CIS 432/532 Introduction to Computer Networks Name:

Prof. Daniel Zappala

Midterm

7 November 2002

Instructions

Show your work for all problems. Be direct and convincing with your written answers.

Section A: Questions (10 points each)

1. Why is it important for a DNS root name server to implement recursive rather than iterative queries?

   Using an iterative query is expensive because the server has to query the authoritative name server, wait for the reply, and then forward this reply back to the original name server that requested the lookup. This requires the root name server to keep state about each of its queries and maintain a timer in case a name server does not respond. Using recursive queries offloads this burden to the originating name server; the root name server just replies with the name of the authoritative name server.

2. Why is persistent HTTP, without pipelining, faster than non-persistent HTTP? Why is HTTP with pipelining faster than HTTP without pipelining? Assume in both cases that the objects being transferred are small.

   Persistent HTTP without pipelining is faster than non-persistent HTTP because the browser eliminates the handshaking delay required to setup additional connections when more than one object comes from the same server. Thus, instead of 2 RTT of propagation delay for each request (to open the connection and send the request), the browser waits only 1 RTT per request.

   Using pipelining can be even faster because all of the requests for objects (after the first one) can be sent together, incurring one total RTT for all requests.

   In both cases, this delay savings is significant when the objects being transferred are small, since propagation delay dominates transmission delay.

3. Give an example of when it would be faster to use web caching rather than a CDN. Why would a content provider pay a lot of money to set up a CDN when web caching (which is paid for by the users and not the content provider) also places content near the users?

   It would be faster to use web caching rather than a CDN if the web cache is closer to the user than the closest CDN server. For example, a web cache on the user's subnet will have a shorter propagation delay and equal or greater bandwidth available than a distant CDN server.

   Despite this example, and the greater cost of a CDN, content providers still pay money to a CDN provider because the CDN will guarantee their content is available to users at various places in the network. A web cache cannot guarantee that the provider's content will be in the cache – it only holds what other users have recently requested. In addition, some users may not have a web cache available to them.

4. In what two ways can a TCP sender detect loss? How could the network give an explicit signal that congestion is occurring?

   A TCP sender can detect a packet is lost either by a timeout when an ACK for that packet does not arrive or by receiving three duplicate ACKs for the previous packet.

   The network could give an explicit congestion signal by setting a bit in a packet indicating congestion is

occurring and then having the destination "echo" that bit to the sender (as is done with DECBIT). Another possibility is to send a message to the source (e.g. the ICMP Source Quench message).

5. Explain how TCP slow start works when a connection is first started. In your explanation, describe how TCP uses the Threshold and CongWin variables. Why is this considered "slow?"

   TCP slow start begins by setting the CongWin to MSS and then sending the first segment. For each ACK that arrives, the CongWin is increased by MSS until the initial Threshold is reached. At this point, slow start is finished and congestion avoidance begins.

   This is considered "slow" because prior to slow start TCP would use flow control to determine the receiver's window size and then immediately start sending a full window of segments.

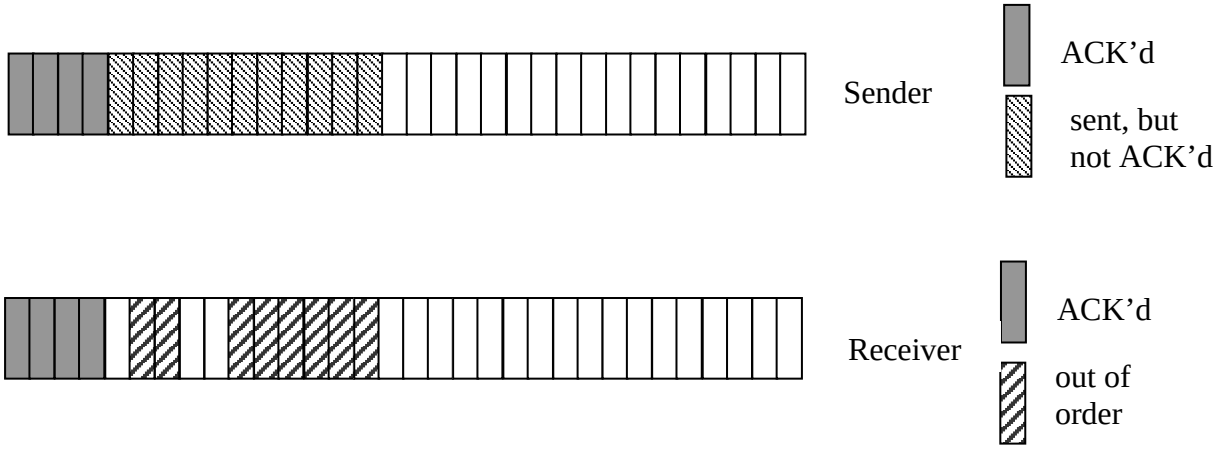
6. What is the difference between a cumulative acknowledgement and a selective acknowledgement? Give a simple example of when it would be preferable to use a selective acknowledgement.

   A cumulative acknowledgement states that a given segment and all previous segments have been received. A selective acknowledgement only states that the given segment has been received.

   A simple example is when the first packet in a window is lost, but the others are received OK. Using selective acknowledgement, the sender would know that only one packet was lost, whereas with cumulative acknowledgement the sender does not know how many are lost. If only one packet is lost, then the sender can send nearly another window of new packets while it is retransmitting the lost packet.

Section B: Problems (20 points each)

1. The diagrams below show the buffers of a TCP sender and TCP receiver. Each buffer contains a 1000 byte packet. The sequence number of the first segment in both buffers is 0.



   a. Assuming the sender has sent as many packets as it can, what is its window size? Which segment starts the base of the window? What is the next sequence number that the sender can transmit?

   The window size is 11000 since 11 segments are outstanding. The base of the window is segment 4000. The next sequence number that the sender can transmit is 15000.

   b. Assume the receiver sends an acknowledgement for every segment it receives. What acknowledgements has it sent to the sender? Will this cause the sender to retransmit a segment? Why or why not?

   The receiver has transmitted an ACK for segments 1000, 2000, 3000, and 4000, plus 8 additional ACKs for segment 4000 (the segment it is expecting to receive) because of the 8 out-of-order packets it gets.

This WILL cause the sender to retransmit a segment because TCP will retransmit a segment after 3 duplicate ACKs.

c. The sender retransmits the first segment that the receiver is missing. What sequence number will the receiver send in its acknowledgement? How many additional segments can the application now read? When the sender gets this acknowledgement, how will it change the base of its window? How many additional packets may it now send?

The sequence number the receiver will send is 7000, because this is the next segment it will expect to receive (it has received everything prior to this number). The application can now read 3 additional segments (4000 – 6000 are now buffered and in order). When the sender gets its acknowledgement, it will increase its window base to 7000 and it can now send 3 additional segments.

2. Two machines, A and B, are connected by a router. The link from A to the router is 10 Mbps and has a propagation delay of 10 ms. The link from B to the router is 100 Mbps and has a propagation delay of 30 ms. The router uses store-and-forward switching and imposes an average queuing delay of 20 ms. Packets sent between the two machines contain 1 KB of data.

   a. How long does it take to transmit a 10 KB file from machine A to machine B?

   We can find the total delay by considering the last packet sent. This packet has to wait for the previous 9 packets to be transmitted on the link, propagate to the router, wait for the queuing delay, and then be transmitted on the last link and propagate to the last machine. So the total delay is:

   delay = 10 KB / 10 Mbps + 10 ms + 20 ms + 1 KB / 100 Mbps + 30 ms
   delay = 8.2 ms + 10 ms + 20 ms + 0.082 ms + 30 ms
   delay ~ 68 ms

   (1) We do not account for 10 KB of transmission delay on the last link, because by the time to transmit the other 9 packets overlaps with the time that the last packet waits in the queue.

   (2) We do not account for 10 packets worth of queuing delay because of pipelining. The last packet joins the queue while the other packets are already in the queue, waiting their turn. The queuing delay for all of the packets overlaps. This is because the queuing delay in this example is so large.

   b. How large does a file need to be before transmission delay begins to be larger than propagation delay? How does this change if both links are 100 Mbps?

   Of the total delay, 40 ms is for propagation delay. So:

   L/10 Mbps + 1 KB/100 Mbps > 40 ms
   L/10 Mbps > 40 ms
   L > 400,000 bits
   L > 48.8 KB

   If the first link is made faster, then:

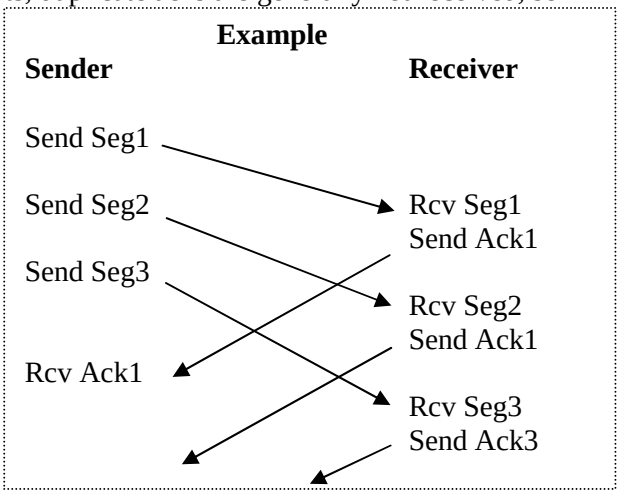
   L/100 Mbps + 1/100 Mbps > 40 ms
   L/100 Mbps > 40 ms
   L > 488 KB

Section C: Graduate Students Only: Problems (20 points each)

1. The STCP protocol is a version of TCP that uses selective acknowledgements rather than cumulative acknowledgements. With selective acknowledgements, duplicate acks are generally not received, so packet loss can only be detected by a timeout.

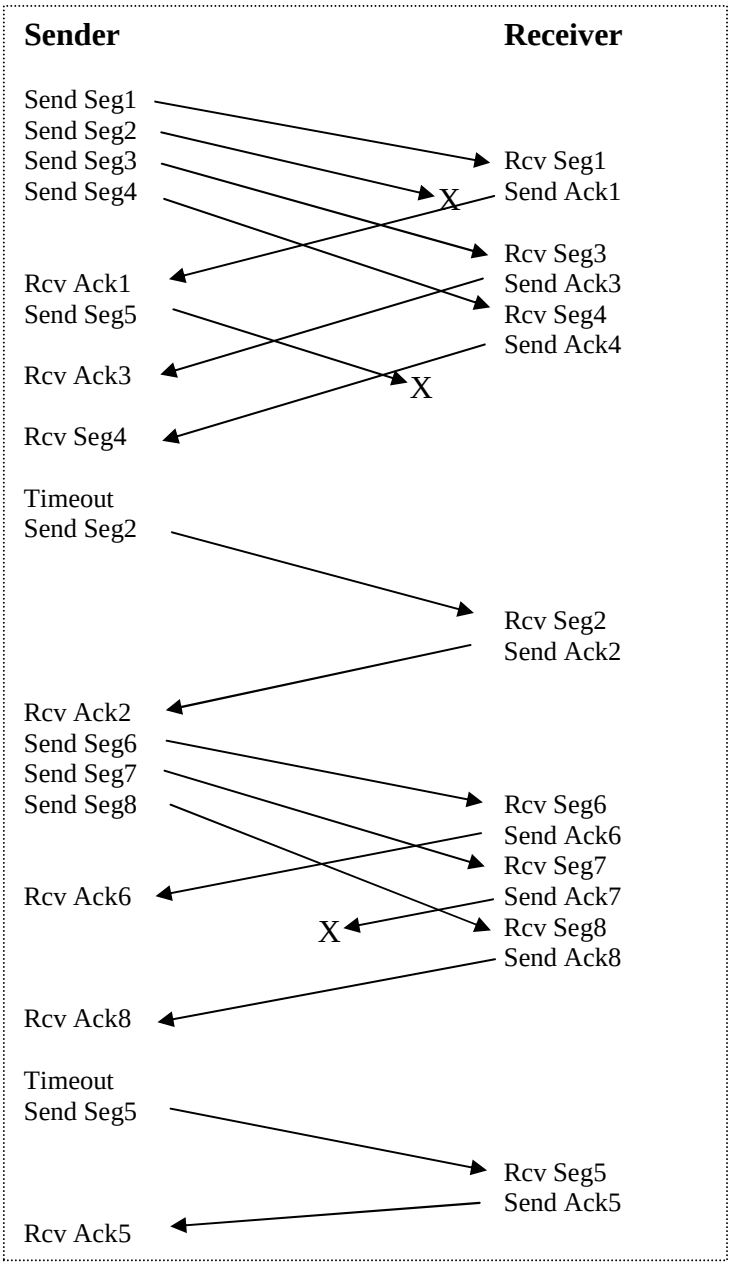
a. Draw a diagram of the interaction between an STCP sender and receiver, using the timeline format shown at right. Assume that STCP uses a static window size of 4 segments. The entire window can be sent before any acknowledgments are received.

A total of 8 segments need to be sent. Segment #2 and segment #5 are lost the first time they are sent. The first Ack for segment #7 is lost. No other segments are lost.



In the solution shown at right, one more timeout is needed for Seg 7, and will look just like the timeout for Seg 5.

In the solution I show here I use a sliding window, as discussed in the book. This means that when the Acks for segments 3 and 4 are received, new segments cannot be sent. The base of the window is still 2, and so the maximum segment that can be sent is 2 + 4 – 1 = 5. Once the Ack for segment 2 is received, then the window base slides to 5, so segments 6, 7, and 8 can now be sent. In your solution, you may have used a non-window version of selective acks that operates differently. Since I did not specify your implementation, I did not deduct any points for this.



b. Invent a rule for when STCP should send a negative acknowledgment. Give an example of how this would help to improve the protocol.

STCP should send a NACK whenever it receives 2 segments that are out of order. In the example shown above, this would mean that the receiver sends an ack for Segment 2 once it receives Segment 4. This would allow the sender to retransmit Segment 2 before its timer expires. Once the sender gets the ack for Segment 2 it can send segments 6-8, which will result in a NACK for Segment 5. In the end, the sender will retransmit sooner (before any timer expires) and as a result the data would be transferred more quickly.