Network HW2：
weihanchu (wcm350)

Review question 3-r14:

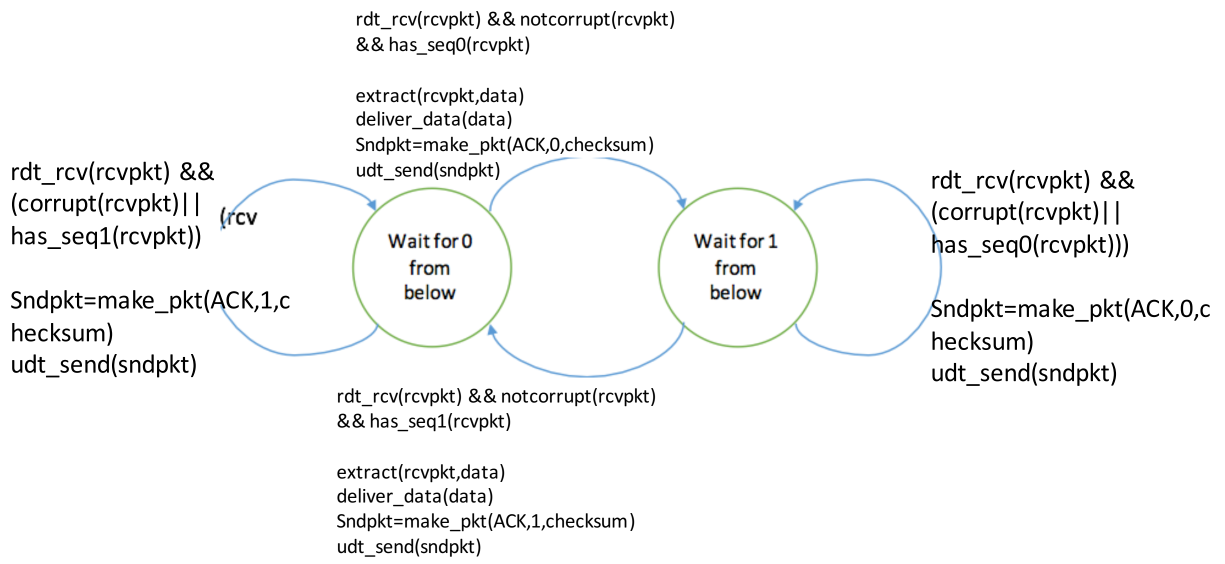
a. False
b. False, rwnd changes all the time
c. True
d. False, the sequence number are over the stream of transmitted bytes and not over the series of transmitted segments
e. True, the header has a 16 bytes for receive window
f. False,
g. False, The acknowledgment number that Host A puts in its segment is the sequence number of the next byte Host A expecting form Host B.

P3: UDP and TCP use 1s complement for their checksums. Suppose you have the following three 8-bit bytes: 01010011, 01100110, 01110100. What is the 1s complement of the sum of these 8-bit bytes? (Note that although UDP and TCP use 16-bit words in computing the checksum, for this problem you are being asked to consider 8-bit sums.) Show all work. Why is it that UDP takes the 1s complement of the sum; that is, why not just use the sum? With the 1s complement scheme, how does the receiver detect errors?

01010011+01100110+01110100=01110100. note that this last addition had overflow, which should be wrapped around, we can get 00101110. The 1s complement is obtained by converting all the 0s to 1s and all 1s to 0s. Then we finally get 11010001.

The receiver uses the checksum to check whether there are any error. At the receiver, all the four 16-bit words are added, including the checksum. If no errors are introduced into the packet, then clearly the sum at the receiver will be 1111111111111111. If one of the bits is a 0, then we know that errors have been introduced into the packet. With this method, all one-bite errors will be detected, but two-bit errors may not be detected.

P8:



P12:The sender side of rdt3.0 simply ignores (that is, takes no action on) all received packets that are either in error or have the wrong value in the acknum field of an acknowledgment packet. Suppose that in such circumstances, rdt3.0 were simply to retransmit the current data packet. Would the protocol still work? (Hint : Consider what would happen if there were only bit errors; there are no packet losses but premature timeouts can occur. Consider how many times the n th packet is sent, in the limit as n approaches infinity.)

In the previous situation, if the receiver send send a wrong acknowledgement, the sender will just ignore and do nothing. It will retransmit the data until the time out. Now in this problem the sender will immediately retransmit the current data packet once it receives a wrong acknowledgment. The protocol will still work. And the premature occurs, which reduces the waiting time and the current data packet will be sent again. the nth packet will be sent infinity times as n approach infinity.

P22:

a. if the sender has received all the other ACKeds, the sender's window is [k,k+3];
if the sender hasn't received any ACKeds, so the sender need to resend all the previous packets, the window thus is [k-4,k-1]

In conclusion, the possible sets of sequence number inside the sender's window is[k-4,k]

b. because the receiver is waiting for packet k, which means is has received all the previous packet. So all possible messages currently propagating back to the sender at time t is [k-4,k-1]

P24:

a.True. For example, a sender has a window whose size is 3. This sender sent 1,2 two packets. Then the receiver received 1,2 and sent ACKs 1,2 to the sender. But the sender didn't receive the ACKS and time out and resent 1,2. After that, the sender received the ACKs 1,2 the receiver sent. Then the receiver received a duplicate ACK 1,2 and sent the ACK1,2 again to the sender. But now the sender moves to 3,4 already, so the packet fall outside the window.

b. True, same thing happened here as the above.

c. True

d. True, alternating bit protocol which is also known as rdt3.0, will work in the same way as GBN when the window size is 1.

P32:

a.

$$Estimate\ RTT1 = SampleRTT4$$

$$Estimate\ RTT2 = (1-\alpha)SampleRTT4 + \alpha SampleRTT3$$

$$Estimate\ RTT3 = (1-\alpha)((1-\alpha)SampleRTT4 + \alpha SampleRTT3) + \alpha SampleRTT2$$
$$= (1-\alpha)^2 SampleRTT4 + \alpha(1-\alpha)SampleRTT3 + \alpha SampleRTT2$$

$$Estimate\ RTT4 = (1-\alpha)^3 SampleRTT4 + \alpha(1-\alpha)^2 SampleRTT3 + \alpha(1-\alpha)SampleRTT2 + \alpha SampleRTT1$$

The final result is

$$(1-\alpha)^3 SampleRTT4 + \alpha(1-\alpha)^2 SampleRTT3 + \alpha(1-\alpha) SampleRTT2 + \alpha SampleRTT1$$

b.

$$Estimate\ RTT4 = (1-\alpha)^{n-1} SampleRTTn + \alpha(1-\alpha)^{n-2} SampleRTTn-1 + \alpha(1-\alpha)^{n-3} SampleRTTn-2 + ...+ \alpha SampleRTT1$$
$$= (1-\alpha)^{n-1} SampleRTTn + \alpha \sum_{j=1}^{n-1} (1-\alpha)^{j-1} SampleRTTj$$

c.

$$Estimate\ RTT(\infty) = (1-\alpha)^{\infty-1} SampleRTT\infty + \alpha \sum_{j=1}^{\infty-1} (1-\alpha)^{j-1} SampleRTTj$$

The averaging procedure decays exponentially.

P40:

a. [1-6] ,[23-26],because in this times the window size increase exponentially
b. [6-16], [17-22], because in this times increase linearly
c. triple duplicate AC, because then congestion avoidance is operating
d. time out, because then slow start is operating
e. 32,because the initial threshold is the congestion window size when congestion avoidance begins
f. 21, because at 16 the congestion window size is 42 and when there is a packet loss is detected, the threshold is set to half. so the threshold is 21 now during the 18th transmission round
g. 14, same reason as above question
h. 7th. because in the 1st, 1 is sent. in the 2nd, 2-3 are sent. in the 3rd,4-7 are sent . in the 4th, 8-15 are sent. in the 5th, 16-31 are sent. in the 6th 32-63 are sent. in the 7th,64-127 are sent. so the 70th is sent in 7th transmission round
i. 4, because the threshold will be set half.
j. The congestion window will be set to 1 while the threshold will not change
k. 52, because 1+2+4+8+16+21=52