Scarlett Jones
CSCE 3530
Homework #4
Due 11/07/2016, 11:55 PM

1. Differentiate (at least three differences) between Go-Back-N (GBN) and Selective Repeat (SR) protocols.
    a. Go-Back-N
        i. Sender can have up to N unacked pakcets in pipeline
        ii. Receiver only sends cumulative ack, and doesn't ask a packet if there's a gap
        iii. Sender has a timer for oldest unacked packet, and when the timer expires it retransmits all unacked packets
    b. Selective Repeat
        i. Sender can have up to N unack'ed packets in pipeline
        ii. Receiver sends individual ack for each packet
        iii. Sender maintains timer for each unacked packet, and when each timer expires it retransmits only that unacked packet.
2. Explain each of the TCP segment fields with the TCP segment structure diagram.

<table>
<tr><td colspan="8">Source port number</td><td>Destination port number</td></tr>
<tr><td colspan="9">Sequence number</td></tr>
<tr><td colspan="9">Acknowledgment number</td></tr>
<tr><td>Header length</td><td>reserved</td><td>U</td><td>A</td><td>P</td><td>R</td><td>S</td><td>F</td><td>Receive window</td></tr>
<tr><td colspan="8">checksum</td><td>Urgent data pointer</td></tr>
<tr><td colspan="9">Options (variable length)</td></tr>
<tr><td colspan="9">Application data (variable length)</td></tr>
</table>

    a. Source port # is needed to tell the receiver where the segment is coming from. This is 16 bits.
    b. Destination port # is needed to tell the sender where to send the segment. This is 16 bits.
    c. Sequence number starts at a random number on the sender. This counts by bytes. The numbers are the byte stream number of first byte in segment's data. The receiver uses this sequence number plus 1 to be its next acknowledgment number. This is 32 bits.
    d. Acknowledgement number starts at a random number on the sender. This represents the sequence number of the next byte expected from the other side of the connection. This is a cumulative ack. The receiver uses this acknowledgement number as its next sequence number. This is 32 bits.
    e. Head len is the header length. This is 4 bit.
    f. Not used is reserved. This is 6 bits.
    g. UAPRSF – These are each a bit that can be set to 1 to activate the bit.
        i. The U means urgent data, and is generally not used.
        ii. The A tells you whether or not the ACK # is valid.
        iii. The P means push data now, and is generally not used.
        iv. The R means Reset. This is for connection establishment.
        v. The S means Sync. This is for connection establishment.
        vi. The F means Final. This is for connection establishment.

h. Receive window is the number of bytes the receiver is willing to accept. This is 16 bits.
i. Checksum is the internet checksum (as in UDP). This is 16 bits.
j. Urg data pointer is an urgent data pointer. This is 16 bits.
k. Options is of variable length. There could be no options.
l. Application data is of variable length. There could be no data here.

3. Assume that the timeout values for GBN, SR, and TCP protocols are sufficiently long such that 7 consecutive data segments can be received (if not lost in the channel) by the receiving host (Host B) and their corresponding ACKs can received by the sending host (Host A). Suppose Host A sends 7 data segments to Host B, and the 3rd segment (sent from A) is lost. In the end, all 7 data segments have been correctly received by Host B. For GBN, SR, and TCP (no delayed ACK) protocols how many segments has Host A sent in total and how many ACKs has Host B sent in total? What are their sequence numbers?
   a. Go-Back-N
      i. Host A sent 16 segments in total. Sequence numbers were: seg0, seg1, seg2, seg3, seg4, seg5, seg6, seg7, seg8, seg2, seg3, seg4, seg5, seg6, seg7, seg8.
      ii. Host B sent 15 ACKs in total. Sequence numbers were: ack0, ack1, ack1, ack1, ack1, ack1, ack1, ack2, ack3, ack4, ack5, ack6, ack7, ack8.
   b. Selective Repeat
      i. Host A sent 10 segments in total. Sequence numbers were: seg0, seg1, seg2, seg3, seg4, seg5, seg6, seg7, seg8, seg2.
      ii. Host B sent 9 ACKs in total. Sequence numbers were: ack0, ack1, ack3, ack4, ack5, ack6, ack7, ack8, ack2.
   c. TCP
      i. Host A sent 8 segments in total. Sequence numbers were: seg0, seg1, seg2, seg3, seg4, seg5, seg2, seg6.
      ii. Host B sent 7 ACKs in total. Sequence numbers were: ack1, ack2, ack2, ack2, ack2, ack6, ack7.

4. Five values of SampleRTT are 35 ms, 45 ms, 50 ms, 65 ms, and 70 ms. Compute the EstimatedRTT, DevRTT, and Timeoutinterval after each of these SampleRTT values is obtained. Assume α = 0.125 and β = 0.25. Initial value of EstimatedRTT was 30 ms and initial value of DevRTT was 40 ms just before the first of these five samples was obtained.

$$EstimatedRTT = (1 - \alpha) * EstimatedRTT + \alpha * SampleRTT$$

$$DevRTT = (1 - \beta) * DevRTT + \beta * |SampleRTT - EstimatedRTT|$$

$$TimoutInterval = EstimatedRTT + 4 * DevRTT$$

	SampleRTT	EstimatedRTT	DevRTT	TimoutInterval
Initial		30ms	40ms	
1st	35ms	$(1 - 0.125) * 30 * 10^{-3} + 0.125 * 35 * 10^{-3} = 30.625ms$	$(1 - 0.25) * 40 * 10^{-3} + 0.25 * \|35 * 10^{-3} - 30.625 * 10^{-3}\| = 31.094ms$	$30.625 * 10^{-3} + 4 * 31.094 * 10^{-3} = 155.001ms$

2nd	45ms	$(1-0.125)*30.625*10^{-3}+0.125*45*10^{-3}=32.422ms$	$(1-0.25)*31.094*10^{-3}+0.25*\lvert 45*10^{-3}-32.422*10^{-3}\rvert=26.465ms$	$32.422*10^{-3}+4*26.465*10^{-3}=138.282ms$
3rd	50ms	$(1-0.125)*32.422*10^{-3}+0.125*50*10^{-3}=34.619ms$	$(1-0.25)*26.465*10^{-3}+0.25*\lvert 50*10^{-3}-34.619*10^{-3}\rvert=23.694ms$	$34.619*10^{-3}+4*23.694*10^{-3}=129.395ms$
4th	65ms	$(1-0.125)*34.619*10^{-3}+0.125*65*10^{-3}=38.417ms$	$(1-0.25)*23.694*10^{-3}+0.25*\lvert 65*10^{-3}-38.417*10^{-3}\rvert=24.416ms$	$38.417*10^{-3}+4*24.416*10^{-3}=136.081ms$
5th	70ms	$(1-0.125)*38.417*10^{-3}+0.125*70*10^{-3}=42.365ms$	$(1-0.25)*24.416*10^{-3}+0.25*\lvert 70*10^{-3}-42.365*10^{-3}\rvert=25.221ms$	$42.365*10^{-3}+4*25.221*10^{-3}=143.249ms$

5. What is TCP fast retransmit? Use an example and explain the working of TCP fast retransmit.
    a. TCP fast retransmit will resend an unacked segment with the smallest sequence number after the sender received 3 acks for the same data. This is done because it is likely that the unacked segment is lost, and this way we don't have to wait for the timeout to pass before sending the data that is lost.
    b. For example, if the sender sends 8 bytes of data starting at sequence number 92, the next ack sent by the receiver will be ack 100. This is asking the sender for the next data. The sender sends out 20 bytes of data at sequence number 100, but it fails to reach the receiver. The receiver continues to send 3 more acks at number 100. This is all before the timeout completes. If we waited to retransmit the data, the sender would waste time continually sending the next amount of data that the receiver has not yet asked for. This risks filling up the buffer and losing data. We don't want to have to retransmit data that has already been successfully sent. So after these 3 duplicate ack 100's, the sender retransmits the sequence number 100 with 20 bytes of data before the timeout has passed.
6. Explain how TCP flow control is done?
    a. The receiver controls the sender so that the sender won't overflow the reciever's buffer by transmitting too much too fast. The receiver has a rwnd value in the TCP header of receiver to sender segments. This rwnd value tells the amount of free buffer space that the receiver has to the sender. The sender then limits the amount of unacked data to the receiver's rwnd value. This guarantees the receive buffer will not overflow. The receive buffer is made up of the buffered data plus the free buffer space. The receive buffer (RcvBuffer) value is set via socket options, but is typically set to 4096 bytes. Many operating systems autoadjust the receive buffer.
7. What are the steps involved in opening and closing a TCP connection?
    a. A TCP connection must be opened with the TCP 3-way handshake. The client and the server must each choose an initial random sequence number. Say the client picked x and

the server choose y. The SYN bit must be set to 1 to establish a connection. Once the server receives the SYN bit from the client, asking to make a connection, the server sends an acknowledgment to the client. The ACK bit gets set to 1. The ACK number is the client's sequence number, x, plus 1. After the client receives the SYNACK, meaning the server is available for connection, the client sends the ACK for the SYNACK, meaning the client still wants to connect. The client's ACK number is the server's sequence number, y, plus 1. Once the server receives the client's acknowledgement, it knows that the client is still ready to connect and a connection is established.

b. To close a TCP connection, the client and the server must each close their side of the connection. The client sends the server a segment with the FIN bit set to 1 at sequence number x. The client can no longer send but can receive data. The server sends the client an acknowledgment with the ACK bit set to 1 and at an ACK number equal to the client's sequence number, x, plus 1. The server can still send data at this point. The client is now waiting for the server to close. The server then sends a segment to the client with the FIN bit set to 1 at sequence number y. The server can no longer send data now. The client sends an ACK to the server with ACK bit set to 1 at an ACk number equal to the server's sequence number, y, plus 1. Then the connection is closed on the server side first. The client side waits to close the connection for 2 times the maximum segment lifetime, just in case there was some data from the server that the client hadn't received yet.

8. Differentiate (at least two differences) between TCP Reno and TCP Tahoe.
    a. TCP Reno
        i. Cuts congestion window down to half of the congestion window where the loss was detected.
        ii. Continues as linear increase, starting from the new threshold.
    b. TCP Tahoe
        i. Cuts congestion window down to 1 MSS.
        ii. Resets to the slow start state and continues as exponential increase until threshold is reached, then continues as linear increase.
9. A 80 Mbps (about to be congested) non-buffered link is used to send a huge file between two hosts. The receiving host has a large buffer than the congestion window. Assume the turnaround time is 4 ms, and the TCP Reno connection is always in congestion avoidance phase. Answer the following questions.
    a. What is the maximum window size in bytes that this TCP connection can achieve?
        i. $W = RTT * Rate = (4 * 10^{-3}) * (80 * 10^{6}) = 320{,}000\ bits$
    b. What is the average window size in bytes and the average throughput in bps of this TCP connection?
        i. $Avg.Window = \frac{3}{4} * W = \frac{3}{4} * (320{,}000) = 240{,}000 bits$
        ii. $Avg.Throughput = \frac{3}{4} * \frac{W}{RTT} = \frac{3}{4} * \frac{320{,}000}{4*10^{-3}} = 60{,}000{,}000 bps = 60 Mbps$
    c. How long would it take for this TCP connection to reach its maximum window again after recovering from a packet loss if the TCP segment size is 2,000 bytes?
        i. $1 MSS = 2{,}000 Bytes$

ii. $Max\ window\ size\ in\ segments = \frac{W}{1MSS} = \frac{320{,}000}{2000*8} = 20\ Segments$

iii. $Recovery\ Time = \frac{20}{2} = 10 segments * RTT = 10 * (4 * 10^{-3}) = 40ms$

d. What is the maximum window size and average window size in segments if the TCP segment size is 2,000 bytes?

i. $Max\ window\ size\ in\ segments = \frac{W}{1MSS} = \frac{320{,}000}{2000*8} = 20\ Segments$

ii. $Avg.window\ size\ in\ segments = \frac{Avg.W}{1MSS} = \frac{240{,}000}{2000*8} = 15\ Segments$