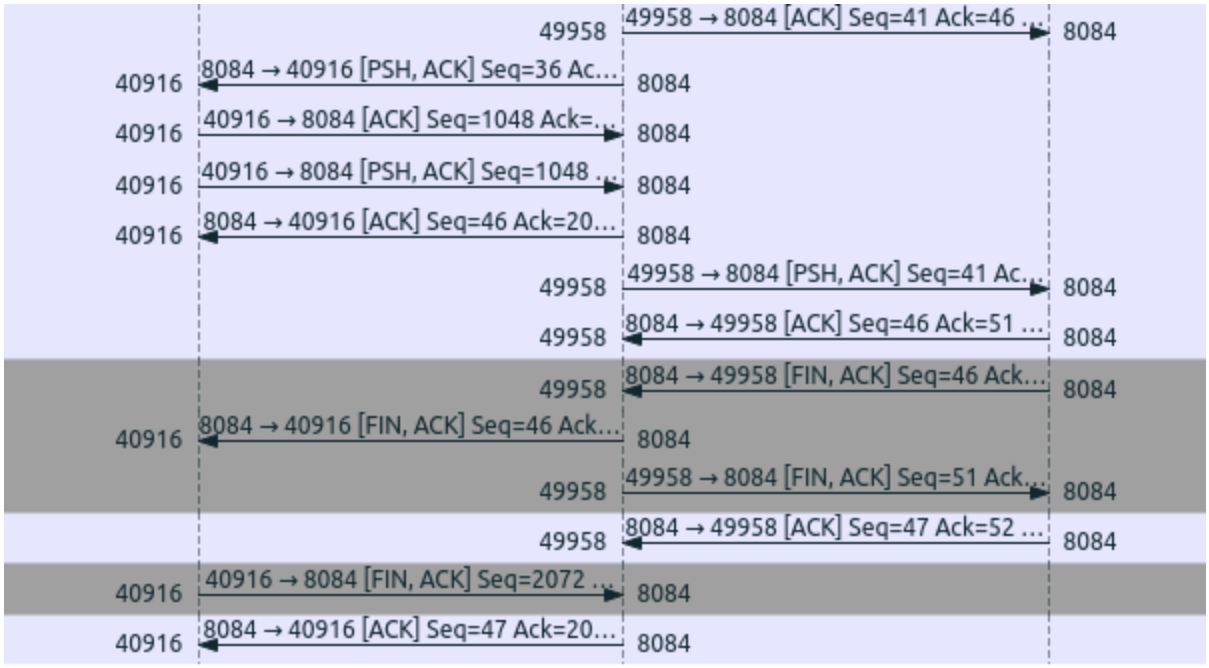
When Trudy receives the chat STARTTLS from Alice, it blocks the message from Bob and sends chat STARTTLS NOT SUPPORTED to Alice.

As TLS cannot be started at this time, ALice will continue with insecure mode chat and complete the application data transfer between the two sites. This is Trudy's downgrade attack.

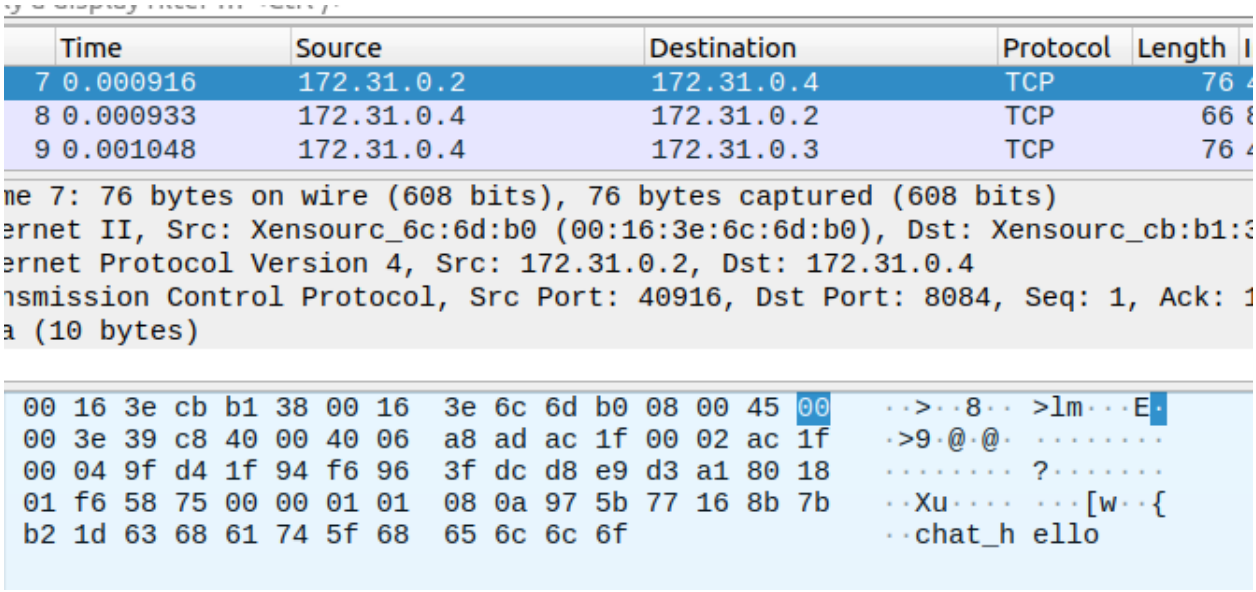
PCAP traces at Trudy :

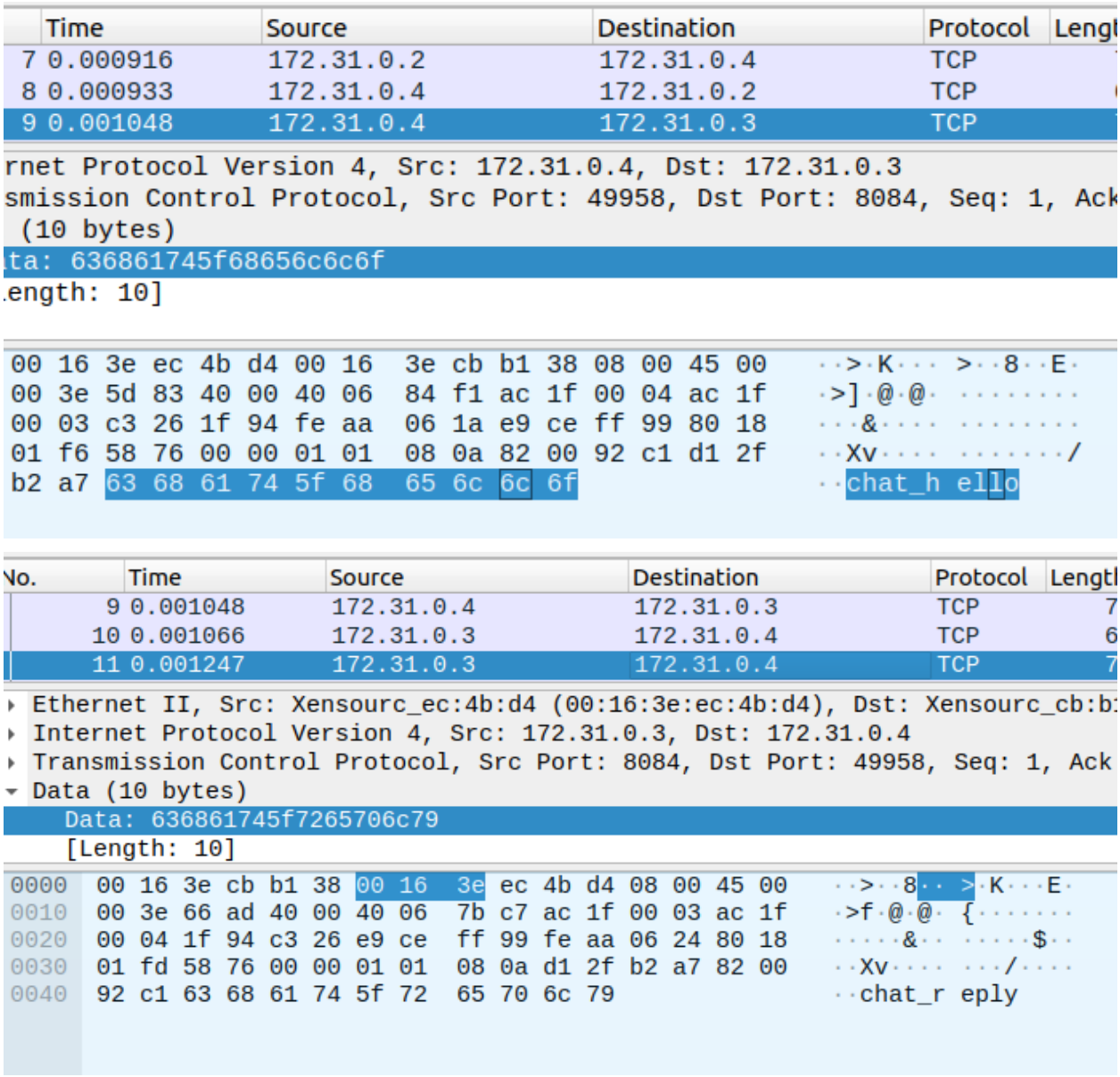
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.31.0.2	172.31.0.4	TCP	74	40916 → 8084 [
2	0.000061	172.31.0.4	172.31.0.2	TCP	74	8084 → 40916 [
3	0.000101	172.31.0.2	172.31.0.4	TCP	66	40916 → 8084 [
4	0.000295	172.31.0.4	172.31.0.3	TCP	74	49958 → 8084 [
5	0.000346	172.31.0.3	172.31.0.4	TCP	74	8084 → 49958 [
6	0.000365	172.31.0.4	172.31.0.3	TCP	66	49958 → 8084 [
7	0.000916	172.31.0.2	172.31.0.4	TCP	76	40916 → 8084 [
8	0.000933	172.31.0.4	172.31.0.2	TCP	66	8084 → 40916 [
9	0.001048	172.31.0.4	172.31.0.3	TCP	76	49958 → 8084 [
10	0.001066	172.31.0.3	172.31.0.4	TCP	66	8084 → 49958 [
11	0.001247	172.31.0.3	172.31.0.4	TCP	76	8084 → 49958 [
12	0.001257	172.31.0.4	172.31.0.3	TCP	66	49958 → 8084 [
13	0.001316	172.31.0.4	172.31.0.2	TCP	76	8084 → 40916 [
14	0.002159	172.31.0.2	172.31.0.4	TCP	66	40916 → 8084 [
15	0.002161	172.31.0.2	172.31.0.4	TCP	79	40916 → 8084 [
16	0.002452	172.31.0.4	172.31.0.2	TCP	91	8084 → 40916 [
17	0.002497	172.31.0.4	172.31.0.3	TCP	91	49958 → 8084 [
18	0.002561	172.31.0.3	172.31.0.4	TCP	91	8084 → 49958 [
19	0.046059	172.31.0.4	172.31.0.3	TCP	66	49958 → 8084 [
20	0.046093	172.31.0.2	172.31.0.4	TCP	66	40916 → 8084 [
21	5.046240	172.31.0.2	172.31.0.4	TCP	1090	40916 → 8084 [
22	5.046389	172.31.0.4	172.31.0.3	TCP	71	49958 → 8084 [
23	5.066081	Xensourc_cb:b1:38	Xensourc_ec:4b:d4	ARP	42	Who has 172.31
24	5.066101	Xensourc_cb:b1:38	Xensourc_6c:6d:b0	ARP	42	Who has 172.31
25	5.066386	Xensourc_ec:4b:d4	Xensourc_cb:b1:38	ARP	42	Who has 172.31
26	5.066405	Xensourc_cb:b1:38	Xensourc_ec:4b:d4	ARP	42	172.31.0.4 is
27	5.066392	Xensourc_6c:6d:b0	Xensourc_cb:b1:38	ARP	42	Who has 172.31
28	5.066418	Xensourc_cb:b1:38	Xensourc_6c:6d:b0	ARP	42	172.31.0.4 is
29	5.066422	Xensourc_ec:4b:d4	Xensourc_cb:b1:38	ARP	42	172.31.0.3 is
30	5.066423	Xensourc_6c:6d:b0	Xensourc_cb:b1:38	ARP	42	172.31.0.2 is
31	5.090131	172.31.0.4	172.31.0.2	TCP	66	8084 → 40916 [
32	5.090151	172.31.0.3	172.31.0.4	TCP	66	8084 → 49958 [
33	10.949625	172.31.0.3	172.31.0.4	TCP	76	8084 → 49958 [
34	10.949665	172.31.0.4	172.31.0.3	TCP	66	49958 → 8084 [
35	10.949797	172.31.0.4	172.31.0.2	TCP	76	8084 → 40916 [
36	10.949841	172.31.0.2	172.31.0.4	TCP	66	40916 → 8084 [
37	15.126839	172.31.0.2	172.31.0.4	TCP	1090	40916 → 8084 [
38	15.126884	172.31.0.4	172.31.0.2	TCP	66	8084 → 40916 [
39	15.127035	172.31.0.4	172.31.0.3	TCP	76	49958 → 8084 [
40	15.127072	172.31.0.3	172.31.0.4	TCP	66	8084 → 49958 [
41	15.127209	172.31.0.3	172.31.0.4	TCP	66	8084 → 49958 [
42	15.128276	172.31.0.4	172.31.0.2	TCP	66	8084 → 40916 [
43	15.128359	172.31.0.4	172.31.0.3	TCP	66	49958 → 8084 [
44	15.128391	172.31.0.3	172.31.0.4	TCP	66	8084 → 49958 [
45	15.129300	172.31.0.2	172.31.0.4	TCP	66	40916 → 8084 [
46	15.129322	172.31.0.4	172.31.0.2	TCP	66	8084 → 40916 [

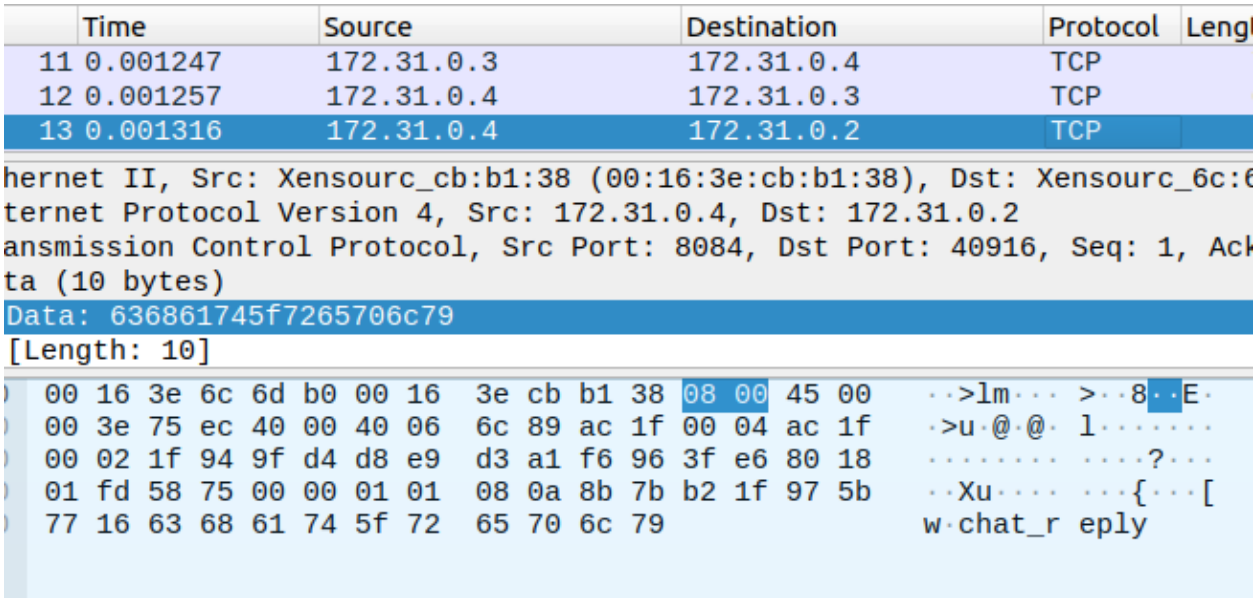

172.31.0.2
172.31.0.4
172.31.0.3
40916 40916 → 8084 [SYN] Seq=0 Win=642… 8084
40916 8084 → 40916 [SYN, ACK] Seq=0 Ack… 8084
40916 40916 → 8084 [ACK] Seq=1 Ack=1 Wi… 8084
49958 49958 → 8084 [SYN] Seq=0 Win=642… 8084
49958 8084 → 49958 [SYN, ACK] Seq=0 Ack… 8084
49958 49958 → 8084 [ACK] Seq=1 Ack=1 Wi… 8084
40916 40916 → 8084 [PSH, ACK] Seq=1 Ack… 8084
40916 8084 → 40916 [ACK] Seq=1 Ack=11 … 8084
49958 49958 → 8084 [PSH, ACK] Seq=1 Ack… 8084
49958 8084 → 49958 [ACK] Seq=1 Ack=11 … 8084
49958 8084 → 49958 [PSH, ACK] Seq=1 Ack… 8084
49958 49958 → 8084 [ACK] Seq=11 Ack=11 … 8084
40916 8084 → 40916 [PSH, ACK] Seq=1 Ack… 8084
40916 40916 → 8084 [ACK] Seq=11 Ack=11 … 8084
40916 40916 → 8084 [PSH, ACK] Seq=11 Ac… 8084
40916 8084 → 40916 [PSH, ACK] Seq=11 Ac… 8084
49958 49958 → 8084 [PSH, ACK] Seq=11 Ac… 8084
49958 8084 → 49958 [PSH, ACK] Seq=11 Ac… 8084
49958 49958 → 8084 [ACK] Seq=36 Ack=36 … 8084
40916 40916 → 8084 [ACK] Seq=24 Ack=36 … 8084
40916 40916 → 8084 [PSH, ACK] Seq=24 Ac… 8084
49958 49958 → 8084 [PSH, ACK] Seq=36 Ac… 8084
40916 8084 → 40916 [ACK] Seq=36 Ack=10… 8084
49958 8084 → 49958 [ACK] Seq=36 Ack=41 … 8084
49958 8084 → 49958 [PSH, ACK] Seq=36 Ac… 8084

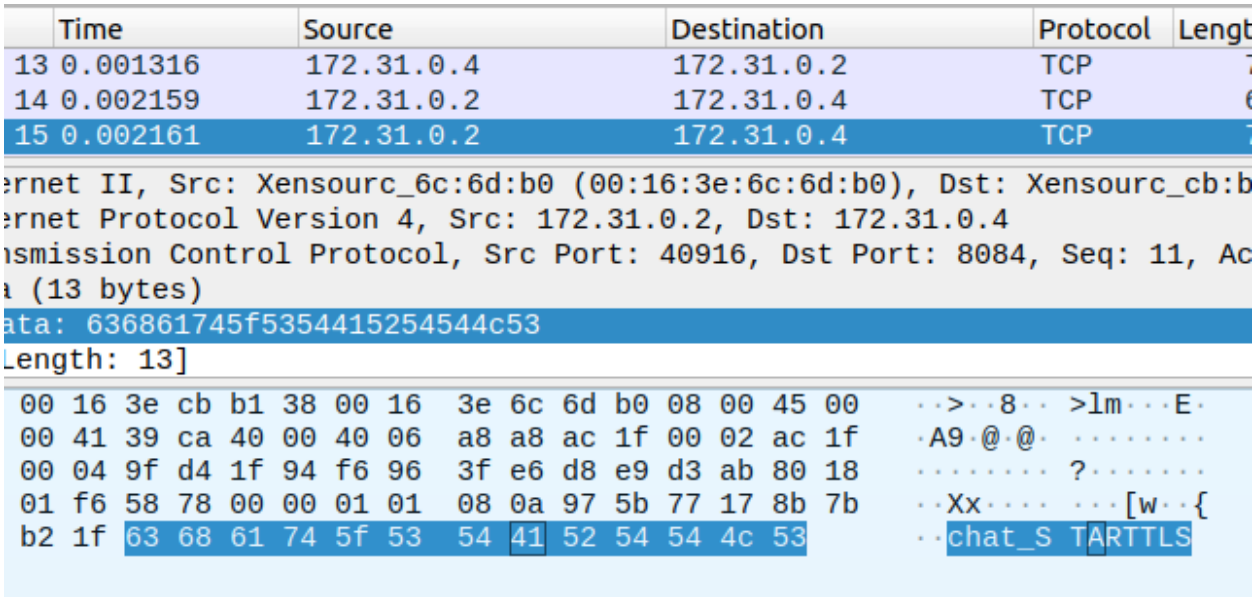


As you can see, there is no TLS protocol used as a downgrade was performed by Trudy. Important Messages in the call flow according to the call flow diagram :

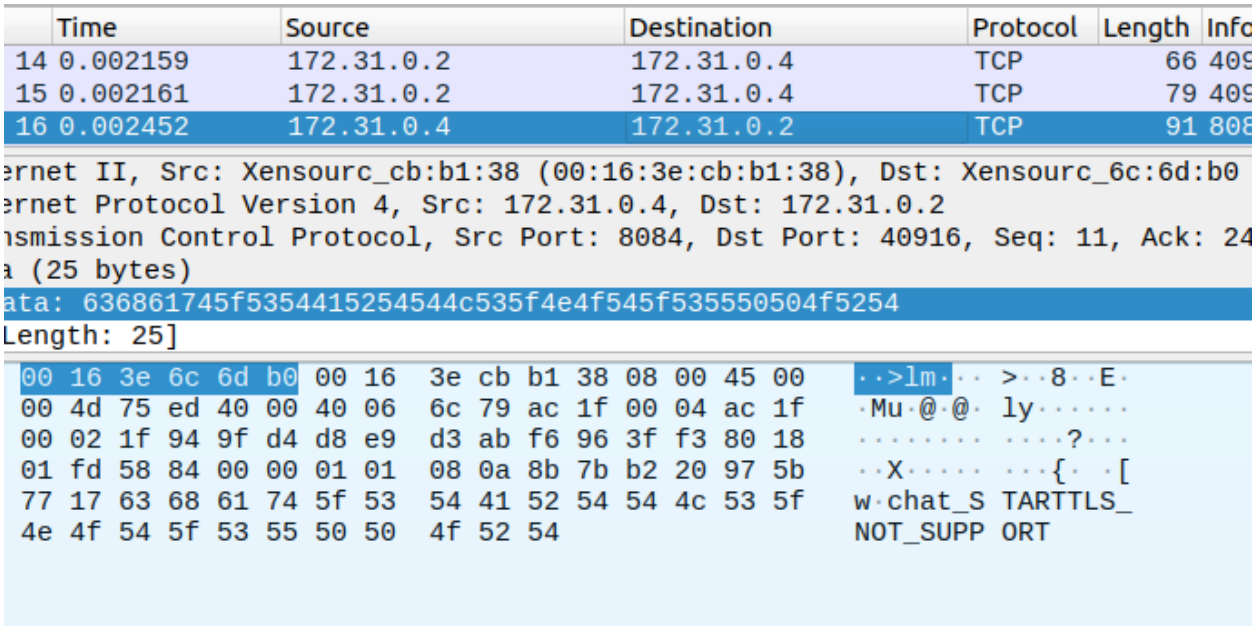



Time Source Destination Protocol Lengt
7 0.000916 172.31.0.2 172.31.0.4 TCP
8 0.000933 172.31.0.4 172.31.0.2 TCP
9 0.001048 172.31.0.4 172.31.0.3 TCP
rnet Protocol Version 4, Src: 172.31.0.4, Dst: 172.31.0.3
smission Control Protocol, Src Port: 49958, Dst Port: 8084, Seq: 1, Ack
(10 bytes)
ta: 636861745f68656c6c6f
ength: 10]
00 16 3e ec 4b d4 00 16 3e cb b1 38 08 00 45 00 ··>·K··· >··8··E·
00 3e 5d 83 40 00 40 06 84 f1 ac 1f 00 04 ac 1f ·>]·@·@· ····· ····
00 03 c3 26 1f 94 fe aa 06 1a e9 ce ff 99 80 18 ···&···· ····· ····
01 f6 58 76 00 00 01 01 08 0a 82 00 92 c1 d1 2f ··Xv···· ····· ···/
b2 a7 63 68 61 74 5f 68 65 6c 6c 6f ··chat_h ello
No. Time Source Destination Protocol Lengt
9 0.001048 172.31.0.4 172.31.0.3 TCP 7
10 0.001066 172.31.0.3 172.31.0.4 TCP 6
11 0.001247 172.31.0.3 172.31.0.4 TCP 7
Ethernet II, Src: Xensourc_ec:4b:d4 (00:16:3e:ec:4b:d4), Dst: Xensourc_cb:b
Internet Protocol Version 4, Src: 172.31.0.3, Dst: 172.31.0.4
Transmission Control Protocol, Src Port: 8084, Dst Port: 49958, Seq: 1, Ack
Data (10 bytes)
Data: 636861745f7265706c79
[Length: 10]
0000 00 16 3e cb b1 38 00 16 3e ec 4b d4 08 00 45 00 ··>··8·· >·K···E·
0010 00 3e 66 ad 40 00 40 06 7b c7 ac 1f 00 03 ac 1f ·>f·@·@· {·····
0020 00 04 1f 94 c3 26 e9 ce ff 99 fe aa 06 24 80 18 ·····&·· ·····$··
0030 01 fd 58 76 00 00 01 01 08 0a d1 2f b2 a7 82 00 ··Xv···· ···/····
0040 92 c1 63 68 61 74 5f 72 65 70 6c 79 ··chat_r eply

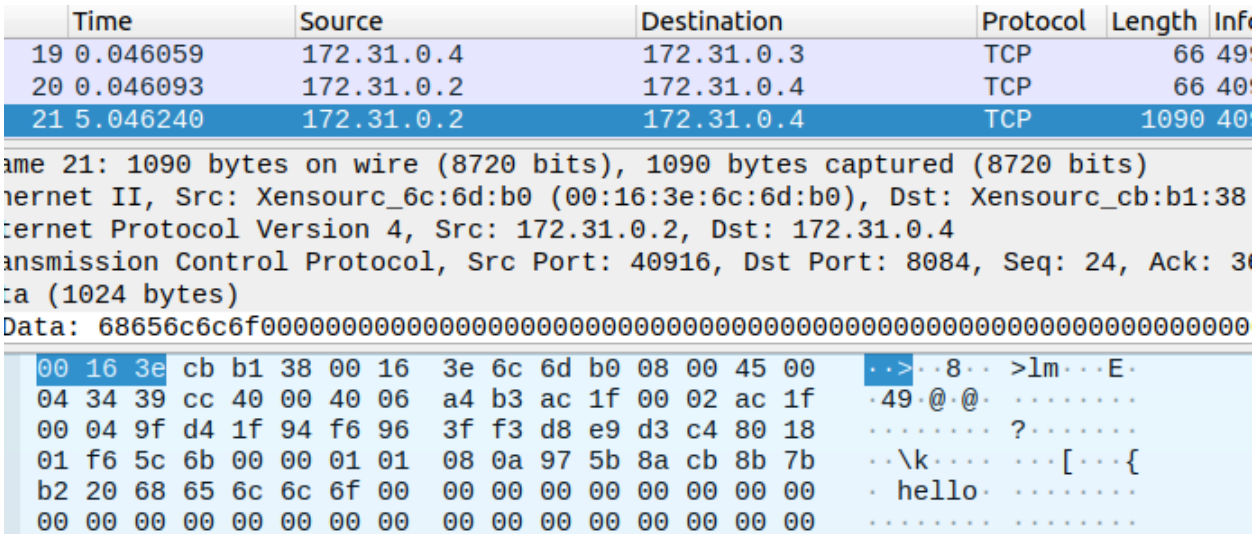


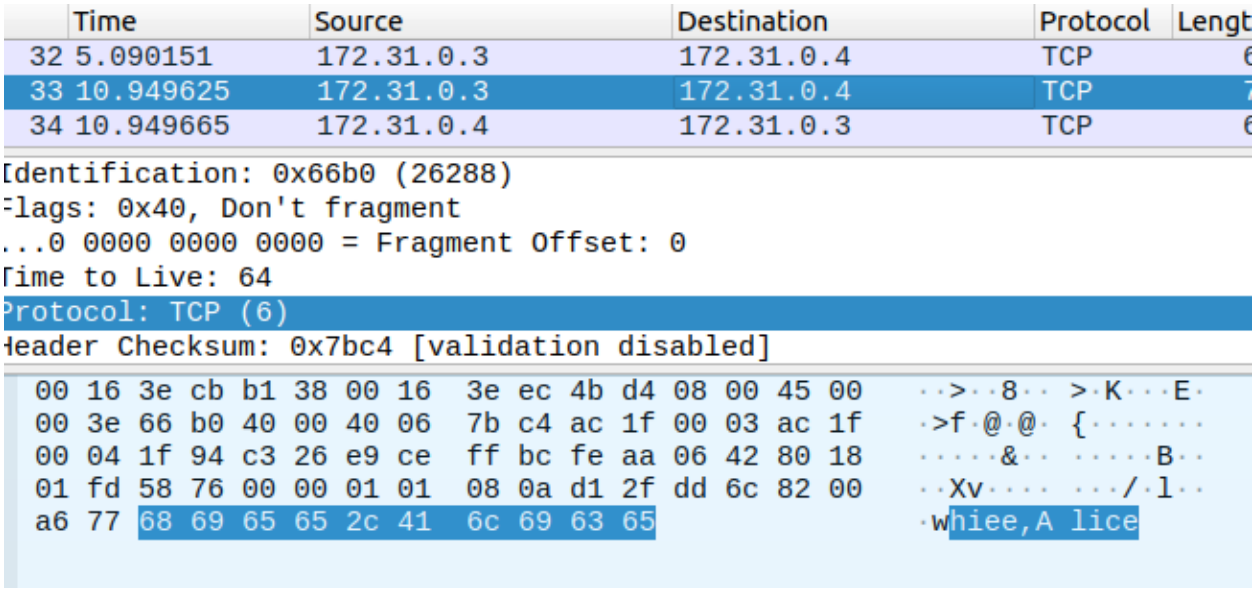


Trudy sends TLS not supported message acting as Bob to Client :



Thus he can decrypt all the messages :





TASK 4 : Downgrade Attack [Active MITM Attack]

This attack is different from the downgrade attack of Task 3. In this, TLS is used, despite that, Trudy is able to decrypt all the messages in between Alice and Bob as she was able to compromise the Intermediate CA. So, he can issue fake certificates and thus possesses the keys of both the connection, towards the client and towards the server, thus can decrypt the traffic in between.

We will use an active MITM attack in this work to mess with Alice and Bob's chat communication. First, we're going to poison the DNS and change the IP addresses such that Trudy serves as Alice's server and Bob's client.

```
./poison-dns-alice1-bob1.sh
```

We use the “downgrade.c++_task4” instead of mitm() :
The Trudy client and server certificates are initially loaded, and a server-side socket is constructed, tied to a port, and listening to see if any clients are connecting.

When the client sends "chat STARTTLS" to the server in an attempt to create a TLS connection, Trudy replies with "chat STARTTLS ACK," and the client SSL connection is formed.

Similar to before, Trudy now assumes the role of the client and requests a TLS connection from the server. When the server responds with the STARTTLS ACK, the SSL connection to the

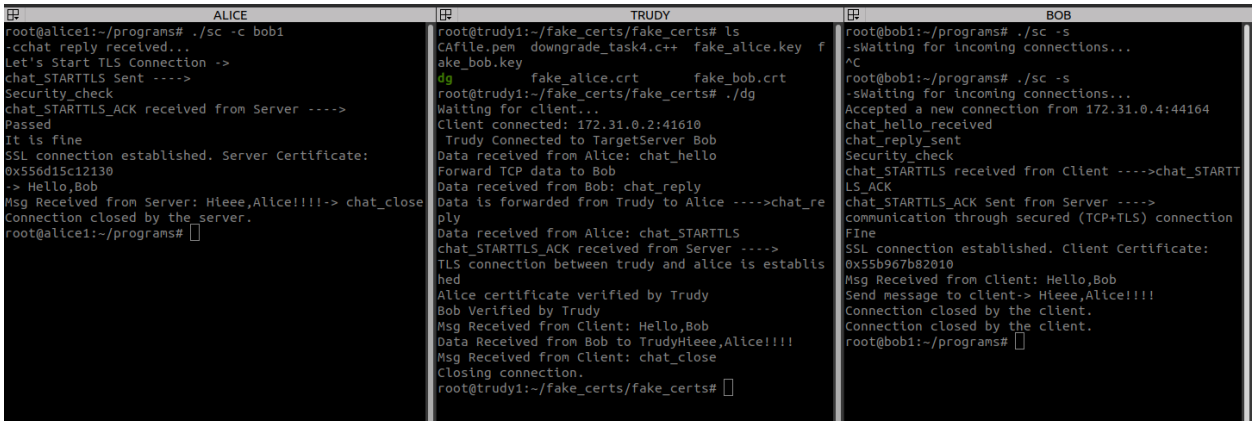
server is established.

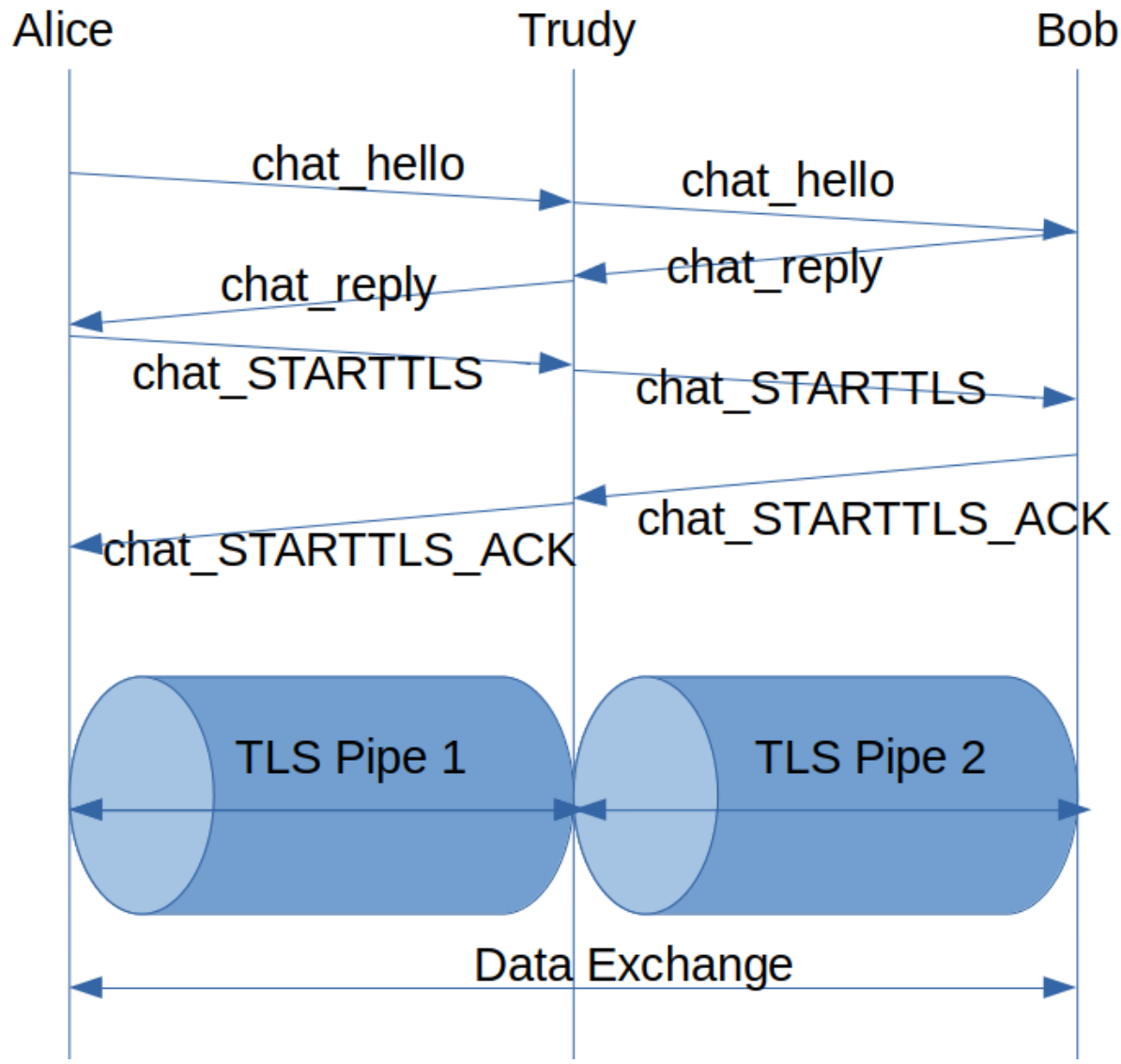
Trudy may now alter texts and data on both sides.
Using the following command, the downgrade is carried out at Trudy container :

```
g++ downgrade_task4.c++ -lssl -lcrypto -o dg
```

Fakebob.crt and fakealice.crt certificates are created by Trudy with key-pairs in .pem files. These thus are verified with the help of openssl. Instead of launching the downgrade in task3, Trudys' certificate gets verified and Client server TLS handshake is carried out. Thus a TLS pipe is established between client and server but in reality there are 2 TLS pipes established : Between Alice and Trudy, and between Trudy and bob.

Call flow in the command line :





Now because Trudy is the MITM, he can tamper any messages in between Alice and Bob. This is possible because the CA is compromised and trudy was able to make fake certificates.

Alice sent chat_STARTTLS message thinking its sending to bob and it is then received by Trudy. chat_STARTTLS_ACK is received by Trudy from Bob and it then sends it to Alice in behalf of bob. A handshake takes place with the help of fake certificates.

Trudy gets chat_hello :

	Time	Source	Destination	Protocol	Length	Info
7	0.000709	172.31.0.2	172.31.0.4	TCP	76	41610
8	0.000717	172.31.0.4	172.31.0.2	TCP	66	8084
9	0.000762	172.31.0.4	172.31.0.3	TCP	76	44164
10	0.000772	172.31.0.3	172.31.0.4	TCP	66	8084
11	0.000884	172.31.0.3	172.31.0.4	TCP	76	8084
12	0.000900	172.31.0.4	172.31.0.3	TCP	66	44164
13	0.000922	172.31.0.4	172.31.0.2	TCP	76	8084
14	0.000935	172.31.0.2	172.31.0.4	TCP	66	41610
15	0.001255	172.31.0.2	172.31.0.4	TCP	79	41610

```
ame 7: 76 bytes on wire (608 bits), 76 bytes captured (608 bits)
thernet II, Src: Xensourc_6c:6d:b0 (00:16:3e:6c:6d:b0), Dst: Xensourc_cb:b1:38 (0
nternet Protocol Version 4, Src: 172.31.0.2, Dst: 172.31.0.4
ransmission Control Protocol, Src Port: 41610, Dst Port: 8084, Seq: 1, Ack: 1, Le
ata (10 bytes)
```

```
0  00 16 3e cb b1 38 00 16  3e 6c 6d b0 08 00 45 00   ··>··8·· >lm···E·
0  00 3e 84 67 40 00 40 06  5e 0e ac 1f 00 02 ac 1f   ·>·g@·@· ^·······
0  00 04 a2 8a 1f 94 d2 21  a5 9e ac dc 09 02 80 18   ·······! ········
0  01 f6 58 75 00 00 01 01  08 0a 97 75 86 f7 8b 95   ··Xu···· ···u····
0  c1 ff 63 68 61 74 5f 68  65 6c 6c 6f               ··chat_h ello
```

Gets chat reply from bob :

	Time	Source	Destination	Protocol	Length	Info
6	0.000618	172.31.0.4	172.31.0.3	TCP	66	441
7	0.000709	172.31.0.2	172.31.0.4	TCP	76	416
8	0.000717	172.31.0.4	172.31.0.2	TCP	66	808
9	0.000762	172.31.0.4	172.31.0.3	TCP	76	441
10	0.000772	172.31.0.3	172.31.0.4	TCP	66	808
11	0.000884	172.31.0.3	172.31.0.4	TCP	76	808
12	0.000900	172.31.0.4	172.31.0.3	TCP	66	441
13	0.000922	172.31.0.4	172.31.0.2	TCP	76	808
14	0.000935	172.31.0.2	172.31.0.4	TCP	66	416

```
me 11: 76 bytes on wire (608 bits), 76 bytes captured (608 bits)
ernet II, Src: Xensourc_ec:4b:d4 (00:16:3e:ec:4b:d4), Dst: Xensourc_cb:b1:38
ernet Protocol Version 4, Src: 172.31.0.3, Dst: 172.31.0.4
nsmission Control Protocol, Src Port: 8084, Dst Port: 44164, Seq: 1, Ack: 11,
a (10 bytes)
```

```
00 16 3e cb b1 38 00 16  3e ec 4b d4 08 00 45 00   ··>··8·· >·K···E·
00 3e 1f 97 40 00 40 06  c2 dd ac 1f 00 03 ac 1f   ·>··@·@· ········
00 04 1f 94 ac 84 1f 16  b8 46 72 71 0a 5e 80 18   ········ ·Frq·^··
01 fd 58 76 00 00 01 01  08 0a d1 49 c2 89 82 1a   ··Xv···· ···I····
a2 a3 63 68 61 74 5f 72  65 70 6c 79               ··chat_r eply
```

	Time	Source	Destination	Protocol	Lengt
15	0.001255	172.31.0.2	172.31.0.4	TCP	
16	0.001284	172.31.0.4	172.31.0.3	TCP	
17	0.044571	172.31.0.4	172.31.0.2	TCP	
18	0.044593	172.31.0.3	172.31.0.4	TCP	

```
ame 15: 79 bytes on wire (632 bits), 79 bytes captured (632 bits)
nernet II, Src: Xensourc_6c:6d:b0 (00:16:3e:6c:6d:b0), Dst: Xensourc_cb:b
ternet Protocol Version 4, Src: 172.31.0.2, Dst: 172.31.0.4
ansmission Control Protocol, Src Port: 41610, Dst Port: 8084, Seq: 11, Ac
ta (13 bytes)
```

```
00 16 3e cb b1 38 00 16  3e 6c 6d b0 08 00 45 00   ··>··8·· >lm···E·
00 41 84 69 40 00 40 06  5e 09 ac 1f 00 02 ac 1f   ·A·i@·@· ^·······
00 04 a2 8a 1f 94 d2 21  a5 a8 ac dc 09 0c 80 18   ·······! ········
01 f6 58 78 00 00 01 01  08 0a 97 75 86 f8 8b 95   ··Xx···· ···u····
c2 00 63 68 61 74 5f 53  54 41 52 54 54 4c 53      ··chat_S TARTTLS
```

Gets ACK from Bob which it then forwards to Alice :

	Time	Source	Destination	Protocol	Length	In
25	5.096820	Xensourc_ec:4b:d4	Xensourc_cb:b1:38	ARP	42	1
26	5.096823	Xensourc_6c:6d:b0	Xensourc_cb:b1:38	ARP	42	1
27	7.723739	172.31.0.3	172.31.0.4	TCP	83	8
28	7.723906	172.31.0.4	172.31.0.2	TCP	83	8

```
ame 27: 83 bytes on wire (664 bits), 83 bytes captured (664 bits)
nernet II, Src: Xensourc_ec:4b:d4 (00:16:3e:ec:4b:d4), Dst: Xensourc_cb:b1:3
ternet Protocol Version 4, Src: 172.31.0.3, Dst: 172.31.0.4
ansmission Control Protocol, Src Port: 8084, Dst Port: 44164, Seq: 11, Ack:
ta (17 bytes)
```

```
00 16 3e cb b1 38 00 16  3e ec 4b d4 08 00 45 00   ··>··8·· >·K···E·
00 45 1f 99 40 00 40 06  c2 d4 ac 1f 00 03 ac 1f   ·E··@·@· ········
00 04 1f 94 ac 84 1f 16  b8 50 72 71 0a 6b 80 18   ········ ·Prq·k··
01 fd 58 7d 00 00 01 01  08 0a d1 49 e0 b3 82 1a   ··X}···· ···I····
a2 a3 63 68 61 74 5f 53  54 41 52 54 54 4c 53 5f   ··chat_S TARTTLS_
41 43 4b                                           ACK
```

Thus TLS connection is established.

TLS Call flow at Trudy's Side :

tls

	Time	Source	Destination	Protocol	Length	Info
29	7.732414	172.31.0.2	172.31.0.4	TLSv1.2	204	Client Hello
31	7.735898	172.31.0.4	172.31.0.2	TLSv1.2	3112	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
33	7.742108	172.31.0.2	172.31.0.4	TLSv1.2	3423	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
35	7.744141	172.31.0.4	172.31.0.2	TLSv1.2	1236	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
36	7.744373	172.31.0.4	172.31.0.3	TLSv1.2	254	Client Hello
38	7.746916	172.31.0.3	172.31.0.4	TLSv1.2	3112	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
40	7.752383	172.31.0.4	172.31.0.3	TLSv1.2	3423	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
42	7.754279	172.31.0.3	172.31.0.4	TLSv1.2	1236	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
45	15.148178	172.31.0.2	172.31.0.4	TLSv1.2	104	Application Data
46	15.148369	172.31.0.4	172.31.0.3	TLSv1.2	104	Application Data
49	20.396136	172.31.0.3	172.31.0.4	TLSv1.2	110	Application Data
51	20.396366	172.31.0.4	172.31.0.2	TLSv1.2	110	Application Data
53	24.188124	172.31.0.2	172.31.0.4	TLSv1.2	105	Application Data
55	24.188363	172.31.0.4	172.31.0.2	TLSv1.2	97	Encrypted Alert
57	24.188617	172.31.0.2	172.31.0.4	TLSv1.2	97	Encrypted Alert
59	24.188846	172.31.0.4	172.31.0.3	TLSv1.2	97	Encrypted Alert
62	24.190837	172.31.0.3	172.31.0.4	TLSv1.2	97	Encrypted Alert

TLS Call flow at Alice's Side :

	Time	Source	Destination	Protocol	Length	Info
16	7.732580	172.31.0.2	172.31.0.4	TLSv1.2	204	Client Hello
18	7.736101	172.31.0.4	172.31.0.2	TLSv1.2	3112	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
20	7.742296	172.31.0.2	172.31.0.4	TLSv1.2	3423	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
22	7.744341	172.31.0.4	172.31.0.2	TLSv1.2	1236	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
24	15.148351	172.31.0.2	172.31.0.4	TLSv1.2	104	Application Data
26	20.396585	172.31.0.4	172.31.0.2	TLSv1.2	110	Application Data
28	24.188244	172.31.0.2	172.31.0.4	TLSv1.2	105	Application Data
30	24.188582	172.31.0.4	172.31.0.2	TLSv1.2	97	Encrypted Alert
32	24.188800	172.31.0.2	172.31.0.4	TLSv1.2	97	Encrypted Alert

TLS Call flow at Bob's Side :

	Time	Source	Destination	Protocol	Length	Info
15	7.743825	172.31.0.4	172.31.0.3	TLSv1.2	254	Client Hello
17	7.746356	172.31.0.3	172.31.0.4	TLSv1.2	3112	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
19	7.751836	172.31.0.4	172.31.0.3	TLSv1.2	3423	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
21	7.753722	172.31.0.3	172.31.0.4	TLSv1.2	1236	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
23	15.147828	172.31.0.4	172.31.0.3	TLSv1.2	104	Application Data
25	20.395502	172.31.0.3	172.31.0.4	TLSv1.2	110	Application Data
27	24.188299	172.31.0.4	172.31.0.3	TLSv1.2	97	Encrypted Alert
30	24.190267	172.31.0.3	172.31.0.4	TLSv1.2	97	Encrypted Alert

ANTI-PLAGIARISM STATEMENT

We certify that this assignment/report is our own work, based on our personal

study and/or research and that we have acknowledged all material and sources used in its preparation, whether they be books, articles, packages, datasets, reports, lecture notes, and any other kind of document, electronic or personal communication. We also certify that this assignment/report has not previously been submitted for assessment/project in any other course lab, except where specific permission has been granted from all course instructors involved, or at any other time in this course, and that we have not copied in part or whole or otherwise plagiarized the work of other students and/or persons. We pledge to uphold the principles of honesty and responsibility at CSE@IITH. In addition, We understand my responsibility to report honor violations by other students if we become aware of it.

Names: Tejas Deshmukh (cs22mtech12005), Mahanth Kumar Valluri (cs22mtech11015)
Date: 27/03/2023
Signature: <Tejas, Mahanth

References:

1. OpenSSL Cookbook: Chapter 1. OpenSSL Command Line (feistyduck.com)
2. /docs/man1.1.1/man3/index.html (openssl.org)
3. OpenSSL client and server from scratch, part 1 – Arthur O'Dwyer – Stuff mostly about C++ (quuxplusone.github.io)
4. ssl — TLS/SSL wrapper for socket objects — Python 3.9.2 documentation
5. Secure programming with the OpenSSL API – IBM Developer
6. Simple TLS Server - OpenSSLWiki

7. The /etc/hosts file (tldp.org)
8. PowerPoint Presentation (owasp.org)
9. SEED Project (seedsecuritylabs.org)