ICSI 416/516 Homework 3 – Network Layer
20 points

Due date: Wednesday 4/18 at 11:59PM as a single PDF file via Blackboard

All parts of the assignment are to be completed independently. Submission of the same text by multiple students will be considered cheating. Students caught cheating will receive 0 points for the assignment and will be reported.

Part 1 [15 points]: Answer the theoretical questions on the following two pages.
Part 2 [5 points]: Complete the Wireshark Lab and answer the questions in the lab. You can download the Wireshark trace for this assignment from
http://www.cs.albany.edu/~mariya/courses/csi416516S16/hw/ip-wireshark-trace-1

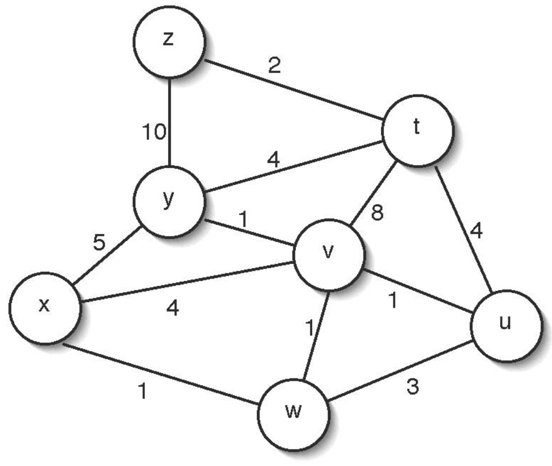
[Theoretical questions]

1. [2 points] Consider a datagram network using 32-bit host addresses. Suppose a router has four links, numbered 0 through 3, and packets are to be forwarded to the link interfaces as follows:

Destination Address Range	Link Interface
11010000 00000000 00000000 00000000 through 11010000 11111111 11111111 11111111	0
11010001 00000000 00000000 00000000 through 11010001 00000000 11111111 11111111	1
11010001 00000000 00000000 00000000 through 11010001 11111111 11111111 11111111	2
Otherwise	3

   a. Provide the corresponding forwarding table that has four entries, uses longest-prefix matching, and forwards packets to the correct link interfaces. In other words, what is the prefix that would correspond to each of the address ranges?
   b. What is the correct interface to use for each of the following addresses:
      - 10011000 10010001 01010001 01010101
      - 11010001 00000000 11000011 00111100
      - 11010001 10000000 00010001 01110111
   c. Rewrite the forwarding table you created for part (a) using the a.b.c.d/x notation instead of the binary string notation.
2. [2 points] Consider the network in Figure 4.22 of your textbook. Suppose that the ISP instead assigns the router the address 128.111.90.32 and that the network address of the home network is 192.168/16.
   a. Assign addresses to all interfaces in the home network.
   b. Suppose each host has two ongoing TCP connections, all to port 80 at host 112.72.30.127. Provide the six corresponding entries in the NAT translation table in the router.
3. [2 points] Now let's think about the impact of NAT on P2P applications. Suppose a peer with user name Anne discovers through querying that a peer with user name Bob has a file she wants to download. Also suppose Bob is behind a NAT whereas Anne isn't. Let 138.76.29.7 be the WAN-side address of the NAT and let 10.0.0.1 be the internal IP address for Bob. Assume that the NAT is not specifically configured for the P2P application. Discuss why Anne's peer cannot initiate a TCP connection to bob's peer, even if Anne know the WAN-side address of the NAT, 138.76.29.7.
4. [2 points] Consider the following prefix 169.226.2.0/24.
   a. Write down the network mask in dotted-decimal notation.
   b. Calculate the corresponding network address.
   c. Calculate the corresponding broadcast address.
   d. How many host addresses are there in this prefix?
   e. How many IP addresses in total are there in this prefix?

5. [2 points] Consider the following network. With the indicated link costs, use Dijkstra's shortest-path algorithm to compute the shortest path from x to all network nodes. Show how the algorithm works by computing a table similar to Table 4.3 in your book.



6. [1 point] Consider the three node topology shown in Figure 4.30 in your book. Rather than having the link costs as shown in the figure, use the link costs of c(x,y)=5, c(y,z)=9, and c(x,z)=3. Compute the distance tables after the initialization step and after each iteration of a synchronous version of the distance-vector algorithm (as shown in the book for Figure 4.30).
7. [1 point] Compare and contrast the advertisements used by RIP and OSPF.
8. [1 point] Why does the Internet need to use different intra-AS and inter-AS protocols?
9. [2 point] Consider sending a 2400-byte datagram into a link that has an MTU of 700 bytes. Suppose the original datagram is stamped with the identification number 422.
    a. How many fragments are generated?
    b. What are the values in the various fields in the IP datagram(s) generated related to fragmentation?
    c. Is fragmentation a preferred method to handle variable MTU in todays Internet? Why/why not? What alternatives exist?

Wireshark Lab: IP

In this lab, we'll investigate the IP protocol, focusing on the IP datagram. We'll do so by analyzing a trace of IP datagrams sent and received by an execution of the `traceroute` program. We'll investigate the various fields in the IP datagram, and study IP fragmentation in detail.

To best understand and work through this lab, there are a few resources you can take advantage of. Section 1.4.3 in the text and section 3.4 of RFC 2151 [ftp://ftp.rfc-editor.org/in-notes/rfc2151.txt] review the operation of the `traceroute` program. You should also read Section 4.4 in the text to be sure you understand IP. If you feel that more detail on IP could help, then you should use RFC 791 [ftp://ftp.rfc-editor.org/in-notes/rfc791.txt].

1. Capturing packets from an execution of traceroute

In order to generate the trace of IP datagrams for this lab, the `traceroute` program was used to send datagrams of different sizes towards some destination, *X*. Recall that `traceroute` operates by first sending one or more datagrams with the time-to-live (TTL) field in the IP header set to 1; it then sends a series of one or more datagrams towards the same destination with a TTL value of 2; it then sends a series of datagrams towards the same destination with a TTL value of 3; and so on. A router must decrement the TTL in each received datagram by 1. If the TTL reaches 0, the router returns an ICMP message (type 11 – TTL-exceeded) to the sending host. As a result of this behavior, a datagram with a TTL of 1 (sent by the host executing `traceroute`) will cause the router one hop away from the sender to send an ICMP TTL-exceeded message back to the sender; the datagram sent with a TTL of 2 will cause the router two hops away to send an ICMP message back to the sender; the datagram sent with a TTL of 3 will cause the router three hops away to send an ICMP message back to the sender; and so on. In this manner, the host executing `traceroute` can learn the identities of the routers between itself and destination *X* by looking at the source IP addresses in the datagrams containing the ICMP TTL-exceeded messages.

For this lab we'll investigate the `traceroute` program by having it send datagrams of various lengths. As with the previous assignments, the trace you will use in Wireshark has been generated for you. However, if you would like to run the traceroute program yourself to experiment with the different options, follow the following instructions.

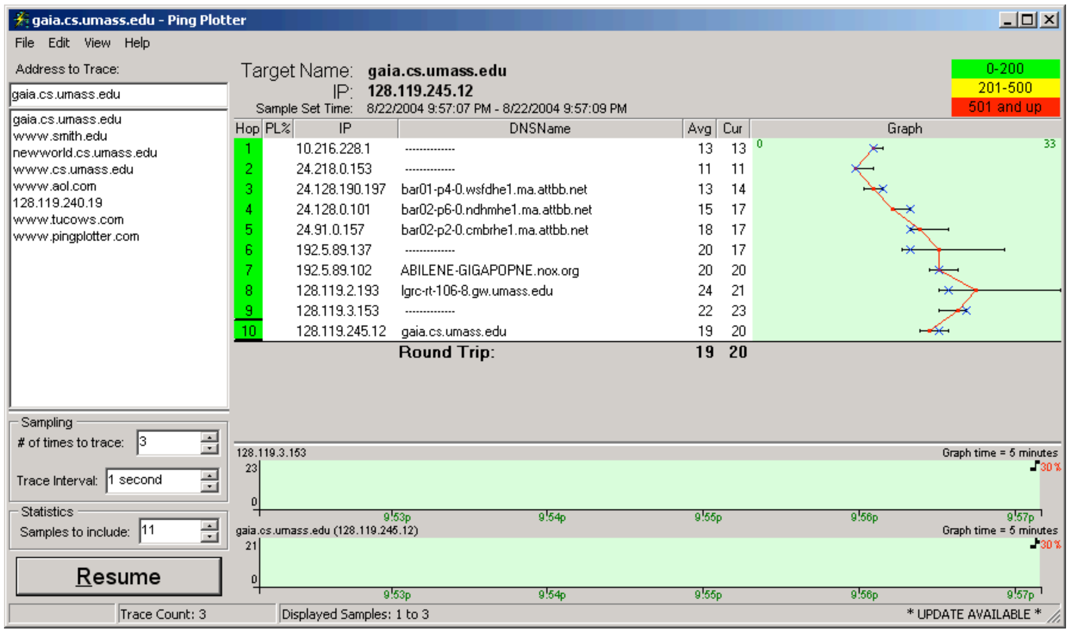
Note: experimenting with traceroute is OPTIONAL, yet encouraged:

- **Windows.** The `tracert` program provided with Windows does not allow one to change the size of the ICMP echo request (ping) message sent by the `tracert` program. A nicer Windows `traceroute` program is *pingplotter*, available both in free version and shareware versions at http://www.pingplotter.com. Download and install *pingplotter*, and test it out by performing a few traceroutes to your favorite sites. The size of the ICMP echo request message can be explicitly set in *pingplotter* by selecting the menu item *Edit->Advanced Options->Packet Options* and then filling in the *Packet Size* field. The default packet size is 56 bytes. Once *pingplotter* has sent a series of packets with the increasing TTL values, it restarts the sending process again with a TTL of 1, after waiting *Trace Interval* amount of time. The value of *Trace Interval* and the number of intervals can be explicitly set in *pingplotter*.
- **Linux/Unix.** With the Unix `traceroute` command, the size of the UDP datagram sent towards the destination can be explicitly set by indicating the number of bytes in the datagram; this value is entered in the `traceroute` command line immediately after the name or address of the destination. For example, to send `traceroute` datagrams of 100 bytes towards gaia.cs.umass.edu, the command would be:

```
        %traceroute gaia.cs.umass.edu 100
```

The Wireshark trace was captured through the following steps:

- Packet capture with Wireshark was started on a Windows machine.
- Three sets of traceroutes were executed. In the first set, three pings with a packet size of 56 bytes were sent to a destination. In the second set, three pings with a packet size of 2000 bytes were sent to a destination. In the final set, three pings with a packet size of 3500 bytes were sent to a destination.
- If you try this out with the pingplotter on Windows, you will get a screenshot that looks something like this:



Address to Trace: gaia.cs.umass.edu

gaia.cs.umass.edu
www.smith.edu
newworld.cs.umass.edu
www.cs.umass.edu
www.aol.com
128.119.240.19
www.tucows.com
www.pingplotter.com

Target Name: **gaia.cs.umass.edu**
IP: **128.119.245.12**
Sample Set Time: 8/22/2004 9:57:07 PM - 8/22/2004 9:57:09 PM

🟩 0-200
🟨 201-500
🟥 501 and up

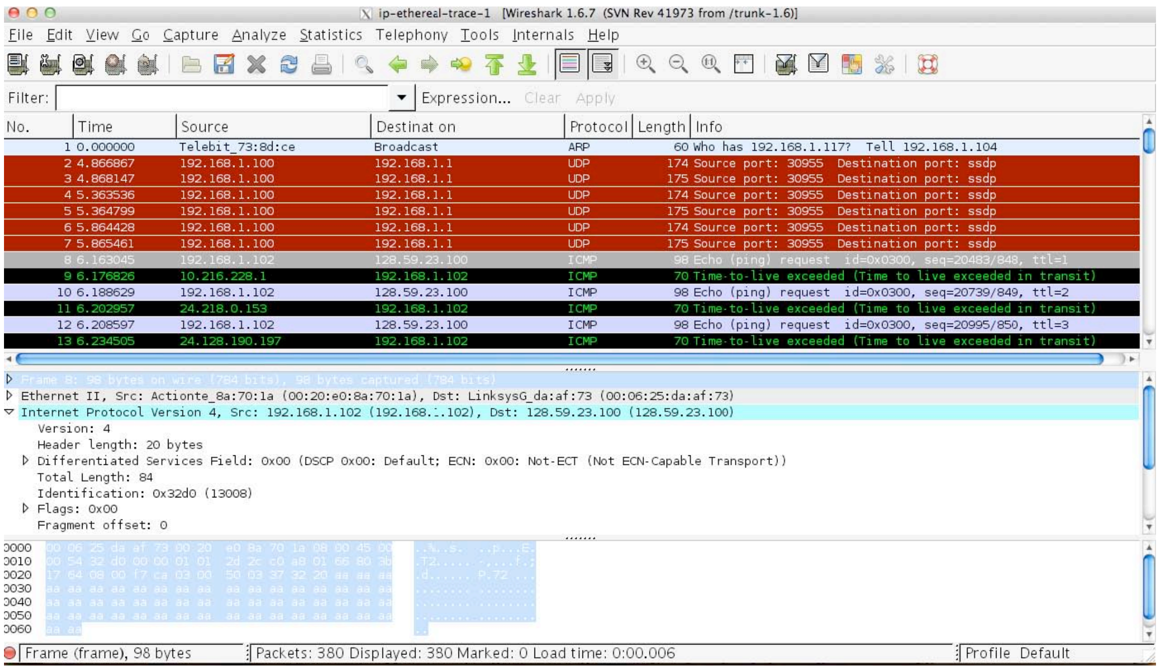
Hop	PL%	IP	DNSName	Avg	Cur
1		10.216.228.1	--------------	13	13
2		24.218.0.153	--------------	11	11
3		24.128.190.197	bar01-p4-0.wsfdhe1.ma.attbb.net	13	14
4		24.128.0.101	bar02-p6-0.ndhmhe1.ma.attbb.net	15	17
5		24.91.0.157	bar02-p2-0.cmbrhe1.ma.attbb.net	18	17
6		192.5.89.137	--------------	20	17
7		192.5.89.102	ABILENE-GIGAPOPNE.nox.org	20	20
8		128.119.2.193	lgrc-rt-106-8.gw.umass.edu	24	21
9		128.119.3.153	--------------	22	23
10		128.119.245.12	gaia.cs.umass.edu	19	20
			Round Trip:	**19**	**20**

Sampling
# of times to trace: 3
Trace Interval: 1 second

Statistics
Samples to include: 11

128.119.3.153 — Graph time = 5 minutes
gaia.cs.umass.edu (128.119.245.12) — Graph time = 5 minutes

Trace Count: 3 | Displayed Samples: 1 to 3 | * UPDATE AVAILABLE *


ip-ethereal-trace-1 [Wireshark 1.6.7 (SVN Rev 41973 from /trunk-1.6)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression...
Clear
Apply

```
No.  Time        Source             Destinat on      Protocol Length Info
   1 0.000000   Telebit_73:8d:ce   Broadcast        ARP         60 Who has 192.168.1.117?  Tell 192.168.1.104
   2 4.866867   192.168.1.100      192.168.1.1      UDP        174 Source port: 30955  Destination port: ssdp
   3 4.868147   192.168.1.100      192.168.1.1      UDP        175 Source port: 30955  Destination port: ssdp
   4 5.363536   192.168.1.100      192.168.1.1      UDP        174 Source port: 30955  Destination port: ssdp
   5 5.364799   192.168.1.100      192.168.1.1      UDP        175 Source port: 30955  Destination port: ssdp
   6 5.864428   192.168.1.100      192.168.1.1      UDP        174 Source port: 30955  Destination port: ssdp
   7 5.865461   192.168.1.100      192.168.1.1      UDP        175 Source port: 30955  Destination port: ssdp
   8 6.163045   192.168.1.102      128.59.23.100    ICMP        98 Echo (ping) request  id=0x0300, seq=20483/848, ttl=1
   9 6.176826   10.216.228.1       192.168.1.102    ICMP        70 Time-to-live exceeded (Time to live exceeded in transit)
  10 6.188629   192.168.1.102      128.59.23.100    ICMP        98 Echo (ping) request  id=0x0300, seq=20739/849, ttl=2
  11 6.202957   24.218.0.153       192.168.1.102    ICMP        70 Time-to-live exceeded (Time to live exceeded in transit)
  12 6.208597   192.168.1.102      128.59.23.100    ICMP        98 Echo (ping) request  id=0x0300, seq=20995/850, ttl=3
  13 6.234505   24.128.190.197     192.168.1.102    ICMP        70 Time-to-live exceeded (Time to live exceeded in transit)

▷ Frame 8: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
▷ Ethernet II, Src: Actionte_8a:70:1a (00:20:e0:8a:70:1a), Dst: LinksysG_da:af:73 (00:06:25:da:af:73)
▽ Internet Protocol Version 4, Src: 192.168.1.102 (192.168.1.102), Dst: 128.59.23.100 (128.59.23.100)
    Version: 4
    Header length: 20 bytes
  ▷ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    Total Length: 84
    Identification: 0x32d0 (13008)
  ▷ Flags: 0x00
    Fragment offset: 0

0000  00 06 25 da af 73 00 20  e0 8a 70 1a 08 00 45 00   ..%..s.  ..p...E.
0010  00 54 32 d0 00 00 01 01  2d 2c c0 a8 01 66 80 3b   .T2.....  -,...f.;
0020  17 64 08 00 f7 ca 03 00  50 03 37 32 20 aa aa aa   .d......  P.72 ...
0030  aa aa aa aa aa aa aa aa  aa aa aa aa aa aa aa aa   ........  ........
0040  aa aa aa aa aa aa aa aa  aa aa aa aa aa aa aa aa   ........  ........
0050  aa aa aa aa aa aa aa aa  aa aa aa aa aa aa aa aa   ........  ........
0060  aa aa                                              ..

Frame (frame), 98 bytes | Packets: 380 Displayed: 380 Marked: 0 Load time: 0:00.006 | Profile: Default
```

2. A look at the captured trace

In the trace, you should be able to see the series of ICMP Echo Requests sent by the client computer and the ICMP TTL-exceeded messages returned to the computer by the intermediate routers.

1. Select the first ICMP Echo Request message sent by the client computer, and expand the Internet Protocol part of the packet in the packet details window, as shown in the figure below. You should see something similar to the image above. What is the IP address of the client?
2. Within the IP packet header, what is the value in the upper layer protocol field? What is the purpose of this field?
3. Has this IP datagram been fragmented? Explain how you determined whether or not the datagram has been fragmented.

Next, sort the traced packets according to IP source address by clicking on the *Source* column header; a small downward pointing arrow should appear next to the word *Source*. If the arrow points up, click on the *Source* column header again. Select the first **ICMP Echo Request message** sent by the client computer, and expand the Internet Protocol portion in the "details of selected packet header" window. In the "listing of captured packets" window, you should see all of the subsequent ICMP messages (perhaps with additional interspersed packets sent by other protocols running on the computer) below this first ICMP. Use the down arrow to move through the ICMP messages sent by the computer.

4. Which fields in the IP datagram *always* change from one datagram to the next within this series of ICMP messages sent by your computer? Why must those fields change?
5. Describe the pattern you see in the values in the Identification field of the IP datagram.

Fragmentation

Sort the packet listing according to time by clicking on the *Time* column.

6. Find the first ICMP Echo Request message that was sent by the computer after the *Packet Size* was changed to 2000. How many datagrams has that message been fragmented across?
7. What information in the IP header indicates that the datagram been fragmented? What information in the IP header indicates whether this is the first fragment versus a latter fragment?
8. Now look at the second fragment of the IP datagram. What information in the IP header indicates that this is not the first datagram fragment? Are there more fragments? How can you tell?

Now find the first ICMP Echo Request message that was sent by the computer after the *Packet Size* was changed to 3500.

9. How many fragments were created from the original datagram?
10. What fields change in the IP header among the fragments?