Kevin Stine
Problems 3

Review Questions:

R2. Consider a planet where everyone belongs to a family of six, every family lives in it's own house, each house has a unique address, and each person in a given house has a unique name

a. Using the solution to Problem R1 above as inspiration, describe a protocol that the delegates can use to deliver letters from a sending family member to a receiving family member.

The family would be required to give the letter to the delegate with only the address of the destination house written on the envelope. Then, they would inform the delegate who the name of the recipient is. The delegate would write down who the recipient is so they know who to deliver it to on the letter. The letter is then given to the mail service to be taken to the destination. The delegate on the receiving end would then open the letter to see who it is addressed to and give it to that particular family member.

b. In your protocol, does the mail service ever have to open the envelope and examine the letter in order to provide its service?

In this protocol, the mail service does not need to open up the letter since they only need to look at the address to know who to deliver to.

R4. Describe why an application developer might choose to run an application over UDP rather than TCP.

The developer might not care about all the bits being received in the correct order and in tact, as they may just prefer the faster delivery speed that UDP offers since it does not have TCP's congestion control.

R9. In our rdt protocols, why did we need to introduce sequence numbers?

Sequence numbers are required for a receiver to find out whether an arriving packet contains new data or is instead being sent as a retransmission.

Problems:

P1. 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:

- Source Port number: 485
- Destination Port number: 23

b. The segments sent from B to S
   - Source port number: 515
   - Destination port number: 23

c. The segments sent from S to A
   - Source port number: 23
   - Destination port number: 485

d. The segments sent from S to B
   - Source port number: 23
   - Destination port number: 515

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?
   - Yes it is possible since they are from different hosts

f. How about if they are the same host?
   - No since they would be from the same host.

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?

```
    01010011
+   01100110
    10111001

    10111001
+   01110100    (Must wrap around since it overflows)
    00101110
```

1s Complement: 11010001

P15. 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 98 percent? Suppose that the size of a packet is 1,500 bytes, including both header fields and data.

$$\frac{1500*8\ bits}{10^9 bps} = 1.2 * 10^{-5} seconds$$

$$98 = \frac{(.012*n)}{30.012} \ \ -> n \ = 2451\ packets$$

P17. Consider two network entities, A and B, which are connected by a perfect bi-directional channel (i.e., any message sent will be received correctly; the

channel will not corrupt, lose, or re-order packets). A and B are to deliver data messages to each other in an alternating manner: First, A must deliver a message to B, then B must deliver a message to A, then A must deliver a message to B and so on. If an entity is in a state where it should not attempt to deliver a message to the other side, and there is an event like rdt_send(data) call from above that attempts to pass data down for transmission to the other side, this call from above can simply be ignored with a call to rdt_unable_to_send(data), which informs the higher layer that it is currently not able to send data. [Note: This simplifying assumption is made so you don't have to worry about buffering data.]

P24. Answer true or false to the following questions and briefly justify your answer:

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

Yes it is possible. With a window size of 3 that sends packets 1, 2, 3, at t 0. At t1, the receiver ACKS 1, 2, 3. At t2 the sender times out and resends 1, 2 and 3. At t3 the receiver duplicates and re-acknowledges 1, 2 and 3. At t4 the sender receives the ACKs that the receiver sent at t1 and advances its window to 4, 5, 6. At t5 the sender receives ACKs 1, 2, 3 the receiver sent at t2 which are outside of it's current window.

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

Yes it is possible. Using the same situation from question a, you can see that the ACK falls outside of the current window.

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

With a window size of 1, the alternating bit protocol and the SR protocol are functionally the same. The window size of 1 prevents out of order packets within the window.

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

With a window wise of 1, the alternating bit protocol and the GBN protocol are functionally the same. The window size of 1 prevents out of order packets within the window

P38. In our description of TCP in Figure 3.53, the value of the threshold, ssthresh, is set as ssthresh=cwnd/2 in several places and ssthresh value is referred to as being set to half the window size when a loss event occurred. Must the rate at which the sender is sending when the loss event occurred be approximately equal to cwnd segments per RTT? Explain your answer. If your answer is no, can you suggest a different manner in which ssthresh should be set?

Yes the rate at which the sender is sending when the loss event occurred must be approximately equal to cwnd segments per RTT. The value of cwnd is set to 1MSS and the value of ssthresh is set to half the value of cwnd when the loss event occurred it must be approximately equal.

P40. Consider Figure 3.58. Assuming TCP Reno is the protocol experiencing the behavior shown above, answer the following questions. In all cases, you should provide a short discussion justifying your answer.

a. Identify the intervals of time when TCP slow start is operating.
    - From 0-6 and 23-26 the TCP slow start is operating
b. Identify the intervals of time when TCP congestion avoidance is operating.
    - Congestion avoidance takes place from 6-16 and 17-22
c. After the 16th transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?
    - Segment loss is detected by a triple duplicate ACK since it does not drop down to 1 which would be the case for a timeout
d. After the 22nd transmission round, is the segment loss detected by a triple duplicate ACK or by a timeout?
    - Segment loss is detected by a timeout since it drops down to 1
e. What is the initial value of ssthresh at the first transmission round?
    - The initial value of ssthresh is set to 32 since that's when the slow start stops and the congestion avoidance begins
f. What is the value of ssthresh at the 18th transmission round?
    - The value of ssthresh at the 18th transmission round is set to 24 since that's after the triple duplicate ACK and congestion avoidance begins again
g. What is the value of ssthresh at the 24th transmission round?
    - The value of ssthresh at the 24th transmission round would be 12, since the value of ssthresh is set to half the size of the cwnd when the segment loss is detected
h. During what transmission round is the 70th segment sent?
    - The 70th segment is sent during the 7th transmission round. They packets are grouped together by the different transmission rounds below. Each transmission round has the number of segments it sends with it.
    - T1 = 1, t2=2, t3=4, t4=8, t5=16, t6=31, t7=62, t8=124
i. Assuming a packet loss is detected after the 26th round by the receipt of a triple duplicate ACK, what will be the values of the congestion window size and of sshtresh?

- The congestion window would be set to half of what it was when the loss was detected. Therefore it would go from 8 -> 4.

j. Suppose TCP Tahoe is used and assume that triple duplicate ACKs are received at the 16th round. What are the ssthresh and the congestion window size at the 19th round?
   - The cwnd would drop to one and then increase to 4

k. Again suppose TCP Tahoe is used, and there is a timeout event at 22nd round. How many packets have been sent out from 17th round till 22nd round, inclusive?
   - 17th: 1, 18th: 2, 19th: 4, 20th: 8, 21st: 16, 22nd: 24
   - Total: 55 total packets are sent

P43. Host A is sending an enormous file to Host B over a TCP connection. Over this connection there is never any packet loss and the timers never expire. Denote the transmission rate of the link connecting Host A to the Internet by R bps. Suppose that the process in Host A is capable of sending data into its TCP socket at a rate S bps, where S = 10*R. Further suppose that the TCP receive buffers is large enough to hold the entire file, and the send buffer can hold only one percent of the file. What would prevent the process in Host A from continuously passing data to its TCP socket at rate S bps? TCP flow control? TCP congestion control? Or something else? Elaborate.

Since the entire file can be held in the receiver's buffer, there is no danger of the possibility of overflowing. Since the timers never expire, there is no loss and TCP congestion control will not throttle the sender. However, the process in host A will not continuously pass data to the socket because the send buffer will fill up quickly. Once the send buffer is full, the process will pass data at R << S

P55. In this problem we investigate whether either UDP or TCP provides a degree of end-point authentication.

a. Consider a server that receives a request within a UDP packet and responds to that request within a UDP packet. If a client with IP address X spoofs its address with address Y, where will the server send its response?
   - The server will send its response to address Y since it has a matching IP address.

b. Suppose a server receives a SYN with IP source address Y, and after responding with a SYNACK, receives an ACK with IP source address Y with the correct acknowledgement number. Assuming the server chooses a random initial sequence number and there is no "man-in-the-middle", can the server be certain that the client is indeed at Y (and not at some other address X that is spoofing Y)?
   - The server can not be certain that it was sent to the valid IP address Y as opposed to the spoofed address which could be address x. The client could have retried the sequence number that corresponded with the correct acknowledgement number so it's unsure where it will be sent.