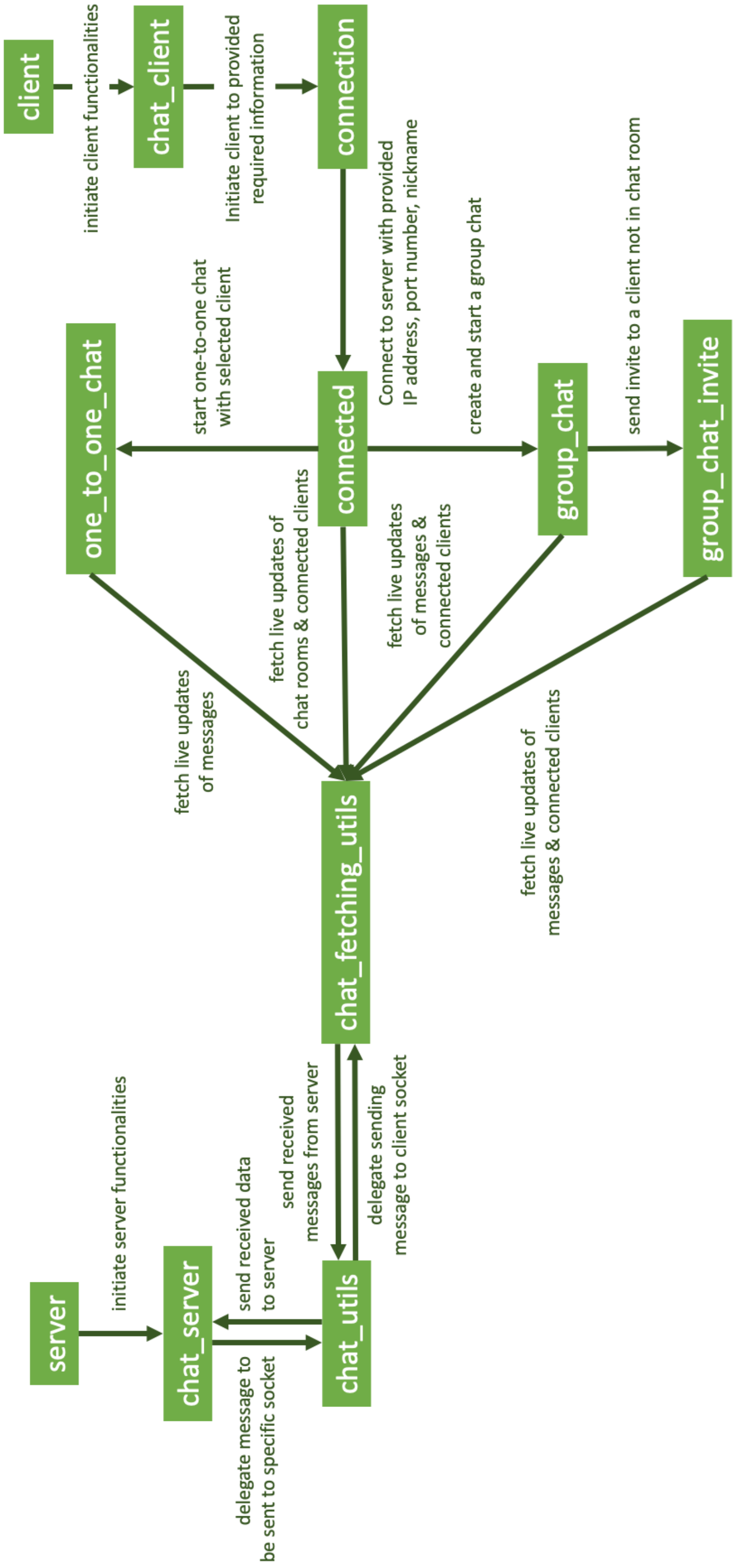
hseo227
207386547

Assignment 2 – Chatting Program

Quiz

1. **Show a system diagram of your chatting program, and explain each sub-system or module.**


client
initiate client functionalities
chat_client
Initiate client to provided required information
connection
Connect to server with provided IP address, port number, nickname
connected
start one-to-one chat with selected client
one_to_one_chat
create and start a group chat
group_chat
send invite to a client not in chat room
group_chat_invite
fetch live updates of messages
fetch live updates of chat rooms & connected clients
fetch live updates of messages & connected clients
fetch live updates of messages & connected clients
chat_fetching_utils
send received messages from server
delegate sending message to client socket
chat_utils
send received data to server
delegate message to be sent to specific socket
chat_server
initiate server functionalities
server

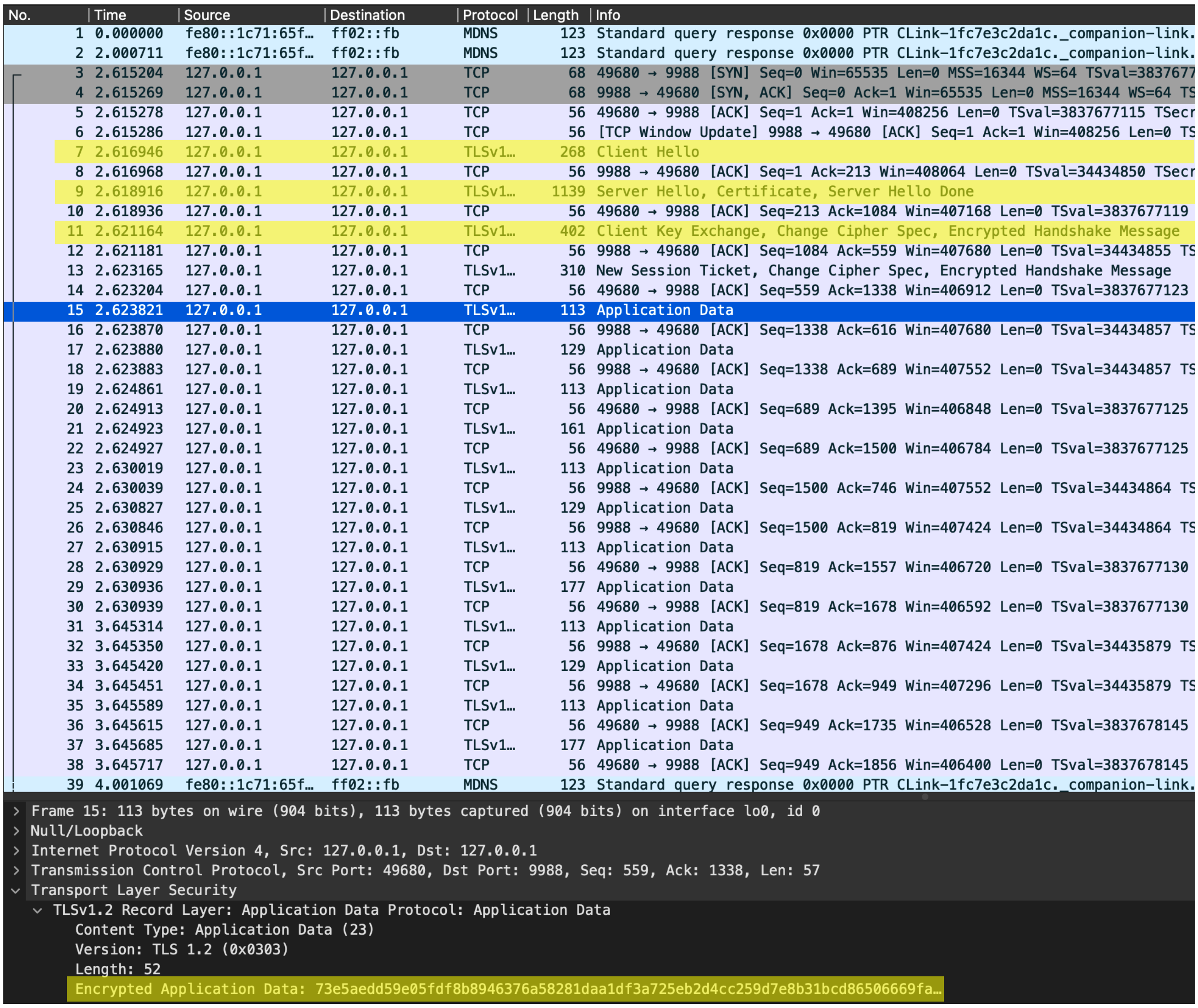
2. **Describe the encryption algorithm of your system, and show evidence using wireshark if the encryption works correctly.**

The encryption system used is AES. AES has a fixed block size of 128 bits and thus, processes data in 128-bit blocks. It has a key size of 128, 192, 256 bits, and for this chatting program, 128-bit keys are used.

cert.pem file had to be generated for the use of AES, which includes a certification and a key using *openssl*. Without *cert.pem,* the program will not function.

SSL context for the socket was created on the server side, which is a layer for TCP socket that will secure the data transferred through the sockets. Then, the certificate chain is loaded on the SSL context with *cert.pem* discussed above. This provides a certificate to be used to validate server from the client. After that, the cipher of the transferring data to AES128-SHA was set. AES128-SHA indicates ciphers to be used because AES had 128-bit key. Lastly, the socket used to receive messages for the server is wrapped with created SSL context.

SSL context for the socket used to receive message for the client, for the client side of the system. Then, the socket is wrapped with SSL context created to used SSL for any communication using this port.



```
No.    Time      Source            Destination   Protocol  Length  Info
     1 0.000000  fe80::1c71:65f…   ff02::fb      MDNS         123  Standard query response 0x0000 PTR CLink-1fc7e3c2da1c._companion-link.
     2 2.000711  fe80::1c71:65f…   ff02::fb      MDNS         123  Standard query response 0x0000 PTR CLink-1fc7e3c2da1c._companion-link.
     3 2.615204  127.0.0.1         127.0.0.1     TCP           68  49680 → 9988 [SYN] Seq=0 Win=65535 Len=0 MSS=16344 WS=64 TSval=3837677
     4 2.615269  127.0.0.1         127.0.0.1     TCP           68  9988 → 49680 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=16344 WS=64 TS
     5 2.615278  127.0.0.1         127.0.0.1     TCP           56  49680 → 9988 [ACK] Seq=1 Ack=1 Win=408256 Len=0 TSval=3837677115 TSecr
     6 2.615286  127.0.0.1         127.0.0.1     TCP           56  [TCP Window Update] 9988 → 49680 [ACK] Seq=1 Ack=1 Win=408256 Len=0 TS
     7 2.616946  127.0.0.1         127.0.0.1     TLSv1…       268  Client Hello
     8 2.616968  127.0.0.1         127.0.0.1     TCP           56  9988 → 49680 [ACK] Seq=1 Ack=213 Win=408064 Len=0 TSval=34434850 TSecr
     9 2.618916  127.0.0.1         127.0.0.1     TLSv1…      1139  Server Hello, Certificate, Server Hello Done
    10 2.618936  127.0.0.1         127.0.0.1     TCP           56  49680 → 9988 [ACK] Seq=213 Ack=1084 Win=407168 Len=0 TSval=3837677119
    11 2.621164  127.0.0.1         127.0.0.1     TLSv1…       402  Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
    12 2.621181  127.0.0.1         127.0.0.1     TCP           56  9988 → 49680 [ACK] Seq=1084 Ack=559 Win=407680 Len=0 TSval=34434855 TS
    13 2.623165  127.0.0.1         127.0.0.1     TLSv1…       310  New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
    14 2.623204  127.0.0.1         127.0.0.1     TCP           56  49680 → 9988 [ACK] Seq=559 Ack=1338 Win=406912 Len=0 TSval=3837677123
    15 2.623821  127.0.0.1         127.0.0.1     TLSv1…       113  Application Data
    16 2.623870  127.0.0.1         127.0.0.1     TCP           56  9988 → 49680 [ACK] Seq=1338 Ack=616 Win=407680 Len=0 TSval=34434857 TS
    17 2.623880  127.0.0.1         127.0.0.1     TLSv1…       129  Application Data
    18 2.623883  127.0.0.1         127.0.0.1     TCP           56  9988 → 49680 [ACK] Seq=1338 Ack=689 Win=407552 Len=0 TSval=34434857 TS
    19 2.624861  127.0.0.1         127.0.0.1     TLSv1…       113  Application Data
    20 2.624913  127.0.0.1         127.0.0.1     TCP           56  49680 → 9988 [ACK] Seq=689 Ack=1395 Win=406848 Len=0 TSval=3837677125
    21 2.624923  127.0.0.1         127.0.0.1     TLSv1…       161  Application Data
    22 2.624927  127.0.0.1         127.0.0.1     TCP           56  49680 → 9988 [ACK] Seq=689 Ack=1500 Win=406784 Len=0 TSval=3837677125
    23 2.630019  127.0.0.1         127.0.0.1     TLSv1…       113  Application Data
    24 2.630039  127.0.0.1         127.0.0.1     TCP           56  9988 → 49680 [ACK] Seq=1500 Ack=746 Win=407552 Len=0 TSval=34434864 TS
    25 2.630827  127.0.0.1         127.0.0.1     TLSv1…       129  Application Data
    26 2.630846  127.0.0.1         127.0.0.1     TCP           56  9988 → 49680 [ACK] Seq=1500 Ack=819 Win=407424 Len=0 TSval=34434864 TS
    27 2.630915  127.0.0.1         127.0.0.1     TLSv1…       113  Application Data
    28 2.630929  127.0.0.1         127.0.0.1     TCP           56  49680 → 9988 [ACK] Seq=819 Ack=1557 Win=406720 Len=0 TSval=3837677130
    29 2.630936  127.0.0.1         127.0.0.1     TLSv1…       177  Application Data
    30 2.630939  127.0.0.1         127.0.0.1     TCP           56  49680 → 9988 [ACK] Seq=819 Ack=1678 Win=406592 Len=0 TSval=3837677130
    31 3.645314  127.0.0.1         127.0.0.1     TLSv1…       113  Application Data
    32 3.645350  127.0.0.1         127.0.0.1     TCP           56  9988 → 49680 [ACK] Seq=1678 Ack=876 Win=407424 Len=0 TSval=34435879 TS
    33 3.645420  127.0.0.1         127.0.0.1     TLSv1…       129  Application Data
    34 3.645451  127.0.0.1         127.0.0.1     TCP           56  9988 → 49680 [ACK] Seq=1678 Ack=949 Win=407296 Len=0 TSval=34435879 TS
    35 3.645589  127.0.0.1         127.0.0.1     TLSv1…       113  Application Data
    36 3.645615  127.0.0.1         127.0.0.1     TCP           56  49680 → 9988 [ACK] Seq=949 Ack=1735 Win=406528 Len=0 TSval=3837678145
    37 3.645685  127.0.0.1         127.0.0.1     TLSv1…       177  Application Data
    38 3.645717  127.0.0.1         127.0.0.1     TCP           56  49680 → 9988 [ACK] Seq=949 Ack=1856 Win=406400 Len=0 TSval=3837678145
    39 4.001069  fe80::1c71:65f…   ff02::fb      MDNS         123  Standard query response 0x0000 PTR CLink-1fc7e3c2da1c._companion-link.
```

```
> Frame 15: 113 bytes on wire (904 bits), 113 bytes captured (904 bits) on interface lo0, id 0
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 49680, Dst Port: 9988, Seq: 559, Ack: 1338, Len: 57
⌄ Transport Layer Security
  ⌄ TLSv1.2 Record Layer: Application Data Protocol: Application Data
      Content Type: Application Data (23)
      Version: TLS 1.2 (0x0303)
      Length: 52
      Encrypted Application Data: 73e5aedd59e05fdf8b8946376a58281daa1df3a725eb2d4cc259d7e8b31bcd86506669fa…
```

We can clearly see the encryption in the *Wireshark* capture above, through the handshake messages of '*Client Hello*' in packet frame 7 and '*Server Hello*' in packet frame 9 followed by the '*Application Data*'. We can also see the *encrypted handshake* happening in packet frame 13.

Looking into the *Transport Layer Security* of the '*Application Data*', we are able to see the encrypted data being sent.