**Assignment- 2**

**Problem Statement:**

The objective of this laboratory exercise is to look at the details of the Transmission Control Protocol (TCP). TCP is a transport layer protocol. It is used by many application protocols like HTTP, FTP, SSH etc., where guaranteed and reliable delivery of messages is required. To do this exercise you need to install the Wireshark tool. This tool would be used to capture and examine a packet trace. Wireshark can be downloaded from [www.wireshark.org](http://www.wireshark.org).

**Step1: Capture a Trace**

1. Launch Wireshark
2. From Capture→Options select Loopback interface
3. Start a capture with a filter of “ip.addr==127.0.0.1 and tcp.port==xxxx”, where xxxx is the port number used by the TCP server.
4. Run the TCP server program on a terminal.
5. Run two instances of the TCP client program on two separate terminals and send some dummy data to the sever.
6. Stop Wireshark capture

**Step2: TCP Connection Establishment**

To observe the three-way handshake in action, look for a TCP segment with SYN flag set. A ”SYN” segment is the start of the three-way handshake and is sent by the TCP client to the TCP server. The server then replies with a TCP segment with SYN and ACK flag set. And finally the client sends an ”ACK” to the server. For all the above three segments record the values of the sequence number and acknowledgment fields. Draw a time sequence diagram of the three-way handshake for TCP connection establishment in your trace. Do it for all the client connections.

**Step3: TCP Data Transfer**

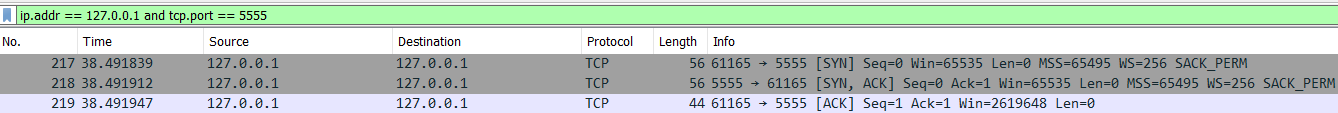
For all data segments sent by the client, record the value of the sequence number and acknowledge number fields. Also, record the same for the corresponding acknowledgements sent by the server. Draw a time sequence diagram of the data transfer in your trace. Do it for all the client connections.

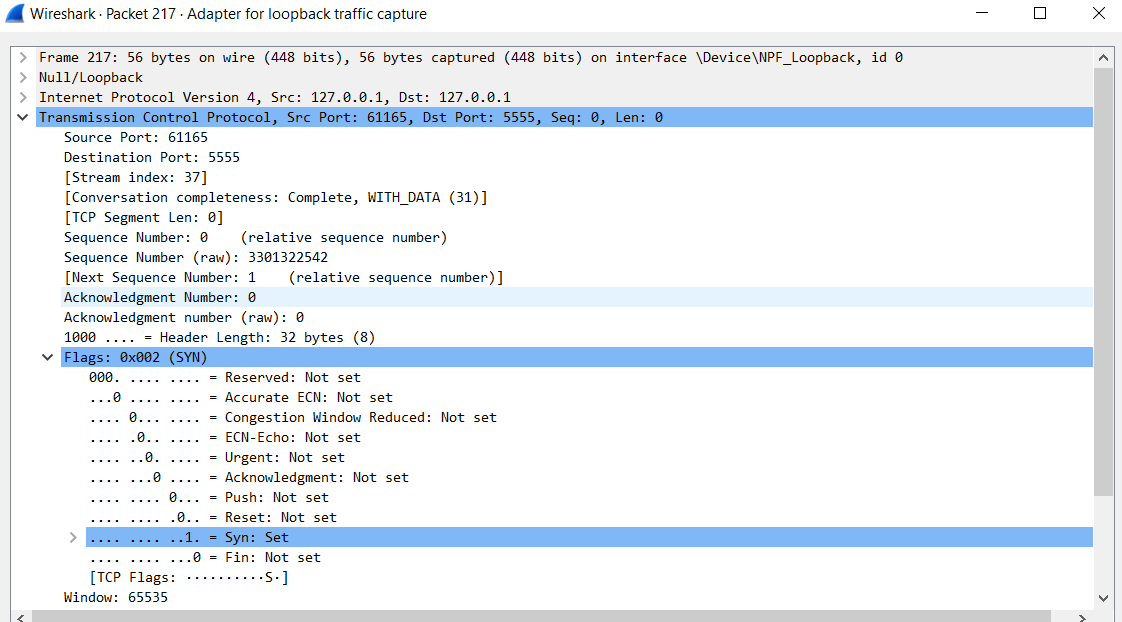
**Step4: TCP Connection Termination**

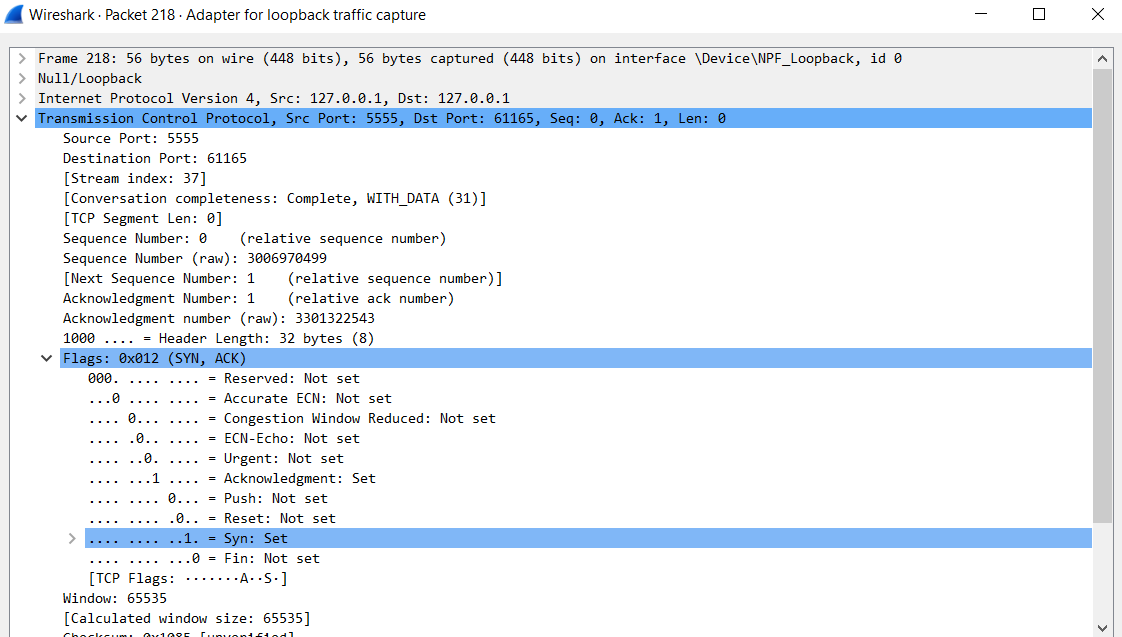
Once the data transfer is over, the client initiates the connection termination by sending TCP segment with FIN flag set, to the server. Server acknowledges it and sends it’s own intention to terminate the connection by sending a TCP segment with FIN and ACK flags set. The client finally sends an ACK segment to the server. For all the above three segments record the values of the sequence number and acknowledgment fields. Draw a time sequence diagram of the three-way handshake for TCP connection termination in your trace. Do it for all the client connections.

**TCP connection establishment:**

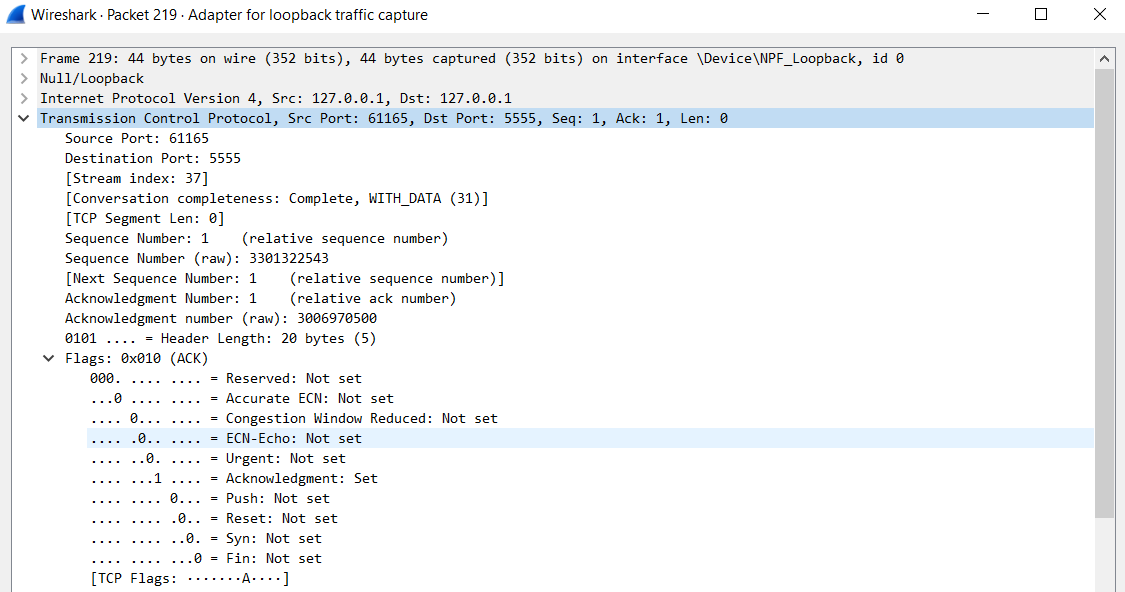
A snapshot of the 3-way handshake for TCP connection establishment.



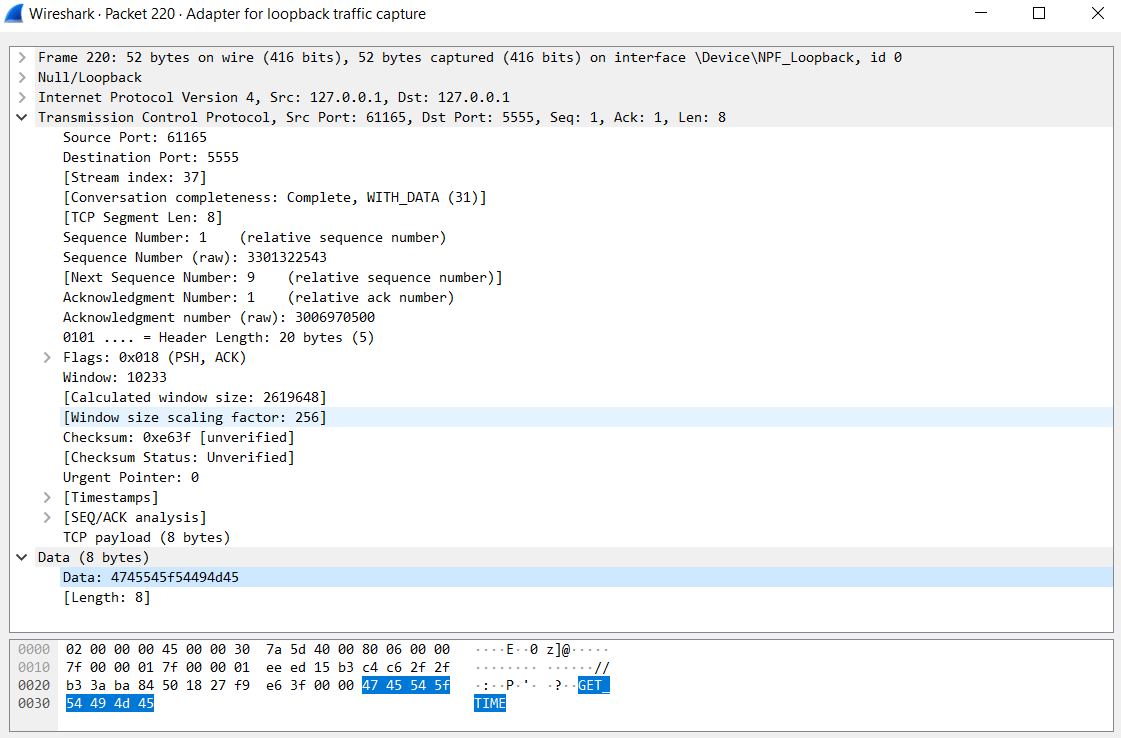
**Step-1:** Client sends SYN (Synchronize) message to the server. The message includes SYN flag set to 1 and it contains a unique sequence number which is any 32-bit number and acknowledge number which is set to 0.

**Step-2:** Now the server responds with SYN and ACK (Acknowledge) message to the client. Here the SYN and ACK flags are set to 1. The ACK number is one higher than the SYN sequence number received by the client. But the sequence number will be different.

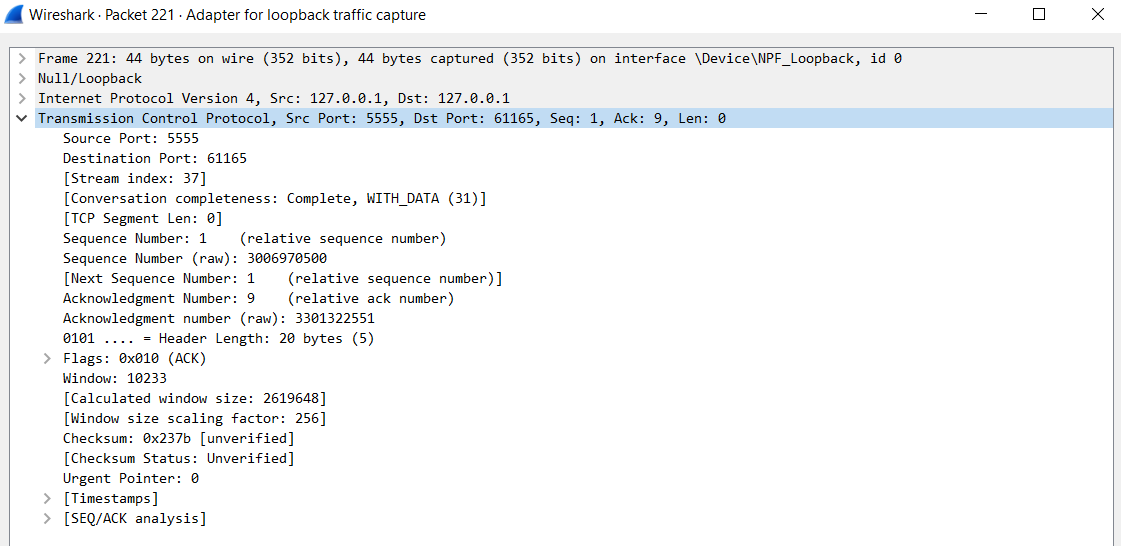
**Step-3:** After client has received the SYN – ACK message from server it sends and Acknowledge message to the server. The ACK flag is set to 1 and the ACK number will be one higher than the previously received sequence number from server.



**TCP Data Transfer:**

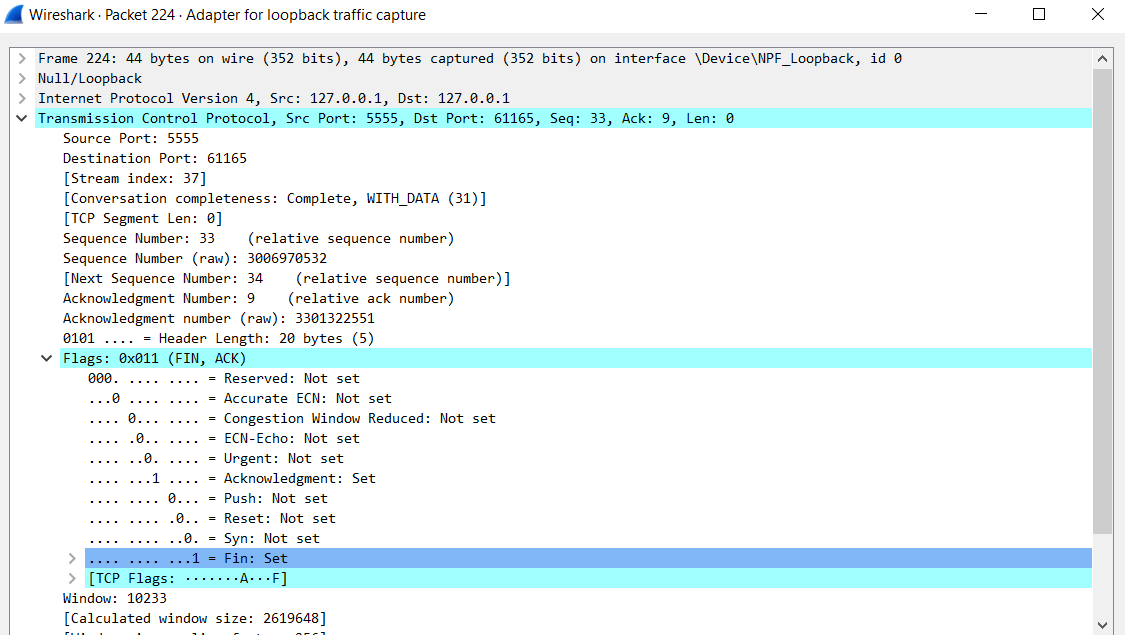
After successful connection data can be transferred in between the server and the client. Whoever wants to send data will send a message with flags PSH and ACK set to 1 along with the actual data.

When the other computer receives the data, it will send another ACK message to the sender. In this case the ACK number will be sequence number of previous message + number of bytes sent.

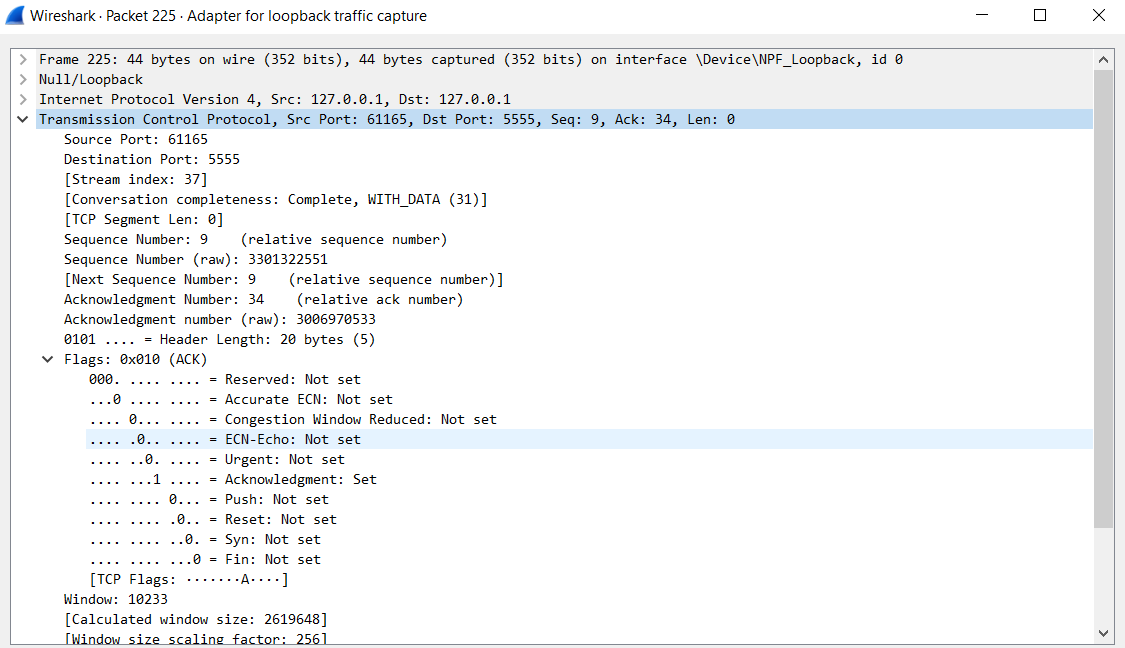


**TCP Connection Termination:**

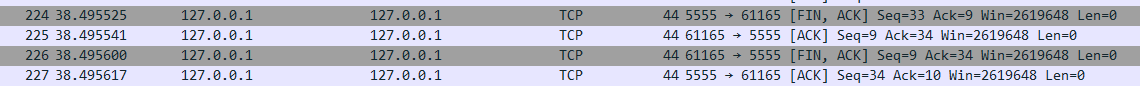
For connection termination one computer say the server will send a message with FIN and ACK flag set to 1. The client upon receiving the message will send an ACK message to the previous sender as acknowledge for closing the connection.

Similarly, the client will now send a message with FIN and ACK set to 1 to server and server will send the acknowledge as FIN, ACK back to the client.

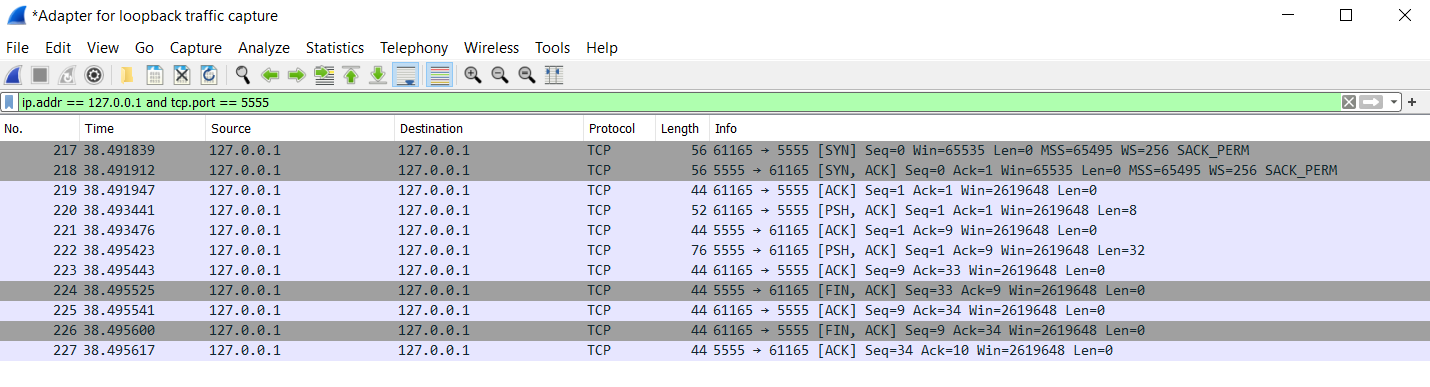
For the ACK message the ACK number will be one higher than the sequence number of the previous (FIN, ACK) message.



This is called a 4-way handshake, which is required for closing a TCP connection. This is actually a set of 2-way handshake.

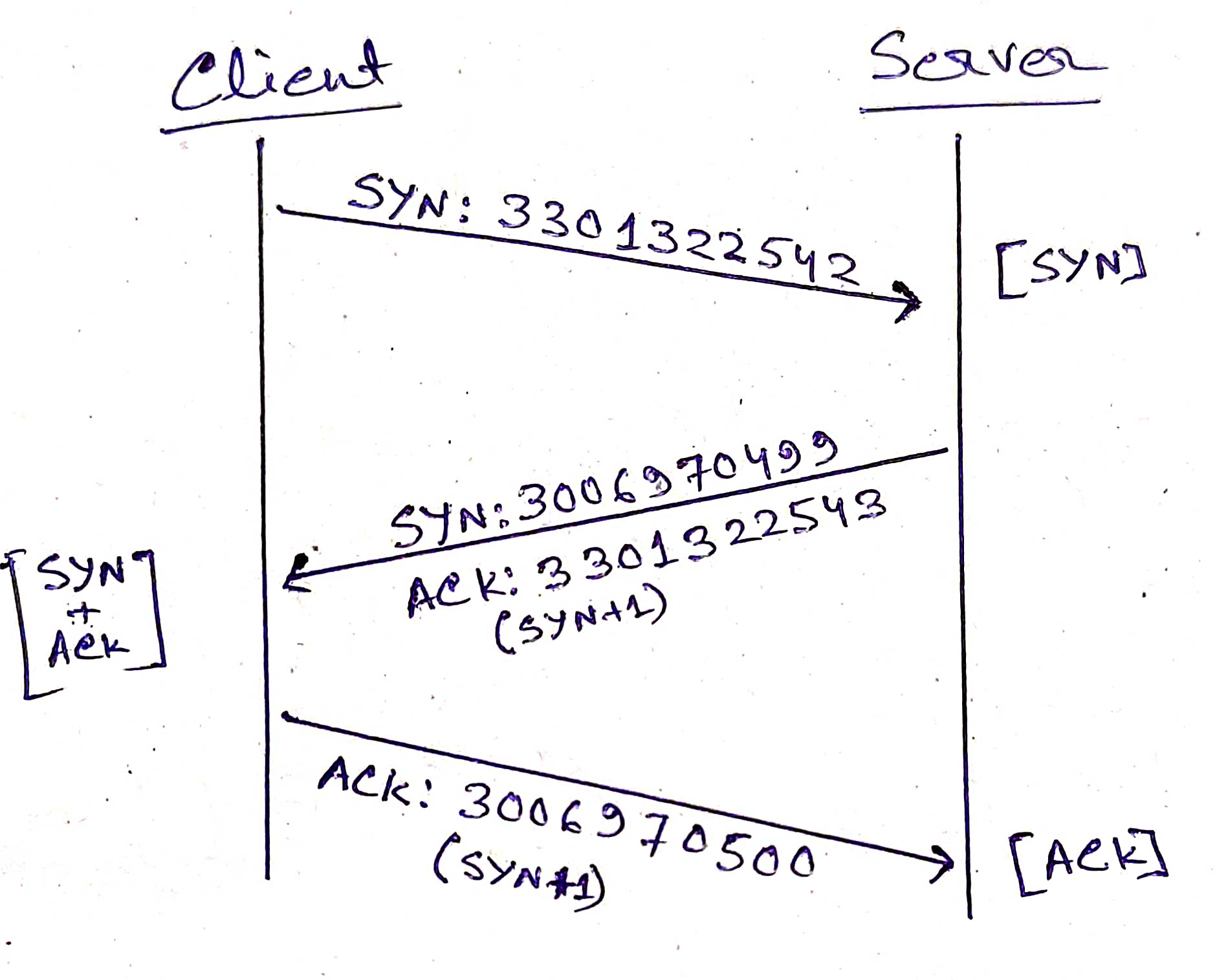


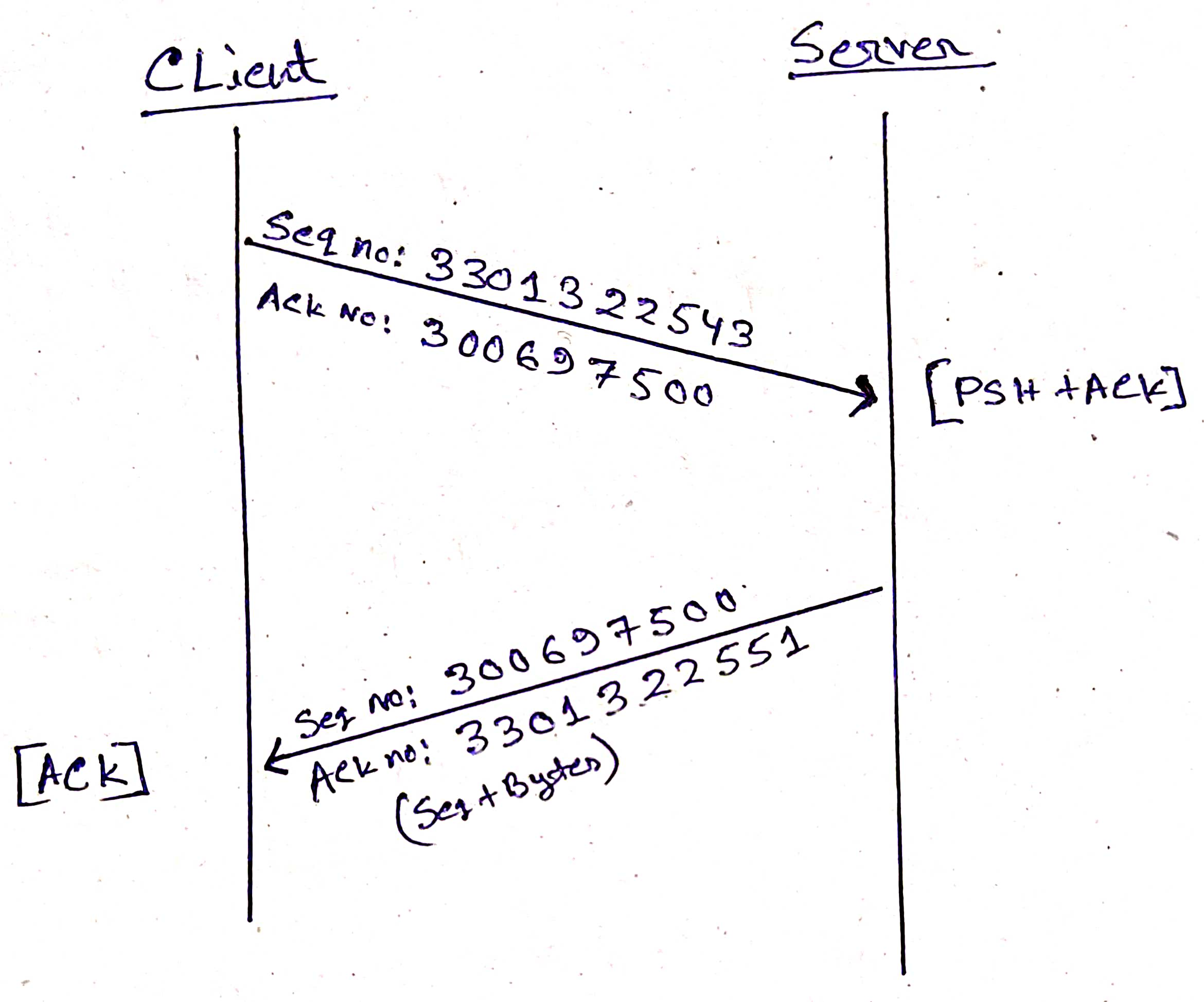
**Snapshot of everything altogether:**



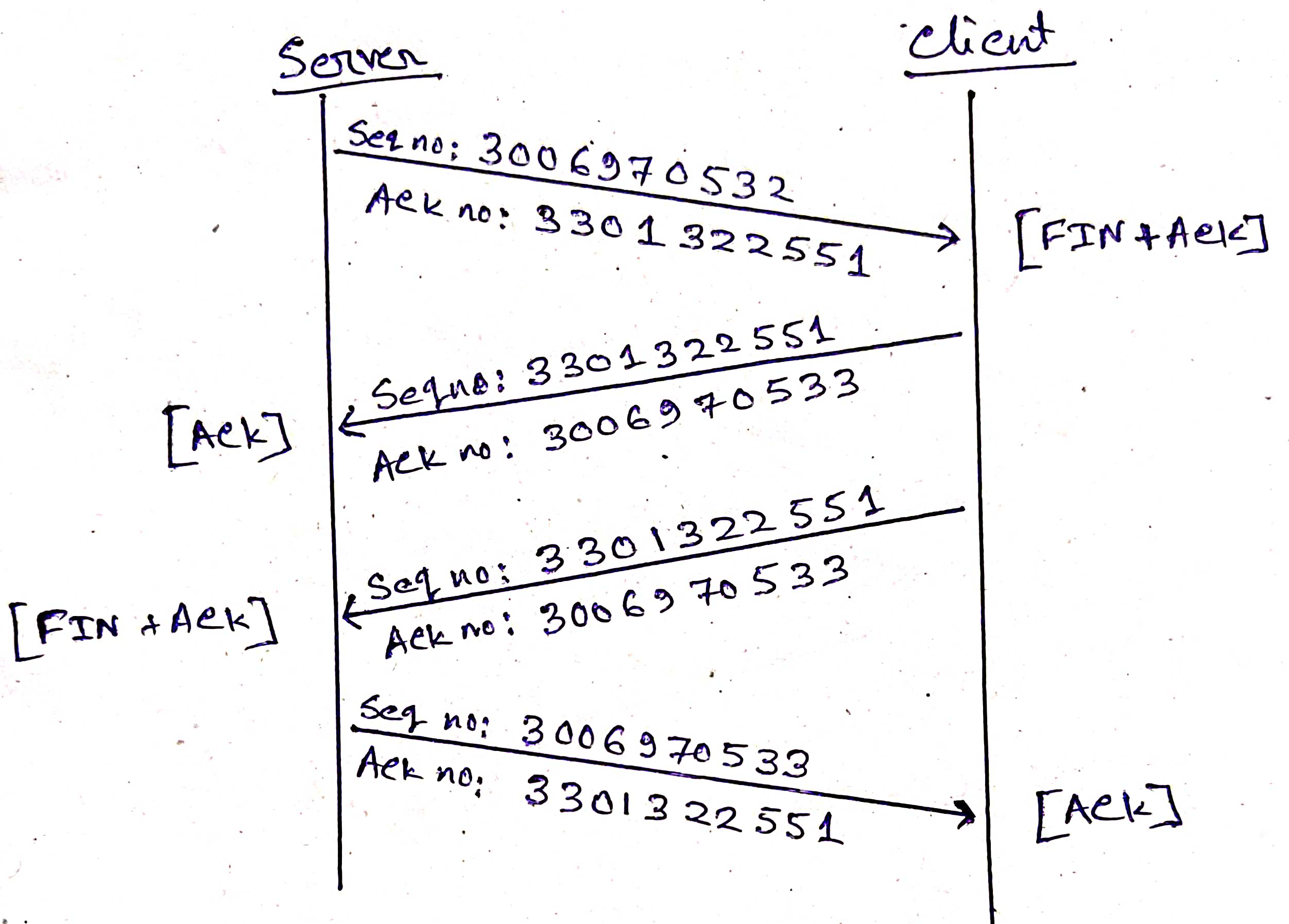
**Timing sequence diagram:**

**TCP connection:**



**Data transfer:**

**Closing connection:**

****