Name: CENSORED

Date: 7/26/19

Assignment: HW3

Class: COMP-4320

1. In protocol rdt3.0, the ACK packets flowing from the receiver to the sender do not have sequence numbers (although they do have an ACK field that contains the sequence number of the packet they are acknowledging). Why is it that our ACK packets do not require sequence numbers?

**Answer:** In early versions of this protocol, the sender needed sequence numbers so the receiver can determine if packets are duplicates. In version 3.0, the sequence numbers became redundant since the receiver can determine a duplicate ACK by changing state after every received ACK, eliminating the need for the receiver to determine if packets are duplicates. In short, the burden of determining if a packet was lost became the responsibility of the sender instead of the receiver, and this trumped the use of ACK sequence numbers in terms of response times. This means that sequence numbers are only necessary on the packets and not the ACKs.

1. Suppose that the seven measured SampleRTT values (see Section 3.5.3) are 85 ms, 130 ms, 108 ms, 72 ms, 142 ms, 64 ms, and 153 ms. Compute the EstimatedRTT after each of these SampleRTT values is obtained, using a value of α = 0.2 and assuming that the value of EstimatedRTT was 110 ms just before the first of these seven samples were obtained. Compute also the DevRTT after each sample is obtained, assuming a value of β = 0.25 and assuming the value of DevRTT was 10 ms just before the first of these seven samples was obtained. Last, compute the TCP TimeoutInterval after each of these samples is obtained.

**Answer:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **EstimatedRTT** | **DevRTT** | **Timeout Interval** |
| **SampleRTT = 85ms** | (0.8)\*110 + (0.2)\*85 = 105ms | (0.75)\*10 + (0.25)\*|85 – 105| = 12.5ms | 105 + 4\*(12.5) = 155ms |
| **SampleRTT = 130ms** | (0.8)\*105 + (0.2)\*130 = 110ms | (0.75)\*12.5 + (0.25)\*|130 – 110| = 14.375ms | 110 + 4\*(14.375) = 167.5ms |
| **SampleRTT = 108ms** | (0.8)\*110 + (0.2)\*108 = 109.6ms | (0.75)\*14.375 + (0.25)\*|108 – 109.6| = 11.18125ms | 109.6 + 4\*(11.18125) = 154.325ms |
| **SampleRTT = 72ms** | (0.8)\*109.6 + (0.2)\*72 = 102.08ms | (0.75)\*11.18125 + (0.25)\*|72 – 102.08| = 15.9059375ms | 102.08 + 4\*(15.9059375) = 165.70375ms |
| **SampleRTT = 142ms** | (0.8)\*102.08 + (0.2)\*142 = 110.064ms | (0.75)\*15.9059375 + (0.25)\*|142 – 110.064| = 19.913453125ms | 110.064 + 4\*(19.913453125) = 189.7178125ms |
| **SampleRTT = 64ms** | (0.8)\*110.064 + (0.2)\*64 = 100.8512ms | (0.75)\*19.913453125 + (0.25)\*|64 – 100.8512| = 24.1478898438ms | 100.8512 + 4\*(24.1478898438) = 197.4427593752ms |
| **SampleRTT = 153ms** | (0.8)\*100.8512 + (0.2)\*153 = 111.28096ms | (0.75)\*(24.1478898438) + (0.25)\*|153 – 111.28096| = 28.5406773829ms | 111.28096 + 4\*(28.5406773829) = 225.4436695316ms |

1. Host A and B are communicating over a TCP connection, and Host B has already received all bytes up through byte 256. Suppose Host A then sends two segments to Host B back-to-back. The first and second segments contain 90 and 140 bytes of data, respectively. In the first segment, the sequence number is 257, the port number is 3120, and the destination port number is 5470. Host B send an acknowledgement whenever it receives a segment from Host A.
2. In the second segment sent from Host A to Host B, what are the sequence number, source port number, and destination port number?

**Answer:** The sequence number is equal to the first segment sequence number + the first segment’s amount of data, so the sequence number for the second segment is = 257 + 90 = 347, the source port number is 3120, and the destination port number is 5470.

1. If the second segment arrives before the first segment, in the acknowledgement of the first arriving segment, what is the acknowledgement number?

**Answer:** The ACK of the first arriving segment is the first segment’s sequence number (257) + the first segment’s amount of bytes (140) = 397, indicating that the receiver is still waiting for bytes 397 and onward.

1. If the first segment arrives before the second segment, in the acknowledgement of the first arriving segment, what are the acknowledgement number, the source port number, and the destination port number?

**Answer:** The acknowledgement number for the first segment is equal to the sequence number of the second segment (i.e., 347), the source port number is 5470 and the destination port number is 3120 (since it is an ACK being sent from Host B to A).

1. Suppose the two segments sent by A arrive in order at B. The first acknowledgement is lost and the second acknowledgement arrives after the first timeout interval. Draw a timing diagram, showing these segments and all other segments and acknowledgements sent. (Assume there is no additional packet loss.) For each segment in your figure, provide the sequence number and the number of bytes of data; for each acknowledgement that you add, provide the acknowledgement number.

**Answer:** **A close up of text on a whiteboard

Description automatically generated**

1. Consider the GBN and SR protocols. Suppose the sequence number space is of size *X*. What is the largest allowable sender window that will avoid the occurrence of problems such as that in Figure 3.27 in the above textbook for *each* of these protocols?

**Answer:** In order to avoid these problems shown in Figure 3.27, we need to make sure that the leading edge of the receiver’s window does not wrap around and overlap the trailing edge of the sender’s window. This means that our sequence number space X must be large enough to fit the entire receiver/sender windows without overlapping; to determine this, we must figure out how large a range of sequence numbers can be covered by the receiver/sender windows. Suppose that the lowest sequence number that the receiver is waiting for is packet Y. Now suppose that W represents the size of the window. Then, the receiver window is [Y, Y + W – 1] and it has received/ACKed packet Y – 1 and the W – 1 packets’ that came before that. If none of the W ACKs have been received by the sender yet, then ACK messages with values of [Y – W, Y – 1] may still be coming back to the sender. If no ACKs with these ACK numbers have been received by the sender, then the sender’s window would be [Y – W, Y – 1]. Thus, the lower edge of the sender’s window is Y – W, and the leading edge of the receiver’s window is Y + W – 1, giving us a total sequence number space of X = [Y – W, Y + W – 1]. In order for the leading edge of the receiver’s window to not overlap with the trailing edge of the sender’s window, the sequence number space must thus be big enough to accommodate 2W sequence numbers (i.e., the sequence number space X must be at least twice as large as the window size W); this holds true for both the GBN and SR protocols.

1. Consider a TCP connection has an initial Threshold of 32 kB and a Maximum Segment Size (MSS) of 6 kB. The receiver advertised window is 40 kB. Suppose all transmission attempts are successful except for a *timeout* at transmission *number 6* and a *triple duplicate ACK* received (for the same previously transmitted data) on the *number 12* transmission. The first transmission attempt is number 0. Find the size of the sender’s *congestion window* for the first 18 transmission attempts (number 0-17) assuming the sender’s TCP implementation is using the slow-start congestion control scheme.

**Answer:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Transition Number** | **Sender Window (kB)** | **Threshold (kB)** | **Receiver Window (kB)** |
| 0 | 6 | 32 | 40 |
| 1 | 12 | 32 | 40 |
| 2 | 24 | 32 | 40 |
| 3 | 32 | 32 | 40 |
| 4 | 38 | 32 | 40 |
| 5 | 40 | 32 | 40 |
| 6 | 40 | 32 | 40 |
| 7 | 6 | 20 | 40 |
| 8 | 12 | 20 | 40 |
| 9 | 20 | 20 | 40 |
| 10 | 26 | 20 | 40 |
| 11 | 38 | 20 | 40 |
| 12 | 40 | 20 | 40 |
| 13 | 20 | 20 | 40 |
| 14 | 26 | 20 | 40 |
| 15 | 32 | 20 | 40 |
| 16 | 38 | 20 | 40 |
| 17 | 40 | 20 | 40 |

1. Compare GBN, SR and TCP (no delayed ACK). Assume that the timeout values for all three protocols are sufficiently long such that 7 consecutive data segments and the 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 7 data segments to Host B, and the third segment (sent from A) is lost. In the end, all 7 data segments have been correctly received by Host B.
2. 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.

**Answer:** A close up of text on a whiteboard

Description automatically generated

1. If the timeout values for all three protocols are much longer than 7 RTT, then which protocol successfully delivers all 7 data segments in the shortest time interval?

**Answer:** TCP would be the fastest protocol in this case, since it would resend the lost packet after a triple ACK instead of waiting for the timeout like SR. Since this happens, the lost packet would more than likely be resent at some point before the timeout value is reached.