Problem 1

Host A and B are communicating over a TCP connection, and Host B has already received from A all bytes up through byte 326. Suppose Host A then sends two segments to Host B back-to-back. The first and second segments contain 80 and 40 bytes of data, respectively. In the first segment, the sequence number is 327, the source port number is 40200, and the destination port number is 80. Host B sends an acknowledgment whenever it receives a segment from Host A. Fill in the blanks for questions (a) – (c) directly; work out the diagram in the box for question (d).

(a) In the second segment sent from Host A to B, the sequence number is _____, source port number is _____, and destination port number is _____.

(b) If the first segment arrives before the second segment, in the acknowledgment of the first arriving segment, the ACK number is _____, the source port number is _____, and the destination port number is _____.

(c) If the second segment arrives before the first segment, in the acknowledgment of the first arriving segment, the ACK number is _____.

(d) Suppose the two segments sent by A arrive in order at B. The first acknowledgment is lost and the second acknowledgment arrives after A's timeout intervals right after the time out interval of the first segment. Draw a timing diagram in the box below, showing these segments and acknowledgments until A receives all the acknowledgments of re-transmitted packets. Assume no additional packet loss. For each segment in your diagram, provide the sequence number and the number of bytes of data; for each acknowledgment that you add, provide the ACK number.

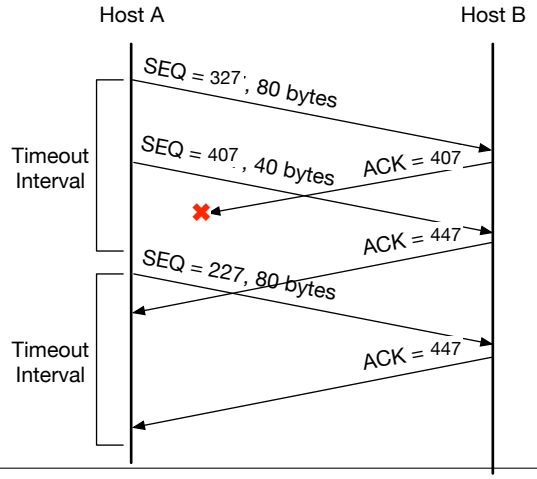
(a) 407, 400002, 80. In the second segment from Host A to B, the sequence number is 327 + 80 = 407, source port number is 40002, and destination port number is 80.

(b) 407, 80, 400002. If the first segment arrives before the second, in the acknowledgment of the first arriving segment, the ACK number is the next expected byte, which is 407, the source port number is 80 and the destination port number is 40002.

(c) 327. If the second segment arrives before the first segment, in the acknowledgment of the first arriving segment, the ACK number is still 327, indicating that it is still waiting for bytes 327 and onwards.

(d) The question statement is wrong. It should be "second acknowledgment arrives after A's timeout intervals right after the time out interval of the first segment" (now it has been fixed). Therefore, we will not deduct points on this.

The figure for the corrected questions is shown as follows.



Problem 2

One of the three functions of a sliding window scheme is the orderly delivery of packets which arrive out of sequence. In Go-back-N, the receiver drops packets which arrives out of order. Assume the receiver sends an ACK for every packet it receives.

(a) In Go-back-N, what is the required buffer size (receiver's window size, RWS) at the receiver if sender's window size (SWS) = 23? What the answer will be in Selective Repeat?

(b) In sliding window with SWS = RWS = 5, the minimum required `SeqNumSize` (the number of available sequence numbers) is 10. Calculate the minimum required `SeqNumSize` for

(i) a sliding window scheme with SWS = 6 and RWS = 3

(ii) a Go-back-N scheme with SWS = 6

(a) For Go-Back-N, RWS = 1. For selective repeat, RWS = 23 (or $\leq$ 23). In general the RWS should be $\leq$ SWS to avoid waste of memory.
(b) the minimum required sequence size for a sliding window is (SWS + RWS). If SWS=N and RWS=M, consider the following worst-case scenario: The sender sends N packets. All are received in order at the receiver, but all of its ACKs are corrupted. The receiver is expecting packets from (N+1) to (N+M). After time-out, the sender repeats the same packets, but the receiver is expecting new ones. For RWS=M, the confusion is avoided if the (minimum) total number of SeqNum is (N+M).
Therefore: i) 9 ii) 7

Problem 3

Suppose that three measured SampleRTT values are 106 ms, 120 ms, and 150 ms. Compute the EstimatedRTT after each of these SampleRTT values is obtained, assuming that the value of EstimatedRTT was 100 ms just before the first of these three samples were obtained. Compute also the DevRTT after each sample is obtained, assuming the value of DevRTT was 5 ms just before the first of these three samples was obtained. Last, compute the TCP TimeoutInterval after each of these samples is obtained. Round your calculation results to two decimals (e.g., 123.45).

Note: students may have different precision in their results and this is allowed.

$$\text{EstimatedRTT} = \alpha\text{SampleRTT} + (1-\alpha)\text{EstimatedRTT}$$

$$\text{DevRTT} = \beta\ |\text{SampleRTT} - \text{EstimatedRTT}| + (1-\beta)\text{DevRTT}$$

$$\text{TimeoutInterval} = \text{EstimatedRTT} + 4\text{DevRTT}$$

(a) After obtaining first sampleRTT = 106 ms:

$$\text{EstimatedRTT} = 0.125 \times 106 + 0.875 \times 100 = 100.75 \text{ ms},$$

$$\text{DevRTT} = 0.25 \times |106 - 100.75| + 0.75 \times 5 = 5.06 \text{ ms},$$

$$\text{TimeoutInterval} = 100.75 + 4 \times 5.06 = 120.99 \text{ ms};$$

(b) After obtaining second sampleRTT = 120 ms:

$$\text{EstimatedRTT} = 0.125 \times 120 + 0.875 \times 100.75 = 103.15 \text{ ms},$$

$$\text{DevRTT} = 0.25 \times |120 - 103.15| + 0.75 \times 5.06 = 8 \text{ ms},$$

$$\text{TimeoutInterval} = 103.15 + 4 \times 8 = 135.15 \text{ ms};$$

(c) After obtaining third sampleRTT = 150 ms:

$$\text{EstimatedRTT} = 0.125 \times 150 + 0.875 \times 103.15 = 109.01 \text{ ms},$$

$$\text{DevRTT} = 0.25 \times |150 - 107.76| + 0.75 \times 8 = 16.56 \text{ ms},$$

$$\text{TimeoutInterval} = 109.01 + 4 \times 16.56 = 175.25 \text{ ms}.$$

Problem 4

Compare Go-Back-N, Selective Repeat, and TCP (no delayed ACK). Assume that timeout values for all three protocols are sufficiently long, such that 10 consecutive data segments and their corresponding ACKs can be received (if not lost in the channel) by the receiving host (Host B) and the sending host (Host A), respectively. Suppose Host A sends 10 data segments to Host B, and the 6th segment (sent from A) is lost. In the end, all 10 data segments have been correctly received by Host B.

(a) How many segments has Host A sent in total and how many ACKs has Host B sent in total? What are their sequence numbers? Answer this question for all three protocols.

(b) If the timeout values for all three protocols are very long (e.g., longer than several RTTs), then which protocol successfully delivers all 10 data segments in shortest time interval?

(a) Go-Back-N:

- A sends 15 segments in total. A first sends 10 segments 1 - 10 and later re-sends 5 segments 6-10.
- B sends 14 ACKs. B first sends 9 ACKs 1,2,3,4,5,5,5,5,5, and later 5 ACKs 6, 7, 8, 9, 10.

Selective Repeat:

- A sends 11 segments in total. A initially sends 10 segments 1-10 and later re-sends one segment 6.
- B sends 10 ACKs. B first sends 9 ACKs 1-5, 7-10, and then sends one ACK 6.

TCP:

- A sends 11 segments in total. A initially sends segments 1-10 and later re-sends one segment 6.
- B sends 10 ACKs. B first sends 9 ACKS with ACK number 2,3,4,5,6,6,6,6,6 and then sends one ACK 11. Note that TCP always send an ACK with expected sequence number.

(b) TCP. This is because TCP uses fast retransmit without waiting until time out.

Problem 5

As we have discussed in the class, a timer is a useful component in various protocol designs: because a communicating end cannot see what is going on either inside the network or at the other end, when needed it sets up an "alarm", and takes some action when the alarm goes off.

(a) Does HTTP (not considering the underlying TCP) use any timers? If so, please briefly describe how each is used. If not, please explain why it does not need one.

(b) Does DNS use any timers? If so, please briefly describe how each is used. If not, please explain why it does not need one.

(c) Does TCP use any timer? If so, please briefly describe how each is used. If not, please explain why it does not need one.

(d) Does UDP use any timer? If so, please briefly describe how each is used. If not, please explain why it does not need one.

(a) No. HTTP is stateless protocol. Thus, it does not use any timers. Note if students mentioned conditional GET's timer or Persistent HTTP's timer, that's also correct.

(b) Yes. DNS uses a timer for sending queries to name servers. If there is no responses from name servers then the host send queries again.

(c) Yes. TCP uses a timer to retransmit the packets that are lost or delayed.

(d) No. UDP is unreliable data transport protocol.