Chapter 3 - Review Questions

Homework Problems and Questions

• Sections 3.1 - 3.3

1. Consider a TCP connection between host A and host B. Suppose that the TCP segments traveling from host A to host B have source port number \( x \) and destination port number \( y \). What are the source and destination port numbers for the segments traveling from host B to host A?

2. Describe why an application developer may choose to run an application over UDP rather than TCP.

3. Is it possible for an application to enjoy reliable data transfer even when the application runs over UDP? If so, how?

• Section 3.5

4. True or False:

   a. Host A is sending host B a large file over a TCP connection. Assume host B has no data to send A. Host B will not send acknowledgments to host A because host B cannot piggyback the acknowledgments on data.

   b. The size of the TCP RcvWindow never changes throughout the duration of the connection.

   c. Suppose host A is sending host B a large file over a TCP connection. The number of unacknowledged bytes that A sends cannot exceed the size of the receive buffer.

   d. Suppose host A is sending a large file to host B over a TCP connection. If the sequence number for a segment of this connection is \( m \), then the sequence number for the subsequent segment will necessarily be \( m - 1 \).

   e. The TCP segment has a field in its header for RcvWindow.

   f. Suppose that the last SampleRTT in a TCP connection is equal to 1 sec. Then Timeout for the connection will necessarily be set to a value \( \geq 1 \) sec.

   g. Suppose host A sends host B one segment with sequence number 38 and 4 bytes of data. Then in this same segment the acknowledgment number is necessarily 42.

5. Suppose A sends two TCP segments back-to-back to B. The first segment has sequence number 90; the second has sequence number 110. (a) How much data is the first segment? (b) Suppose that the first segment is lost, but the second segment arrives at B. In the acknowledgment that B sends to A, what will be the acknowledgment number?

6. Consider the Telnet example discussed in Section 3.5. A few seconds after the user types the letter “C” the user types the letter “R”. After typing the letter “R” how many segments are sent and what is put in the sequence number and acknowledgment fields of the segments?
• Section 3.7

7. Suppose two TCP connections are present over some bottleneck link of rate $R$ bps. Both connections have a huge file to send (in the same direction over the bottleneck link). The transmissions of the files start at the same time. What is the transmission rate that TCP would like to give to each of the connections?

8. True or False: Consider congestion control in TCP. When a timer expires at the sender, the threshold is set to one half of its previous value.

Problems

1. Suppose client A initiates a Telnet session with server S. At about the same time, client B also initiates a Telnet session with server S. Provide possible source and destination port numbers for:

   a. the segments sent from A to S.

   b. the segments sent from B to S.

   c. the segments sent from S to A.

   d. the segments sent from S to B.

   e. If A and B are different hosts, is it possible that the source port number in the segments from A to S is the same as that from B to S?

   f. How about if they are the same host?

2. UDP and TCP use 1's complement for their checksums. Suppose you have the following three 8-bit words: 01010101, 01110000, 11001100. What is the 1's complement of the sum of these words? Show all work. Why is it that UDP takes the 1's complement of the sum, that is, why not just use the sum? With the 1's complement scheme, how does the receiver detect errors? Is it possible that a 1-bit error will go undetected? How about a 2-bit error?

3. Consider our motivation for correcting protocol rtd2.l. Show that this receiver, when operating with the render shown in Figure 3.12, can lead the render and receiver to enter into a deadlock state, where each is waiting for an event that will never occur.
4. 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?

5. Draw the FSM for the receiver side of protocol rdt 3.0.

6. Give a trace of the operation of protocol rdt3.0 when data packets and acknowledgments packets are garbled. Your trace should be similar to that used in Figure 3.16.

7. Consider a channel that can lose packets but has a maximum delay that is known. Modify protocol rdt2.1 to include receiver timeout and retransmit. Informally argue why your protocol can communicate correctly over this channel.

8. 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 to simply retransmit the current data packet. Would the protocol still work? (Hint: Consider what would happen in the case that there are only bit errors; there are no packet losses but premature timeouts can occur. Consider how many times the nth packet is sent, in the limit as n approaches infinity.)
9. Consider the cross-country example shown in Figure 3.17. How big would the window size have to be for the channel utilization to be greater than 90%?

10. Design a reliable, pipelined, data transfer protocol that uses only negative acknowledgments. How quickly will your protocol respond to lost packets when the arrival rate of data to the sender is low? Is high?

11. In the generic selective repeat protocol that we studied in Section 3.4.4, the sender transmits a message as soon as it is available (if it is in the window) without waiting for an acknowledgment. Suppose now that we want a SR protocol that sends messages two at a time. That is, the sender will send a pair of messages, and will send the next pair of messages only when it knows that both messages in the first pair have been received correctly. Suppose that the channel may lose messages but will not corrupt or reorder messages. Design an error-control protocol for the unidirectional reliable transfer of messages. Give a FSM description of the sender and receiver. Describe the format of the packets sent between sender and receiver, and vice versa. If you use any procedure calls other than those in Section 3.4 (e.g., udt_send( ), start_timer( ), rdt_rcv( ), etc.), clearly state their actions. Give an example (a timeline trace of sender and receiver) showing how your protocol recovers from a lost packet.

12. Consider a scenario in which a host, A, wants to simultaneously send messages to hosts B and C. A is connected to B and C via a broadcast channel - a packet sent by A is carried by the channel to both B and C. Suppose that the broadcast channel connecting A, B, and C can independently lose and corrupt messages (and so, for example, a message sent from A might be correctly received by B, but not by C). Design a stop-and-wait-like error-control protocol for reliably transferring a packet from A to B and C, such that A will not get the data from the upper layer until it knows that both B and C have correctly received the current packet. Give FSM descriptions of A and C. (Hint: The FSM for B should be essentially the same as for C.) Also, give a description of the packet formats(s) used.

13. Consider the Go-Back-N protocol with a sender window size of 3 and a sequence number range of 1,024. Suppose that at time $t$, the next in-order packet that the receiver is expecting has a sequence number of $k$. Assume that the medium does not reorder messages. Answer the following questions:

   a. What are the possible sets of sequence numbers inside the sender's window at time $t$? Justify your answer.

   b. What are all possible values of the ACK field in the message currently prop-agating back to the sender at time $t$? Justify your answer.

14. Suppose we have two network entities, A and B. B has a supply of data messages that will be sent to A according to the following conventions. When A gets a request from the layer above to get the next data (D) message from B, A must send a request (R) message to B on the A-to-B channel. Only when B receives an R message can it send a data (D) message back to A on the B-to-A channel. A should deliver exactly one copy of each D message to the above layer. R messages can be lost (but not corrupted) in the A-to-B channel; D messages, once sent are always delivered correctly. The delay along both channels is unknown and variable.

   Design (give a FSM) description of a protocol that incorporates the appropriate mechanisms to compensate for the loss-prone A-to-B channel and implements message passing to the above layer at entity A, as discussed above. Use only those mechanisms that are absolutely necessary.

15. Consider the Go-Back-N and selective-repeat protocols. Suppose the sequence number space is of size $k$. What is the largest allowable sender window that will avoid problems such as that in Figure 3.26 from occurring for each of these protocols?
16. Answer true or false to the following questions and briefly justify your answer:

a. With the selective repeat protocol, it is possible for the sender to receive an ACK for a packet that falls outside of its current window.

b. With Go-Back-N, it is possible for the sender to receive an ACK for a packet that falls outside of its current window.

c. The alternating-bit protocol is the same as the selective-repeat protocol with a sender and receiver window size of 1.

d. The alternating-bit protocol is the same as the Go-Back-N protocol with a sender and receiver window size of 1.

17. Consider transferring an enormous file of $L$ bytes from host A to host B. Assume an MSS of 1460 bytes.

a. What is the maximum value of $L$ such that TCP sequence numbers are not exhausted? Recall that the TCP sequence number field has four bytes.

b. For the $L$ you obtain in (a), find how long it takes to transmit the file. Assume that a total of 66 bytes of transport, network, and data-link header are added to each segment before the resulting packet is sent out over a 10 Mbps link. Ignore flow control and congestion control, so A can pump out the segments back-to-back and continuously.

18. In Figure 3.31, we see that TCP waits until it has received three duplicate ACK before performing a fast retransmit. Why do you think the TCP designers chose not to perform a fast retransmit after the first duplicate ACK for a segment is received?

19. Consider the TCP procedure for estimating RTT. Suppose that $x = .1$. Let $\text{SampleRTT}_1$ be the most recent sample RTT, let $\text{SampleRTT}_2$ be the next most recent sample RTT, etc. (a) For a given TCP connection, suppose four acknowledgments have been returned with corresponding sample RTTs $\text{SampleRTT}_4$, $\text{SampleRTT}_3$, $\text{SampleRTT}_2$, and $\text{SampleRTT}_1$. Express $\text{EstimatedRTT}$ in terms of the four sample RTTs. (b) Generalize our formula for $n$ sample round-trip times. (c) For the formula in part (b) let $n$ approach infinity. Comment on why this averaging procedure is called an exponential moving average.

20. Refer to Figure 3.51 that illustrates the convergence of TCP’s additive increase, multiplicative decrease algorithm. Suppose that instead of a multiplicative decrease, TCP decreased the window size by a constant amount. Would the resulting additive increase additive decrease converge to an equal share algorithm? Justify your answer using a diagram similar to Figure 3.51.

21. Recall the idealized model for the steady-state dynamics of TCP. In the period of time from when the connection’s window size varies from $(W-MSS)/2$ to $W-MSS$, only one packet is lost (at the very end of the period).

a. Show that the loss rate is equal to

\[
L = \text{loss rate} = \frac{1}{\frac{3}{8} W + \frac{3}{4} w}
\]
b. Use the above result to show that if a connection has loss rate \( L \), then its average bandwidth is approximately given by:

\[
\text{Average bandwidth of connection} \sim 1.22 \cdot \text{MSS} / \left[ \text{RTT} \cdot \sqrt{L} \right]
\]

22. Consider sending an object of size \( O = 100 \) Kbytes from server to client. Let \( S = 536 \) bytes and \( \text{RTT} = 100 \) msec. Suppose the transport protocol uses static windows with window size \( W \).

a. For a transmission rate of 28 Kbps, determine the minimum possible latency. Determine the minimum window size that achieves this latency.

b. Repeat (a) for 100 Kbps.

c. Repeat (a) for 1 Mbps.

d. Repeat (a) for 10 Mbps.

23. Suppose TCP increased its congestion window by two rather than by one for each received acknowledgment during slow start. Thus, the first window consists of one segment, the second of three segments, the third of nine segments, etc. For this slow-start procedure:

a. Express \( K \) in terms of \( O \) and \( S \).

b. Express \( Q \) in terms of \( \text{RTT} \), \( S \), and \( R \).

c. Express latency in terms of \( P = \min(K - 1, Q) \), \( O \), \( R \), and \( \text{RTT} \).

24. Consider the case \( \text{RTT} = 1 \) second and \( O = 100 \) Kbytes. Prepare a chart (similar to the charts in Section 3.5.2) that compares the minimum latency \( (O/R + 2 \ \text{RTT}) \) with the latency with slow start for \( R = 28 \) Kbps, 100 Kbps, 1 Mbps, and 10 Mbps.

25. True or False?

a. If a Web page consists of exactly one object, then nonpersistent and persistent connections have exactly the same response time performance.

b. Consider sending one object of size \( O \) from server to browser over TCP. If \( O > S \), where \( S \) is the maximum segment size, then the server will stall at least once.

c. Suppose a Web page consists of 10 objects, each of size \( O \) bits. For persistent HTTP, the RTT portion of the response time is 20 RTT.

d. Suppose a Web page consists of 10 objects, each of size \( O \) bits. For nonpersistent HTTP with 5 parallel connections, the RTT portion of the response time is 12 RTT.

26. The analysis for dynamic windows in the text assumes that there is one link between server and client. Redo the analysis for \( T \) links between server and client. Assume the network has no congestion, so the packets experience no queuing delays. The packets do experience a store-and-forward delay, however. The
definition of RTT is the same as that given in the section on TCP congestion control. (Hint: The time for the server to send out the first segment until it receives the acknowledgment is \( TS/R + RTT \).)

27. Recall the discussion at the end of Section 3.7.3 on the response time for a Web page. For the case of nonpersistent connections, determine a general expression for the fraction of the response time that is due to TCP slow start.

28. With persistent HTTP all objects are sent over the same TCP connection. As we discussed in Chapter 2, one of the motivations behind persistent HTTP (with pipelining) is to diminish the effects of TCP connection establishment and slow start on the response time for a Web page. In this problem we investigate the response time for persistent HTTP. Assume that the client requests all the images at once, but only when it has received the *enrich* HTML base page. Let \( M + I \) denote the number of objects and let \( O \) denote the size of each object.

a. Argue that the response time takes the form \( (M + I)O/R + 3RTT + \text{latency due to slow-start} \). Compare the contribution of the RTTs in this expression with that in nonpersistent HTTP.

b. Assume that \( K = 1092 (O/S + 1) \) is an integer; thus, the last window of the base HTML file transmits an entire window’s worth of segments, that is, window \( K \) transmits \( 2^{K-1} \) segments. Let \( P' = \min\{Q, K' - 1\} \) and

\[
K' \left\lfloor \log_2 \left( \frac{M + I}{O/S + 1} \right) \right\rfloor
\]

Note that \( K' \) is the number of windows that cover an object of size \( (M + I)O \) and \( P' \) is the number of stall periods when sending the large object over a single TCP connection. Suppose (incorrectly) the server can send the images without waiting for the formal request for the images from the client. Show that the response time is that of sending one large object of size \( (M + I)O \):

\[
\text{Approx response time} = 2RTT + \frac{(M + I)O}{R} + P' \left[ RTT + \frac{S}{R} \right] - \left( 2^{P'} - 1 \right) \frac{S}{R}
\]

c. The actual response time for persistent HTTP is somewhat larger than the approximation. This is because the server must wait for a request for the images before sending the images. In particular, the stall time between the \( K \)th and \( (K + 1) \)st window is not \( [S/R + RTT + 2^{K-1}(S/R)^+] \) but is instead RTT. Show that

\[
\text{Response time} = 3RTT + \frac{(M + I)O}{R} + P' \left[ RTT + \frac{S}{R} \right] - \left( 2^{P'} - 1 \right) \frac{S}{R} - \left[ \frac{S}{R} + RTT - \frac{S}{R} 2^{K-1} \right] +
\]
29. Consider the scenario of $RTT = 100$ msec, $O = 5$ Kbytes, $S = 536$ bytes, and $M = 10$. Construct a chart that compares the response times for nonpersistent and persistent connections for 28 Kbps, 100 Kbps, 1 Mbps, and 10 Mbps. Note that persistent HTTP has substantially lower response time than nonpersistent HTTP for all the transmission rates except 28 Kbps.

30. Repeat the above question for the case of $RTT = 1$ sec, $O = 5$ Kbytes, $S = 536$ bytes, and $M = 10$. Note that for these parameters, persistent HTTP gives a significantly lower response time than nonpersistent HTTP for all the trans rates.

31. Consider now nonpersistent HTTP with parallel TCP connections. Recall that browsers typically operate in this mode when using HTTP/1.0. Let $X$ denote the maximum number of parallel connections that the client (browser) is permitted to open. In this mode, the client first uses one TCP connection to obtain the base HTML file. Upon receiving the base HTML file, the client establishes $M/X$ sets of TCP connections, with each set having $X$ parallel connections. Argue that the total response time takes the form:

$$\text{Response time} = (M + 1)O/R + 2(M/X + 1)RTT + \text{latency due to slow-start stalling}$$

Compare the contribution of the term involving RTT to that of persistent connections and nonpersistent (nonparallel) connections.

### Discussion Question

1. Consider streaming stored audio. Does it make sense to run the application over UDP or TCP? Which one does RealNetworks use? Why? Are there any other streaming stored audio products? Which transport protocol do they use and why?

### Programming Assignment

In this programming assignment, you will be writing the sending and receiving transport-level code for implementing a simple reliable data-transfer protocol – for either the alternating-bit rotocol or a Go-Back-N protocol. This should be fun since your implementation will differ very little from what would be required in a realworld situation.

Since you presumably do not have standalone machines (with an OS that you can modify), your code will have to execute in a simulated hardware/software environment. However, the programming interface provided to your routines (that is, the code that would call your entities form above (from layer 5) and from below (from layer 3)) is very close to what is done in na actual UNIX environment. (Indeed, the software interfaces described in this programming assignment are much more realistic than the infinite-loop senders and receivers that many textbooks describe.) Stopping/starting of timers are also simulated, and timer interrupts will cause your timer-handing routine to be activated.

You can find full details of the programming assignment, as well as C code that you will need to create the simulated hardware/software environment at [http://www.awl.com/kurose-ross](http://www.awl.com/kurose-ross)