15-441 Question Set #3

1. A classic pedagogical problem in computer science is the *Two Armies Problem*. The story goes like this.

   There are two opposing armies. One army, we'll call it the Red Army, is encamped in a mountain valley. The other army, we'll call it the Blue Army, is divided into two companies, each of which is encamped at a different location in the mountains.

   If the two companies of the Blue Army attack the Red Army at exactly the same time, they will reach a rapid victory. But, if they attack at different times, the Red army will destroy each. Furthermore, if the Blue Army doesn't attack, they will starve and/or freeze to death in the mountains.

   So, what is needed is a protocol by which the two companies of the Blue Army can reach mutual agreement about the time to attack. This needs to be done with a messenger(s) on horseback (how else would one communicate a message in the days before the Internet?)

   Please keep in mind that only if both parties know the correct time to attack, and are absolutely confident that the other party knows the correct time to attack, will the Blue Army triumph. If they never agree, they will freeze and/or starve. If either side is not confident that the other side knows the right time to attack, even if it does, it won't attack. Depending on who knew what, this could lead to the starvation of the two companies, or the defeat of one and the starvation of the other - neither is a good option for the Blue Army.

   If possible, design a protocol that will enable the Blue Army to defeat the Red Army. You may assume that, if the messenger communicates a message, it will not be corrupted. But, please keep in mind, that messengers can get delayed, lost, or killed. You may assume that there are as many messengers as necessary and that each can make as many trips as necessary. You should not concern yourself with the privacy of the messages.

   If you believe that it is not possible to design such a protocol, please explain why.

2. In class, we discussed different types of acknowledgements:
   a. simple acknowledgement

   b. cumulative acknowledgement

   c. selective acknowledgement

   Please discuss the costs and benefits of each approach. Your answer should include a best-case and worst-case situation for each policy and explain under what circumstances each should be selected above the others.

3. Consider a sliding window protocol, such as the one we discussed in class. What effect does the window size have upon throughput? Please derive a formula that expresses the maximum *throughput* (bits/second) of the connection between two systems as a function of the *bit rate* of the channel, the one-way *latency* of the channel (seconds), the *window size* (number of frames), and the *frame size* (bits).

4. Again consider the window size of a sliding window protocol. What factor(s) influence the appropriate window size? What happens if the window is too small? Too large?

5. I suggested in class that TCP will generate a duplicate ACK if a segment being lost or reordered. Please draw a picture that demonstrates each of these situations.

6. Does it make sense to use slow start when initiating communication with another host on the same network (LAN)? Why or why not?

7. We talked a lot about congestion control. What causes congestion?

8. Please consider the plot below. It represents the congestion window size (vertical) against the transmission round/time (horizontal). Please identify each of the following, if present:

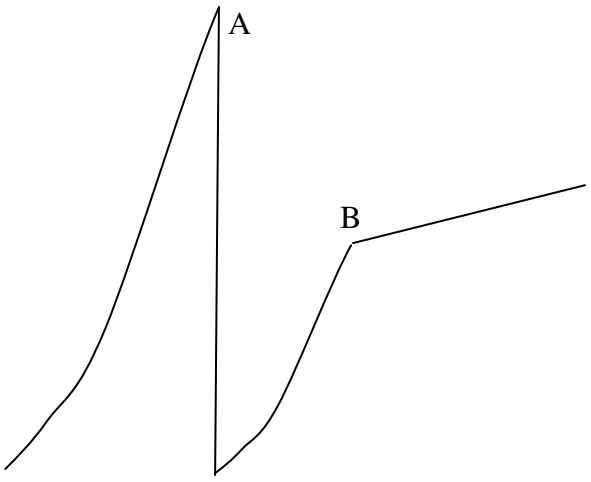
    a. The period(s) of slow-start.

    b. The period(s) of congestion control.

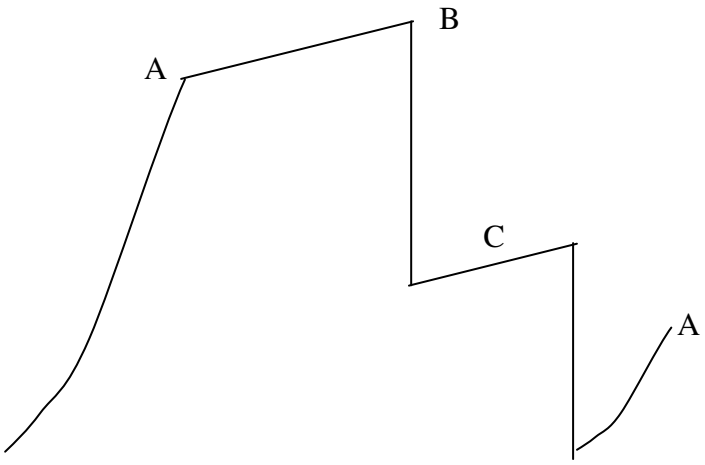
    c. The period(s) of fast recovery

    d. The point(s) at which ssthresh is reached

    e. What happened at each of points A and B? Be specific: What are the two possibilities?



9. Please answer question #8 again, this time, for the plot below:



10. Please consider the plot from question 8, PLOT 8, and the plot from question 9, PLOT 9. Please compare Point A of PLOT 8 and Point B of PLOT 9. In each case the window size drops. But, the two cases are different.

    a. Are the plot necessarily from the same version of TCP? If so, how do you know? If not, what might be the difference in the protocol's behavior? (Alt: What is a possible difference in the version of TCP and how do you know?)

    b. Does Point B of PLOT 9 represent a timeout or multiple duplicate ACKs? How do you know?

11. Given the existing fabric of the standard internet protocols, how do routers communicate to senders that they have reached, or will soon reach, overburdened?