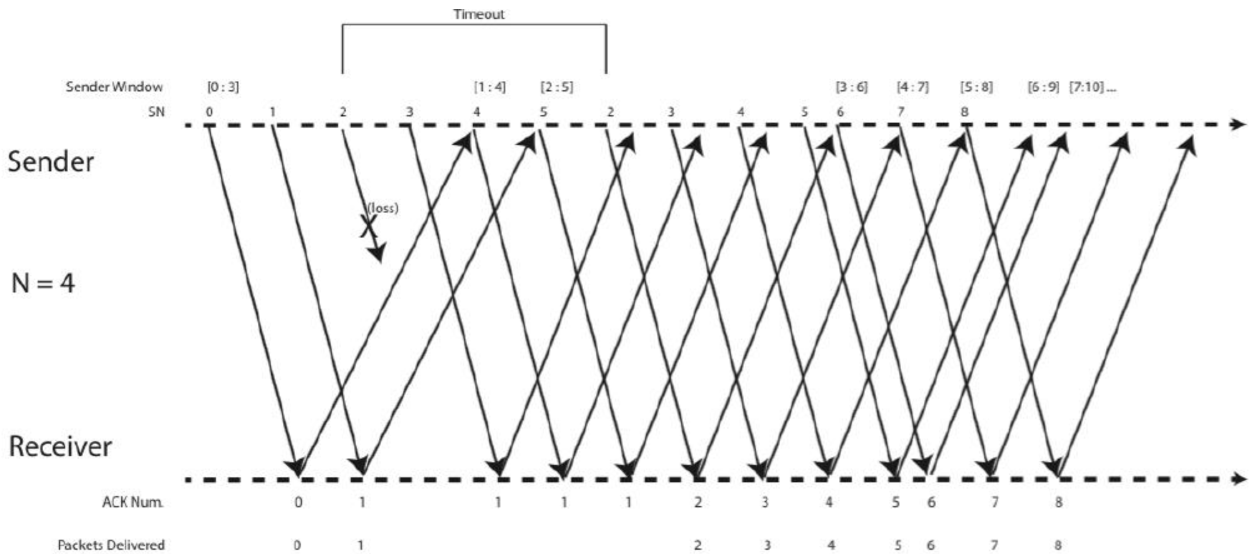
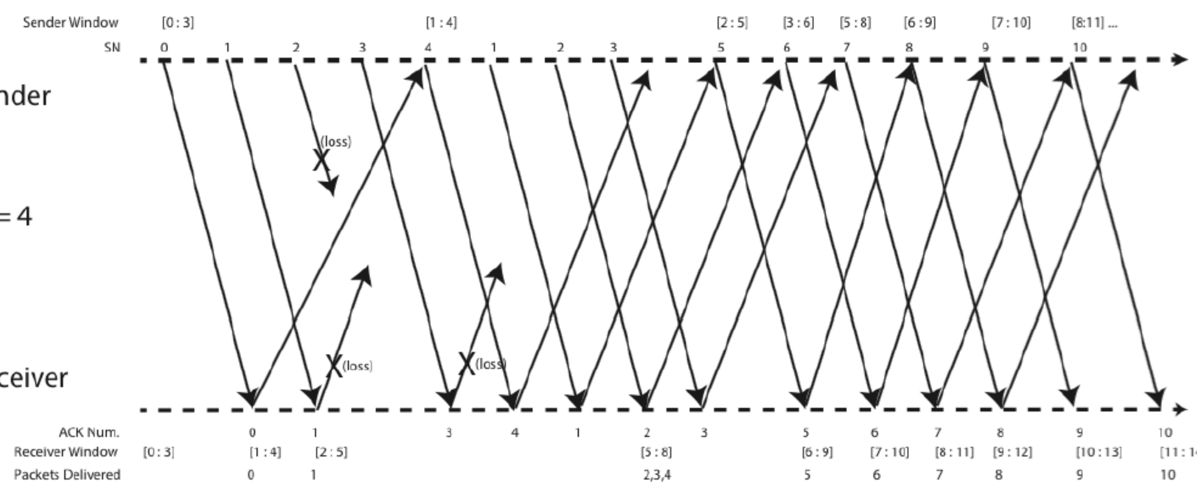
Assignment 3

Problem 1 Solution

a)



b)



Problem 2 Solution

- **a)** Slow Start: [1-6], [7-11], [23-26]
- **b)** Congestion Avoidance: [11-15], [16-22], [26,32]
- **c)** Fast retransmit/recovery: [15-16]
- **d)** Where Fast Recovery could have but did not happen: [6-7], [22-23], either because of packet losses or a small window size, the algorithm falls into slow start.

Problem 3 Solution

a) Let W denote the max window size measured in segments.
Then, $W \times MSS/RTT$=1Mbps, as packets will be dropped if the maximum sending rate exceeds link capacity. Thus, we have $W \times 1200 \times 8/0.16 = 15 \times 10^6$, then W is about 250.

b) As congestion window size varies from W / 2 to W, then the average window size is 0.75W=187.5 segments. Average throughput is $187.5 \times 1200 \times 8/0.16 = 11.25$ Mbps.

c) $(250/2) \times 0.16 = 20$ seconds, as the number of RTTs (that this TCP connections needs in order to increase its window size from W/2 to W) is given by W/2. Recall the window size increases by one in each RTT.

Problem 4 Solution

a) The sequence number is 32 bits, so it will wrap around when we sent 4GB = 4×8Gb = 32Gb of data. The link is 10 Gbps, so it take 32Gb/(10Gbs) = 3.2s, i.e. it will take 3.2 seconds for the sequence number to wrap around.

b) The timestamp will increment 100,000 times during each 3.2 seconds, which means that it will increment by one each 3.2×10^{-5}s = 32μs, so we need $2^{32} \times 3.2 \times 10^{-5}$s. Considering that $2^{32} \cong 4 \times 10^9$, then we need $4 \times 10^9 \times 3.2 \times 10^{-5} = 12.8 \times 10^4 =$ 128000s, which is about 35 hours.

Problem 5 Solution

a) If the remote port is different, then it is for a new connection. Otherwise, if the remote port of both packets are the same, then it is a retransmission if both packets have the same ISN (initial sequence number), because ISN are clock-generated, and a new connection request will have a different ISN.

b)

I) 4RTT+ 6 S/R
Approach:
1 RTT for the TCP connection
1 RTT + 1 S/R for receiving the first packet (cwnd=1)
1 RTT + 1 S/R for receiving the next two packets (cwnd=2)*
1 RTT + 4 S/R for receiving the next four packets (cwnd=4)
* the third packet is received during the next RTT

II) 4RTT + 6 S/R
Approach:
Similar to part I)

III) 3RTT + 7 S/R
Approach:
1 RTT for the TCP connection
1 RTT + 1 S/R for receiving the first packet (cwnd=1)
1 RTT + 2 S/R for receiving the next two packets (cwnd=2)
0 RTT + 4 S/R for receiving the next fourth packets (cwnd=4)
Note: As RTT is shorter than S/R after cwnd=2, the ACKs arrive at the server faster than packets being sent