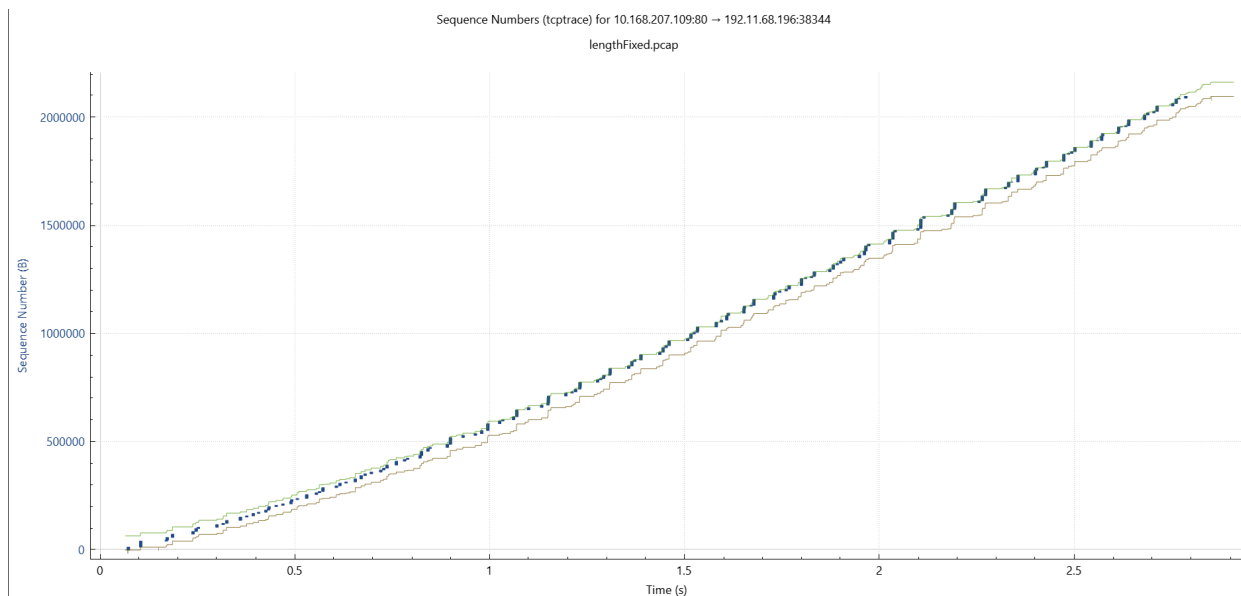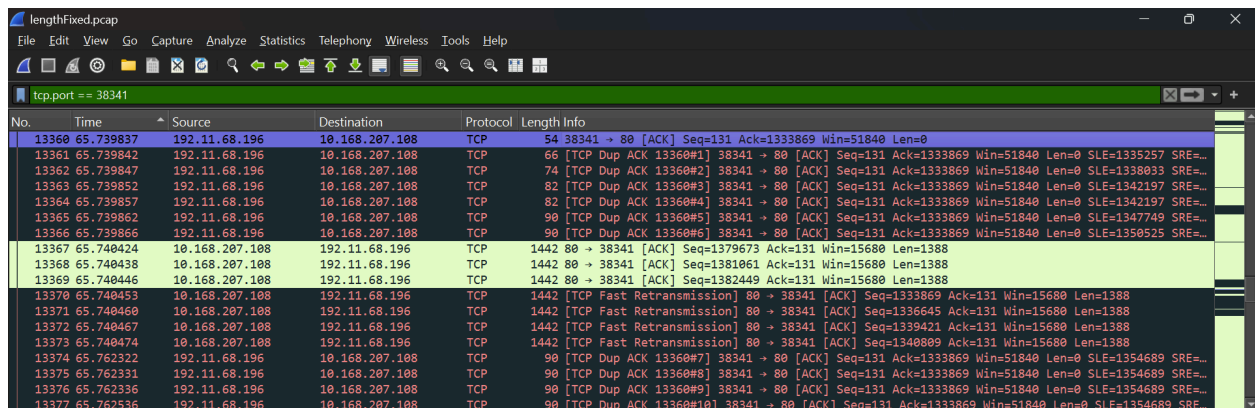Figure below is the TCP Sequence Graph for the connection between Server #2 (10.168.207.109:80) and Client #2 (192.11.68.196:38344). This connection is an example of a stable TCP connection. The discrete blue lines represent the individual TCP segments being sent - the line length scales with the size of the segment. The orange line at the bottom shows the progress of the ACK number from the client receiver, which increments with every packet received by the client. From the analysis, it can be seen that Client #2 advertises a constant 65535 Byte receive window to Server #2 (Scaling Factor of 1), which fits with the linear growth of the green receive-window line (advertised RWIND + Receiver ACK Number). Since the blue packet lines remain very close to the RWIND slope throughout, a pattern can be seen of Server #2 transmitting to the full capability of Client #2's RWIND, then waiting for an ACK from the client to continue transmission. The linear behavior of the graph demonstrates the consistent throughput from server to client, though its throughput is also noticeably lower than Server #1's due to the stricter window scale (719KBps vs Server #1's 961KBps).
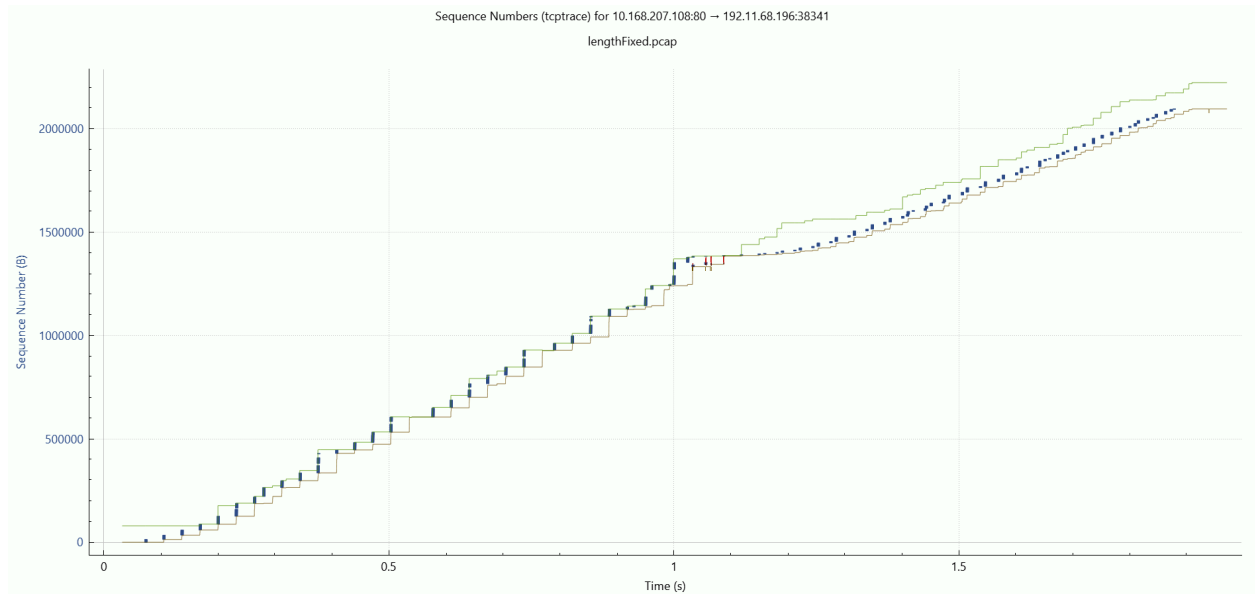

Sequence graph for Server #2 → Client #2

Below is the TCP sequence graph for the connection between Server #1 (10.168.207.108:80) and Client #1 (192.11.68.196:38341). In this case, the client advertises a window scaling factor of 128 to the server using its initial SYN packet as seen in the session analysis. Compared to the consistent throughput as seen with Client/Server #2, this connection shows a faster throughput in the beginning, given the sharper slope. This can be attributed to the larger window size provided to the server. However, this connection also shows greater instability as the server attempts to keep up with the client's window size. Numerous fast retransmissions can be seen after 1 second into the connection life due to duplicate ACKs sent from the client. The server then transitions to a slow-start state, creating a large gap between the receive window and the ACK sequence number.

Both connections demonstrate the impact window scaling can have on the efficiency of a TCP connection. Using default window scaling can keep transmissions stable and lines uncongested, but at the cost of a smaller throughput and underutilization of the bandwidth available. Increased window scaling creates potential for greater throughput in a connection, but is best reserved for when both ends of the connection can take full advantage of the available buffer.
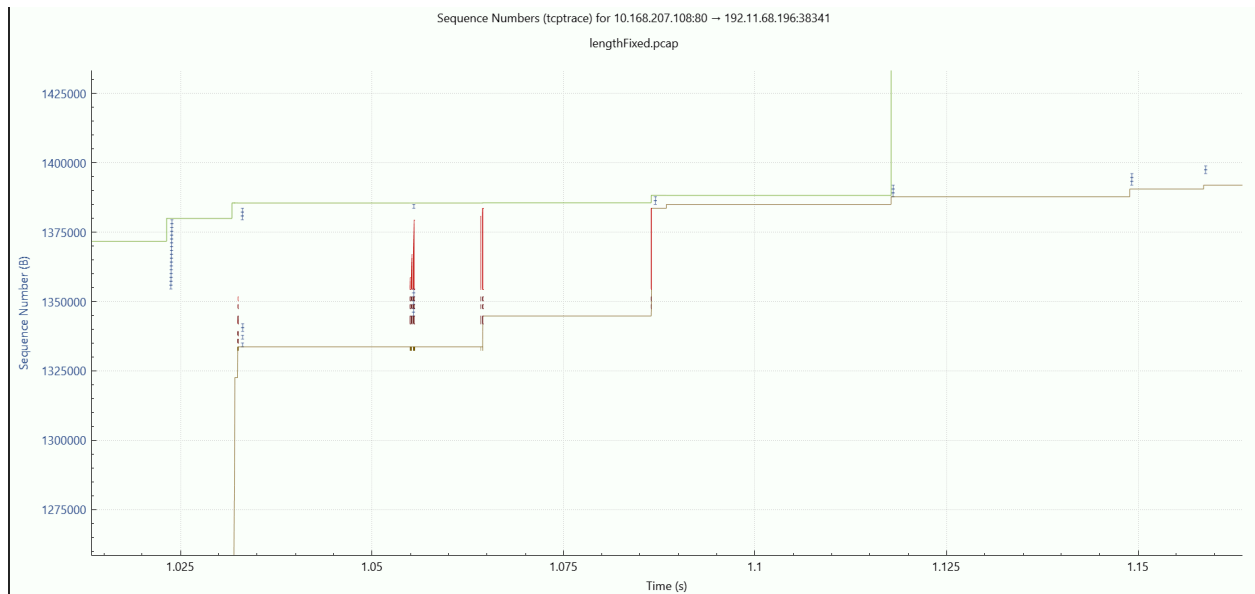

Fast retransmissions from Server #1 after detecting duplicate ACKs from Client #1

Sequence Graph of Server #1 → Client #1

Duplicate ACKs can be seen by the hitches under the yellow ACK line. Retransmissions can be seen by the long red lines.