Fall 2024, CS 3953: Computer Networks

Homework 3

Problem 1 (10 points)

Suppose a process in Host C has a UDP socket with port number 6789. Suppose both Host A and Host B each send a UDP segment to Host C with destination port number 6789. Will both of these segments be directed to the same socket at Host C? If so, how will the process at Host C know that these two segments originated from two different hosts?

Problem 2 (15 points)

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?

Problem 3 (5 points)

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

Problem 4 (10 points)

Considering the TCP 32-bit sequence number. How long will the sequence number will be used up when (Note: For TCP, each byte has a unique sequence number.)

1. The line is 56-kbps.
2. The line is 10-Mbps.
3. The line is 4 Gbps.
4. Suppose a packet can stay in Internet for 0.5 minute at most. Do you think 32-bit sequence number is enough for 1Gbps network? If not enough, how TCP deal with this problem?

Problem 5 (10 points)

We have said that an application may choose UDP for a transport protocol because UDP offers finer application control (than TCP) of what data is sent in a segment and when.

1. Why does an application have more control of what data is sent in a segment?
2. Why does an application have more control on when the segment is sent?

Problem 6 (20 points)

Compare GBN and SR. Assume that the timeout values for both protocols are sufficiently long such that 6 consecutive data segments and their corresponding ACKs can be received (if not lost in the channel) by the receiving host (Host B) and the sending host (Host A) respectively. Suppose Host A sends 6 data segments to Host B, and the 3rd segment (sent from A) is lost. In the end, all 6 data

segments have been correctly received by Host B.
How many segments has Host A sent in total and how many ACKs has Host B sent in total? What are their sequence numbers? Answer this question for both protocols.

Problem 7 (20 points)
Consider Figure 1. 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.
1. Identify the intervals of time when TCP slow start is operating.
2. After the 10th transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?
3. After the 14th transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?
4. What is the initial value of ssthresh at the first transmission round?
5. What is the value of ssthresh at the 12th transmission round?
6. What is the value of ssthresh at the 15th transmission round?
7. During what transmission round is the 60th segment sent?
8. Assuming a packet loss is detected after the 29th round by the receipt of a triple duplicate ACK, what will be the values of the congestion window size and of ssthresh?
9. Suppose TCP Tahoe is used (instead of TCP Reno), and assume that triple duplicate ACKs are received at the 10th round. What are the ssthresh and the congestion window size at the 11th round?

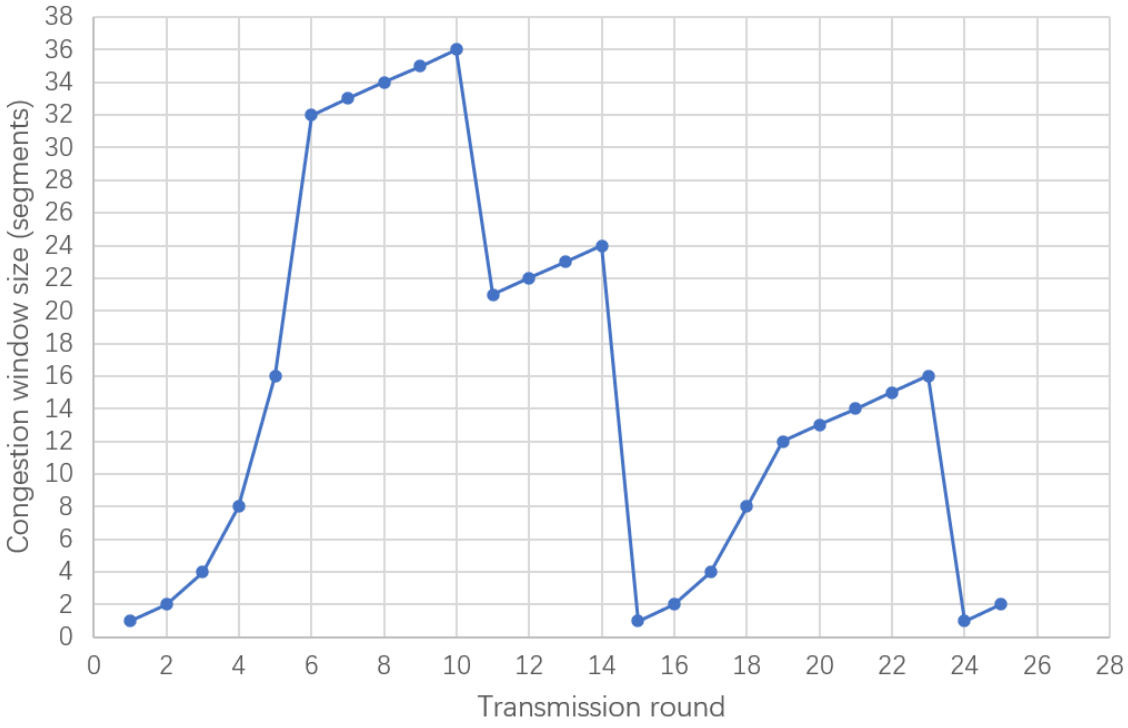

Figure 1: TCP window size as a function of time

Problem 8 (10 points)
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 = 5 R. Further suppose that the TCP receive buffer 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.