NAME:

Login name:

Computer Science 461
Midterm Exam
March 10, 2010
3:00-4:20pm

This test has five (5) questions. Put your name on ***every page***, and write out and sign the Honor Code pledge before turning in the test.

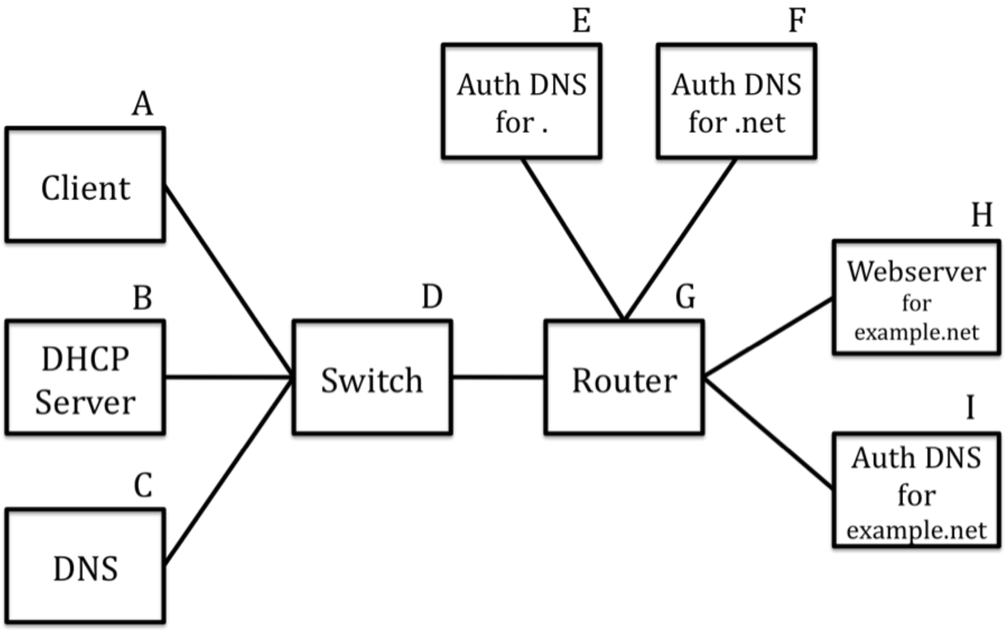
Please look through all of the questions at the beginning to help in pacing yourself for the exam. The exam has 100 points and lasts for 80 minutes, so the number of minutes spent per question should be *less* than its point value. You should spend no more than 15 minutes per question. Show your work for all problems. Partial credit will often be given.

"I pledge my honor that I have not violated the Honor Code during this examination."

Question	Score
1	/ 19
2	/ 19
3	/ 16
4	/ 29
5	/ 17
Total	/ 100

QUESTION 1 : Protocols and Addressing (19 POINTS)

In the topology shown below, a client A (on the left side) joins a network and wants to download the webpage for http://example.net/ from the webserver H (on the right side). All wires/links are Ethernet segments. In this question, you will be asked what packets are on the wire (and the addresses in those packets' headers) for this to occur. When asked to provide the addresses of machines, you can refer to them abstractly: IP-H is the IP address of the webserver H, MAC-H is the MAC address of H. If a packet is being sent to a broadcast address, use "BR" as the destination (e.g., IP-BR or MAC-BR).



Assume the following:

- All the machines, except the client, already have their ARP caches full.
- DNS resolver C has an empty DNS cache.
- http://example.net/index.html fits in a single packet.
- Client just joined the network and so all of its caches are empty (DNS, ARP, DHCP)

When you are asked to write a packet with its headers, you should include both network and link-layer addresses. So a DNS query from client A to server C and the response should look like the following. Note that other packets may be involved with a full DNS lookup that we are not showing in this example.

Order	Link Src	Link Dst	Network Src	Network Dst	Protocol Description
1	MAC-A	MAC-C	IP-A	IP-C	DNS query
2	MAC-C	MAC-A	IP-C	IP-A	DNS resp.

In the following questions, you may not need to use all the lines we provide.

1. [4 points] As it just joined the network, the first thing the client needs to do is get an IP address and other related network state. In the next lines, write the packets on the network involved in that step, including their header information.

Order	Link Src	Link Dst	Network Src	Network Dst	Protocol Description
1	MAC-A	MAC-BR	--	--	DHCP Discover
2	MAC-B	MAC-A	--	--	DHCP Offer
3	MAC-A	MAC-BR	--	--	DHCP Request
4	MAC-B	MAC-A	--	--	DHCP Acknowledgment
5					
6					

2. [10 points] Now that the client has an IP address for itself, it wants to resolve the IP address of example.com, which will be IP_H. Write down all the packets involved in this phase. Remember that the client just joined the network.

If a packet's headers change during transmission (i.e., the IP or MAC address get changed), you should write the packet on two separate lines, corresponding to the two set of headers it sees during transmission. For the description, write the most specific protocol, i.e., "DNS query", not "UDP".

Order	Link Src	Link Dst	Network Src	Network Dst	Protocol Description
1	MAC-A	MAC-BR			ARP Request
2	MAC-C	MAC-A			ARP Reply
3	MAC-A	MAC-C	IP-A	IP-C	DNS Request
4	MAC-C	MAC-G	IP-C	IP-E	DNS Request (to .)
5	MAC-G	MAC-E	IP-C	IP-E	DNS Request
6	MAC-E	MAC-G	IP-E	IP-C	DNS Reply
7	MAC-G	MAC-C	IP-E	IP-C	DNS Reply
8	MAC-C	MAC-G	IP- C	IP-F	DNS Request (to .net)
9	MAC-G	MAC-F	IP-C	IP-F	DNS Request
10	MAC-F	MAC-G	IP-F	IP-C	DNS Reply
11	MAC-G	MAC-C	IP-F	IP-C	DNS Reply
12	MAC-C	MAC-G	IP-C	IP-I	DNS Request (to example.net)
13	MAC-G	MAC-I	IP-C	IP-I	DNS Request
14	MAC-I	MAC-G	IP-I	IP-C	DNS Reply
15	MAC-G	MAC-C	IP-I	IP-C	DNS Reply
16	MAC-C	MAC-A	IP-C	IP-A	DNS Reply
17					

3. [5 points] Now that the client has IP_H, it wants to download the web page http://example.net/index.html. (Recall that the webpage fits in one data packet.) Write down all ***network-layer*** packets that are sent on the wire in order to complete this process. Remember to clean up any transfers.

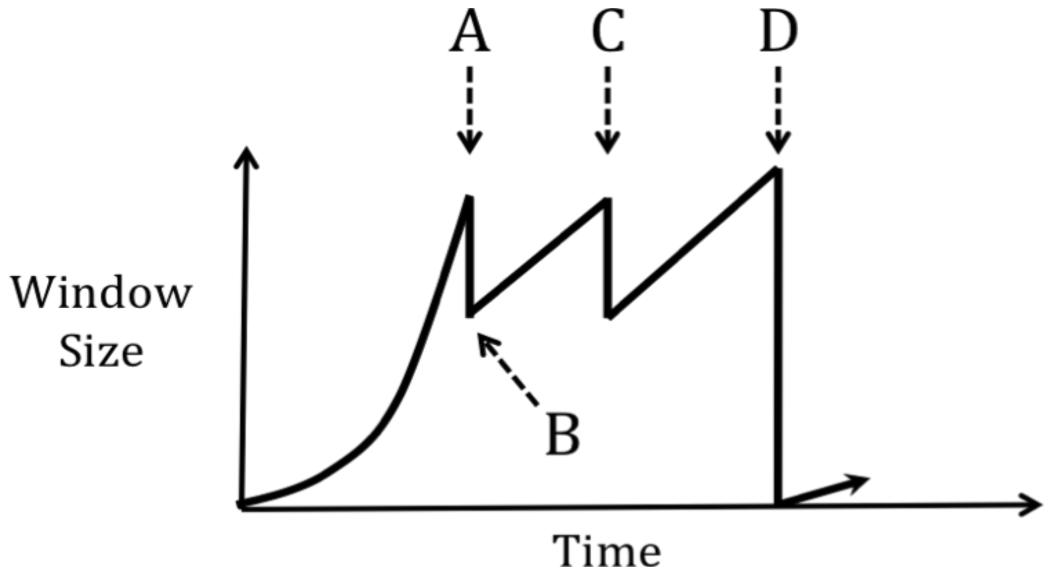
For the "protocol description", if the packet is associated with transport layer, you should write something like "TCP SYN". If it's inherently an application-level message (i.e., data sent over a TCP connection), you should write something like "HTTP request". In this example, you should assume that you are running a "basic" TCP implementation, so it's not doing any fancier optimizations like delayed ACKs or piggybacking.

Order	Network Src	Network Dst	Description
1	IP-A	IP-H	TCP SYN
2	IP-H	IP-A	TCP SYN-ACK
3	IP- A	IP-H	TCP-ACK
4	IP-A	IP-H	HTTP Request
5	IP-H	IP-A	TCP ACK
6	IP-H	IP-A	HTTP Reply
7	IP-A	IP-H	TCP ACK
8	IP-H	IP-A	TCP FIN
9	IP-A	IP-H	TCP FIN-ACK
10	IP-H	IP-A	TCP ACK
11			
12			

** Note: We also accepted 8 as IP-A to IP-H (i.e., close initialized by client), and we also accepted a 4-way handshake of FIN, ACK, FIN, ACK.

QUESTION 2 : TCP and Congestion Control (19 POINTS)

Consider the following graph of TCP throughput (**NOT DRAWN TO SCALE**), where the y-axis describes the TCP window size of the sender. We will later ask you to describe what happens on the right side of the graph as the sender continues to transmit.



1. [3 points] The window size of the TCP sender decreases at the points marked by A, C, and D on the graph, which happens when a packet belonging to the stream is lost. Name the event which occurs that causes the sender to decrease its window.

A: Triple Duplicate ACK

C: Triple Duplicate ACK

D: Timeout

2. [4 points] Assume that the network has an MSS of 1000 bytes. If point A occurs 1 second after the sender begins its transfer, and the sender has written 15,000 bytes to the network by that time, what is the round-trip-time (RTT) of the network? Assume at time 0 the sender attempts to open the connection. Also assume that the sender can "write" a full window's worth of data instantaneously, so the only latency you need to worry about is the actual propagation delay of the network.

First, calculate the number of RTTs needed. 1 RTT for client handshake (SYN → SYN-ACK). Then window size grows during slow start: 1 MSS, 2 MSS, 4 MSS, 8 MSS = 15 MSS = 15,000 bytes. Note that A occurs at the point of detecting triple duplicate ACK, and therefore the client must have seen at least some ACKs from the 8 MSS windows.

Hence a total of 5 RTTs over 1 second = 200 ms

3. [4 points] What is the sender's window size (in bytes) at point B?

After triple duplicate ACK, the sender halves its window size (part of "multiplicative decrease" of AIMD), thus leaving us at ½ cwnd of A = 4 MSS = 4,000 bytes

4. [4 points] If point C occurs 2 seconds after point B, what is the sender's window size (in bytes) at point C?

Network RTT = 200 ms, thus 2 seconds = 10 RTT. The sender's window is currently growing linearly in the "additive increase" part of AIMD, and hence increases by 1MSS every RTT.

Therefore, our prior cwnd was 4 MSS + 10 MSS = 14 MSS = 14,000 bytes

5. [4 points] Assume that the window size at point D is 16,000 bytes, at which time it drops to 1,000 bytes. How much time will it take for the sender to return to a window size of 16,000 bytes (assuming that there is no further packet loss in the network)? You can express your answer in terms of the number of RTTs, as opposed to the number of seconds.

After a timeout, the window drops to 1 MSS. It then proceeds to grow via slow-start (i.e., doubling each RTT) until it reaches ½ of the cwnd at the time it detected the timeout. Afterwards, it grows linearly (+1 MSS / RTT).

Thus, we experience slow-start until 8 MSS, which takes 3 RTT. We then increase linearly until 16 MSS, which takes an additional 8 RTT, giving us a total of 11 RTTs.

QUESTION 3: Socket Programming (16 points)

You are implementing a simple application to transfer a file from the client to the server using TCP.

1. [5 points] In the following, write the basic system calls you need to use in order to transfer the file. Assume that the client already has the IP address of the server. You don't need to show arguments, i.e., name resolution could have just been written as `gethostbyname` (not `struct hostent * gethostbyname(const char *name)`) and you don't need to show any other programmatic logic. Assume that the file can be written/read in a single call to `send/recv.`

In the below table, provide a set of system calls in an appropriate order. Each line should have a single call at the client or server, not both. Some calls at the client and server can happen concurrently; just list them in a reasonable order (i.e., the write at the client should occur before the read at the server). Note: You might not need to use all the rows supplied. Remember to cleanup after you transmit the file.

	Client	***Server***
1		***socket***
2		***bind***
3		***listen***
4		***accept***
5	***socket***	
6	***bind***	
7	***connect***	
8	***send***	
9		***recv***
10	***close***	
11		***close***
12		
13		
14		
15		

*** Note there are multiple different acceptable answers.

2. [10 points] Now consider the pseudocode of the client's `send` call in a bit more depth. Write some pseudocode (it doesn't need to compile, but should look C-like), how you would transmit a 100,000B file. In this code snippet, you *should* use arguments and return values to express the logic flow.

Your code should only be a few lines long; we're only interested in the logic immediately surrounding your use of the `send` system call.

```
// buf points to a buffer of size len
//   that you want to transmit in its entirety
// fd is an open file descriptor connected to server
// RETURN 0 ON SUCCESS, -1 ON FAILURE

int sendall (int fd, char *buf, int len) {

     int sent = 0;

     while (sent < len) {

        int bytes_sent = send (fd, buf+sent, len-sent);

        if (bytes_sent < 0)
             return -1;
        sent += bytes_sent;
     }

     return 0;

 }
```

****** There are multiple other correct solutions here.***

QUESTION 4: Short answers (29 points)

1. [6 points] IP Address Aggregation

Suppose you are network operator for an ISP, and your ISP has two customer networks (MIT and Princeton). For the sake of simplicity, let's assume that MIT only has 3 hosts with the following IP addresses, and Princeton only has 2 hosts:

MIT: 18.0.0.1, 18.0.0.254, 18.255.255.254

Princeton: 128.112.0.1, and 128.112.255.254

You've heard that memory in router line cards is expensive, so you'd like to use as few entries as possible in your line cards. Your colleague at work says he has a solution that only requires two forwarding-table entries.

(a) Give the IP prefix notation that your colleague uses to represent these two networks.

18.0.0.0/8 and 128.112.0.0/16

(b) After you install the forwarding table entry your colleague suggests, you notice that you still see traffic for host 18.255.255.254, but you've stopped seeing any traffic for hosts 18.0.0.1 and 18.0.0.254. So, you come up with a new set of forwarding-table entries – three this time – that, once installed, causes you to receive traffic for all five hosts. What are these three IP prefixes?

18.0.0.0/8, 18.0.0.0/24, and 128.112.0.0./16

(c) Give one possible explanation why this change in part (b) is needed.

18.0.0.0/24 is multi-homed from another provider and, given longest prefix matching for preferred traffic, all traffic was going through its other provider.

Similarly, some other ISP (perhaps again for multi-homing) could be announcing any prefix > 8 bits, e.g., 18.0.0.0/16

2. [4 points] Ethernet and IP

(a) Suppose Ethernet was the only existing LAN technology, so every host in the Internet was part of a local Ethernet and thus had a globally-unique Ethernet address. Would you recommend getting rid of IP addresses by simply using Ethernet addresses instead of IP addresses? Why or why not?

> ***No. It would not scale due to (1) the need for broadcast for discovery and (2) forwarding tables would be large as MAC addresses are not topologically assigned and thus cannot be aggregated.***

(b) What about the other way around, why do we not simply assign IP addresses to network adaptors, instead of dealing with both Ethernet and IP addresses?

> ***No. Either you keep topological aggregation but then require adaptors to get manually reassigned an IP address whenever moving (a configuration nightmare), or adaptors have static IP addresses but then you would sacrifice routing table aggregation (a scalability nightmare).***

3. [8 points] Using TCP, a sender has sent out packets 1 through 20. The sender receives an ACK for packet 10, then it receives three ACKS for packet 11 (i.e., if the final byte of packet 11 was byte no. 5000, then these ACKs were for byte 5001). Given this result:

(a) Which of the 20 packets can the sender assume are lost?

> ***12***

(b) Which of the 20 packets can the sender assume were definitely received?

> ***1 - 11***

Now, the same sender resends the next missing packet and receives an ACK for packet 16 in response. Based on this information:

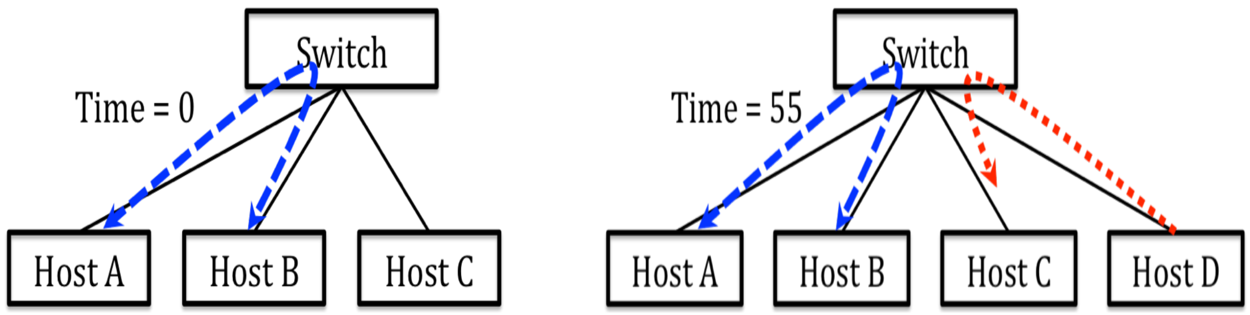
(c) Which packets of the original 20 can the sender assume are still lost, if any?

> ***None***

(d) Which of the original 20 can the sender assume were definitely received?

> ***1-16***

4. [5 points] Assume you have a network with three hosts – A, B, and C – all directly connected to the same switch. Host A first attempts to open a TCP connection to B at time 0, during which time he adds an entry to his ARP cache with a TTL of 60 seconds. Host B begins transferring a large file to host A. A new host D connects to the network, but is configured with the same IP address as already being used by A.



At time 55 seconds, an application running on host D opens a TCP connection to an application on C. A few seconds later, the TCP transfer between hosts A and B breaks. What were the steps that led to this connection breaking?

At time 60, host B's ARP cache expires, so when it re-issues an ARP request for the IP address shared by A and D, D's answer is cached by B. Thus, B's TCP packet gets sent to MAC-D, who, not having such an open TCP connection, will reply with a TCP-RST, causing B's connection to break. (Similar for A and its ARP cache)

5. [6 points] What transport layer protocol (TCP or UDP) you would use if your application needed one of the following properties and then give one succinct reason why. Be specific.

(a) Speed - ***UDP – no connection setup or sender-side buffering***

(b) Long messages – ***TCP – as stream-of-bytes abstraction supports arbitrarily long messages. Many stacks limit UDP messages to some maximum size (e.g., 64KB)***

(c) Short-duration interactions – ***UDP – No connection setup delay***

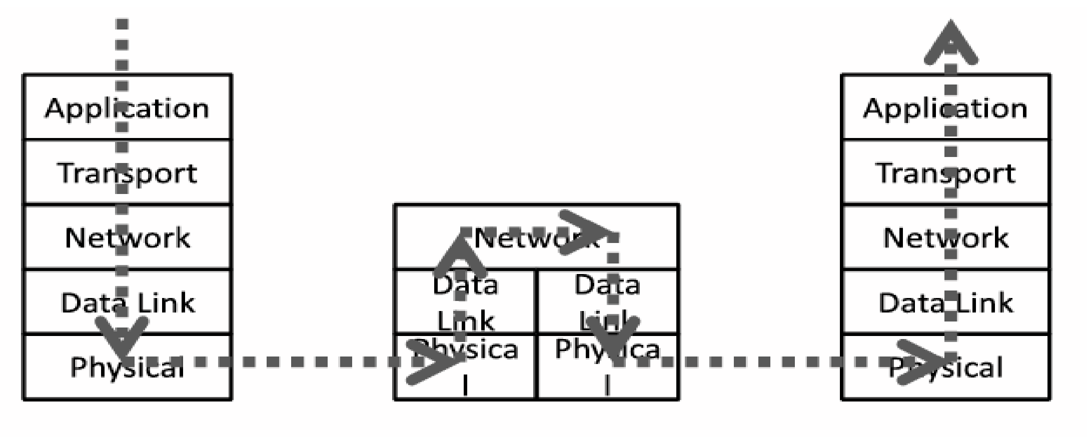
(d) Fairness between competing flows – ***TCP – Fairness through congestion control***

(e) Flooding – ***UDP – No connection state needed***

(f) Flexibility – ***UDP – More application-level control***

QUESTION 5: Multiple choice (17 points)

1. What is the name of the forwarding device in the middle of the diagram below? (Circle ONE answer – 1 point)



(a) Switch
(b) Router
(c) Hub
(d) Repeater

2. When a UDP segment arrives at a host, in order to direct the segment to the appropriate socket, the operating system's network stack uses the following fields: (circle ALL that are true – 2 points)

(a) the source IP address.
(b) the destination IP address
*(*c) the source port number
(d) the destination port number

3. When a TCP segment belonging to an existing connection arrives at a host, in order to direct the segment to the appropriate socket the operating system's network stack uses the following fields (circle ALL that are true – 2 points)

(a) the source IP address.
(b) the destination IP address
(c) the source port number
(d) the destination port number

4. Which of the following link protocols best describes Ethernet? (circle ONE – 1 point)

(a) frequency-division multiple access (FDMA)
(b) time-division multiple access (TDMA)
(c) token-passing
(d) carrier sense multiple access (CSMA)

5. If you are building a network in which only one host will be sending traffic, which of the following link protocols would give the best throughput? (circle ONE – 1 point)

(a) frequency-division multiple access (FDMA)
(b) time-division multiple access (TDMA)
(c) token-passing
(d) carrier sense multiple access (CSMA)

6. Which of the following are advantages that RED has over drop-tail queuing? (circle ALL that are true – 3 points)

(a) has fewer burst losses
(b) has shorter queuing delays
(c) ensures roughly equal throughput for all flows
(d) ensures roughly equal throughput for all TCP flows

7. Which of the following are true about Ethernet networks? (circle ALL true – 3 points)

(a) Ethernet frames have a minimum size to ensure good utilization of the network (i.e., senders could otherwise have a large overhead to send a small piece of data).
(b) Bridges provide greater scalability for Ethernet networks than hubs.
(c) Hosts on different segments of a switched Ethernet network will only see packets to hosts on their segments or to the broadcast address.
(d) When an Ethernet sender detects that the media is idle, it sends a jam signal onto the media to tell other devices not to transmit, and then it sends its packet.

8. Which is true of the UNIX Socket API? (circle ALL that are true – 3 points)

(a) If a chunk of data written to a TCP socket is smaller than the MSS, it will all be sent in the same packet.
(b) Writing a single message to a UDP socket can result in the network stack sending multiple packets.
(c) The socket allocated by the accept() system call is assigned a new port number
(d) You can call connect() on a UDP socket

9. [1 point] Describe one thing you would like to see changed in the class. (Only wrong answer is "nothing")