Midterm Exam
CMPSCI 453: Computer Networks
Fall 2011
Prof. Jim Kurose

Instructions:

- There are 4 questions on this exam.
- **Please use two exam blue books** – answer questions 1, 2 in one book, and the remaining two questions in the second blue book.
- Put your name and student number on the exam books NOW!
- The exam is closed book.
- **You have 80 minutes** to complete the exam. **Be a smart exam taker** - if you get stuck on one problem go on to another problem. Also, don't waste your time giving irrelevant (or not requested) details.
- The total number of points for each question is given in parenthesis. There are 100 points total. An approximate amount of time that would be reasonable to spend on each question is also given; if you follow the suggested time guidelines, you should finish with 5 minutes to spare. The exam is 80 minutes long.
- Show all your work. Partial credit is possible for an answer, but only if you show the intermediate steps in obtaining the answer.
- Good luck.

Question 1: ``Quickies'' (28 points (4 each), 20 minutes)

Answer each of the following questions *briefly*, i.e., in at most *a few sentences*.

a) We have learned that the Internet provides "best effort service." What is meant by a "best effort" service model? Give an example of another service model that a network might offer. *Answer: Best effort service means that the network layer doesn't guarantee to deliver a datagram from source to destination. It will try but makes no guarantees. ATM networks, on the other hand do provide guarantees, for example, a constant bit rate channel, with guaranteed data delivery.*

b) Suppose that Alice wants to send an email message to Bob. This will involve four entities: Alice's mail client (for email composition and sending), Alice's outgoing mail server, Bob's incoming mail server, and Bob's mail client (for email retrieval and viewing). Between which of these four entities does the SMTP protocol operate? What about the IMAP protocol? *Answer: SMTP runs between Alices mail client and her server, and also (separately) between her server and Bob's server. IMAP runs between Bob's server and his mail client to retrieve messages from Bob's server.*

c) What is the purpose of the HTTP "COOKIE:" field? Are the values in the HTTP message's cookie field stored at the client or server or both? Explain briefly. *Answer: a cookie can be used to maintain state between HTTP transactions. For example, if a user uses the same cookie in its HTTP requests whenever dealing with a given e-commerce site, that site can track all interactions with the user associated with that cookie. Both the client and server store the cookie value.*

d) Suppose that we want to change the IP address of gaia.cs.umass.edu from 128.119.40.186 to 128.119.40.187 and change this mapping in the DNS authoritative name server for gaia.cs.umass.edu. Once this mapping is changed in the authoritative name server, will all future references (generated anywhere in the Internet) to gaia.cs.umass.edu then be sent to 128.119.40.187? Explain briefly (in two or three sentences). *Answer: Local DNS caches throughout the Internet will not time out the old mapping of gaia.cs.umass.edu to 128.119.40.186 until the valid interval originally associated with that mapping times out. Until that happen, local DNS caches will not query into the system for gaia.cs.umass.edu and hence would not learn the new mapping.*

e) We saw that TCP and UDP provide two very different service models. Suppose that an application wants all of the functionality provided by UDP but only some of the functionality provided by TCP (e.g., the application wants reliable message transfer and flow control, but not congestion control). How would an application get this different service in today's Internet? *Answer: The application would use UDP sockets and implement the desired additional functionality (e.g., reliability and flow control) in the application itself).*

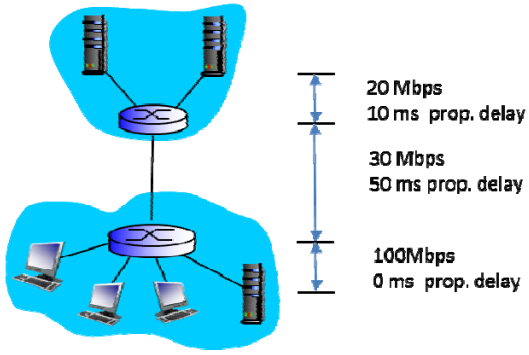
f) Consider a server-side socket that is used to communicate from server to client. In the case of a TCP socket, can data being read from the server-side socket have been sent by more than one client? Explain briefly. In the case of a UDP socket, can data being read from the server-side socket have been sent by more than one client? Explain briefly. *Answer: With TCP, the client-to-server socket is created by the return from the accept() call at the server, binding the client that was accepted to the server via the*

newly created socket. Thus only one client is associated with the TCP socket. With UDP, any client can send to the (same) UDP socket on the server.

g) What is meant by the term "encapsulation"? *Answer: This means that a protocol executing at a given layer in the protocol stack takes a message for the upper layer, wraps it in a new packets with new header (and possible trailer) fields.*

Question 2: Delays and Throughput (18 points, 15 minutes)

Consider the scenario in the figure to the right, in which (from the bottom up) three hosts and a local logging server (that stores information that is sent to it) are connected to a router and to each other by a 100 Mbps link, with an near-zero ms propagation delay. That router in turn is connected to another router over a 30 Mbps link with a 50 ms propagation delay, and that latter router is connected to two remote logging servers, each over a 20 Mbps link with a10 ms propagation delay.



a) Suppose a host sends a logging message directly to one of the *remote* logging servers. The logging message is 10K bits long. What is the end-to-end delay from when the logging message is first transmitted by the host to when it is received at the remote server? Assume that the request goes directly to the server, that there are no queueing delays, and that node (router) packet-processing delays are also zero. *Answer: given the 10K bit packet, it takes .0005 secs to send this packet over a 20 Mbps link. 0.000333 secs to send over a 30 Mbps link, and .0001 secs over the 100 Mbps link. The total transmission time end-to-end is this .0009333 secs. The total propagation delay is 60 ms. Therefore the total end-end delay is .0609333 secs.*

b) Assume that each of the three hosts generate logging messages at the same rate; each host is equally like to send a logging message to either of the two remote servers. No traffic is directed to the local logging server. What is the maximum rate at which the clients can send logging messages to the remote servers? *Answer: the link between routers is the bottleneck link, allowing 30 Mbps to be delivered to the two servers combined, or 15 Mbps to be delivered to each server. Since each message is 10K bits, this is 1.5K logging messages per second.*

c) Now assume that the local logging server is ON and only one host is active (generating) logging messages and that host is only sending messages to *one* of the remote logging servers. Suppose that 50% of the logging messages are directed locally and the other 50% directed to this remote server. What is the maximum rate at which this host can generate and send logging messages (both local and remote combined, given there is a 50/50 ratio of local/remote transmissions) in this scenario? *Answer: The maximum rate at which the host can generate remote logging messages is 20 Mbps or 2K logging messages per second. Local messages can be generated that the same rate, so the overall rate is 40 Mbps or 4K logging messages per second.*

Question 3: A data-pull protocol (35 points, 30 minutes)

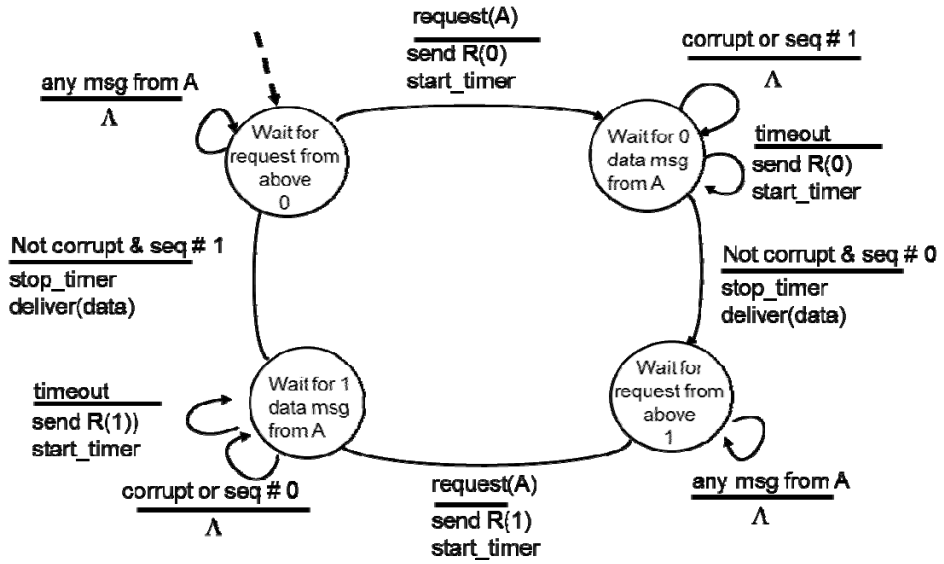
Consider the following scenario. At a requestor node, R, a high level process generates requests for data items that are stored at a remote node A that has an infinite supply of such data items (and so A always has a data item to send). Nodes R and A are connected to by a channel that can *lose* or *corrupt* messages, but *will not reorder* messages.

Design a requester protocol that operates at R and A, operating as follows. At R, requests for data items are received from above via a call **`request(A)`** that requests that the next item be retrieved from A. The request protocol at R eventually delivers a requested data item to the layer above, satisfying such a request, by calling **`deliver_data(data)`**, where **`data`** is a data item received from A . You may assume that a call from above will not be made until the previous call from above requesting a data item has been satisfied via a **`deliver_data()`** call. R sends request messages to A; the fields in the request message are part of your protocol's design. A send data messages containing a requested data item to R; the fields in the data message are part of your protocol's design.

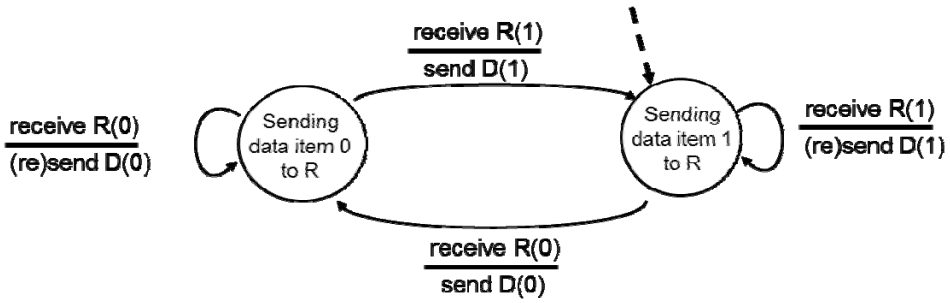
You should design a protocol with which R requests, and A sends, data items, and that R delivers (to the caller above) exactly one copy of the data item requested. You should make sure that your protocol handles corrupted or lost messages on the R-to-A and A-to-R channel.

a) [5 points] Describe the format of messages being sent from R to A, and from A to R. *Answer: The requestor sends an R message to the A, each R message carries a sequence number, which is incremented (modulo 2) after each call from above. A sends a data message to A, taking the sequence number from the request message. When a request for a new piece of data arrives (indicated by a change in the request sequence number), A begins to send the next data item. Note that we do not need ACKs from R to A, since R will only move on to request the next in-sequence data item after it has received the data correctly (as shown in the sender FSM below). Thus, the next request acts as an implicit ACK. We'll also need a checksum field to detect errors.*

b) [20 points] Give a FSM description for the data requesting protocol at R. *You do not need to specify the FSM at the level of detail in the text or notes.* But you should make clear what action is being taken an events occurs, and any necessary values/parameters associated with the event or action taken (e.g., message transmission or reception). *Answer: because messages can be lost, we'll need a timer, and because there can be lost or repeated request messages, we'll need to put a sequence number on each request message, just as we did in rdt 3.0. To see why, note that A needs to know whether a received request is a request for new piece of data, or a retransmitted request for the previous piece of data. Here is the FSM for R:*

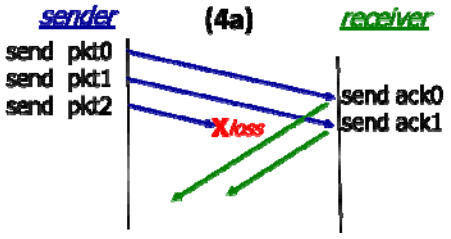


c) [10 points] Give a FSM description for the data requesting protocol at A. *Answer: Note that R will be sending only a series of R(0)s followed by one or more R(1)s, followed by one or more (R(0)s, etc… This makes A pretty simple:*

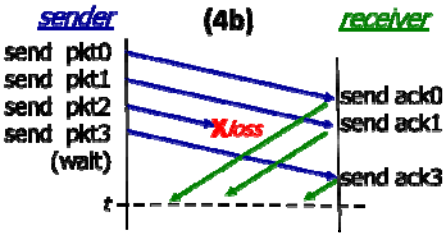


Problem 4. Sliding Window Protocols (19 points, 10 minutes)

a) [3 points] Consider the sliding window protocol in Figure (4a) to the right. Does this figure indicate that Go-Back-N is being used, Selective Repeat is being used, or there is not enough information to tell? Explain your answer briefly. *Answer: there is not enough information to tell, since both GBN and SR will individually ACK each of the first two messages as they are received correction.*

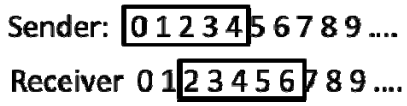


b) [3 points] Consider the sliding window protocol Figure (4b) to the right. Does this figure indicate that Go-Back-N is being used, Selective Repeat is being used, or there is not enough information to tell? Explain your answer briefly. *Answer: This*



must be the SR protocol since pkt 3 is acked even though pkt 2 was lost. GBN uses cumulative ACKs and so would not generate an ACK 3 if pkt 2 was missing.

c) [3 points] Consider Figure (4b) again. Suppose the sender and receiver windows are of size N = 4 and suppose the sequence number space goes from 0 to 15. Show the position of the sender and receiver windows over this sequence number space at time *t* (the horizontal dashed line). *Answer:*



d) [5 points] Give a list of all possible future events at the sender resulting from the ACKs currently propagating from receiver to sender at time *t*. For each of these events, indicate the action take at the sender (only).
Answer:

- *If the next event is ACK0 received, then the sender will advance the window and send pkt4*
- *If the next event is ACK1 received (ACK 0 is lost), then the sender will note that 1 has been ACKed but will not advance the window and will not send anything.*
- *If the next event is ACK2 received (ACK 0 and ACK 1 are lost), then the sender will note that 2 has been ACKed but will not advance the window and will not send anything.*
- *If ACK0, ACK1, and ACK3 are lost the next event will be a timeout, and since no ACKs have been received the sender will resend pkt0, pkt1, pkt 2 and pkt3.*

e) [5 points] Suppose that it take 1 ms to send a packet, with a 10 ms one-way propagation delay between the sender and receiver. The sliding windows size is again $N = 4$. What is the channel/link utilization? *Answer: the utilization is 4/(1+20) or 0.19.*