Homework+Lab 5: Due at start of class on 22-Apr.

Please complete all seven homework problems, and the fourteen lab problems.

Homework Problems

1. An $(n, 1)$ repetition code takes each bit and repeats it n times. What is the minimum distance $d_{\min}$ of such a code?

2. Suppose you have a codebook with minimum distance $d_{\min} = 5$. What is the maximum number of errors this code can correct?

3. Which encoding scheme is better, a Hamming (7,4) code or a (3,1) repetition code? Explain why.

4. For the Hamming (7,4) code, what does the bit sequence 0101 get encoded as? Show your work. You can use a computer to check your answer but make sure you understand the procedure.

5. Consider the (5,2) code

Data Bits	Codeword
00	00000
01	00001
10	10000
11	10001

   Is this a good code? Explain why or why not.

6. Suppose you are using a (7,4) Hamming code and receive the codeword 0101001. Confirm that this is not a valid (7,4) Hamming codeword. Determine the closest valid (7,4) Hamming codeword and identify the incorrect bit in the received codeword.

7. Stallings Problem 6.16.

Lab 5: Network Layer – IP [adapted from J. Kurose and K.W. Ross]

In this lab, we'll investigate the IP protocol, focusing on the IP datagram[1]. We'll do so by analyzing a trace of IP datagrams sent and received by an execution of the traceroute program (the `traceroute` program itself is explored in more detail in the Wireshark ICMP lab). We'll investigate the various fields in the IP datagram, and study IP fragmentation in detail. Before beginning this lab, please read sections 14.1-14.3 in the text and section 3.4 of RFC 2151 `ftp://ftp.rfc-editor.org/in-notes/rfc2151.txt` to familiarize yourself on the operation of the `traceroute` program.

Part I: Capturing packets from an execution of `traceroute`

In order to generate a trace of IP datagrams for this lab, we'll use the `traceroute` program to send datagrams of different sizes towards some destination, X. The `traceroute` application 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. In the TCP/IP protocol suite, routers 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.

We'll want to run `traceroute` and have it send datagrams of various lengths.

- **Windows.** The `tracert` program provided with MS 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 as a trial version at http://www.pingplotter.com[2]. 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→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/OSX.** With the Unix `traceroute` command, the size of the UDP[3] 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 2000 bytes towards `tholian.wpi.edu`, the command would be:

  ```
  traceroute tholian.wpi.edu 2000
  ```

[1] *Datagram* = "packet segment".

[2] Note that the freeware version does not provide all the functionality we will need; you will need to download the 30-day trial of the standard edition.

[3] UDP is a transport layer protocol. Unlike TCP, however, UDP does not guarantee delivery, preservation of sequence, or protection again duplication. For more details, see section 15.3 of the textbook.

Do the following:

- Shut down any programs on your computer that might generate network traffic (e.g. Skype, IM, browsers running Flash, etc). While this isn't crucial, having other network traffic going on at the same time as the capture will make it harder to see the packets relevant to this lab.
- Start up Wireshark and begin packet capture (Capture→Option) and then press OK on the Wireshark Packet Capture Options screen (we'll not need to select any options here).
- If you are using a **Windows** platform, start up pingplotter and enter the name of a target destination in the "Address to Trace Window." Enter 3 in the "# of times to Trace" field, so you don't gather too much data. Select the menu item Edit→Advanced Options→Packet Options and enter a value of 56 in the Packet Size field and then press OK. Then press the Trace button. You should see a pingplotter window that looks something like Fig. 1.

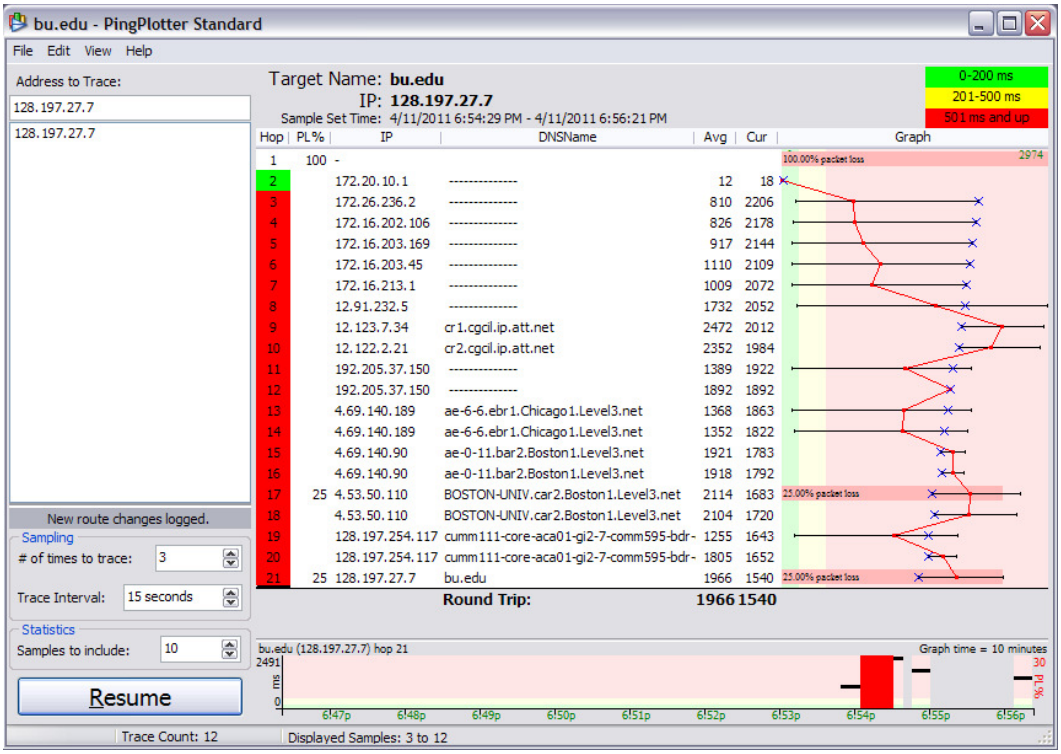

Figure 1: Example trace taken on a train moving at 60 mph (which explains the high ping times)

Next, send a set of datagrams with a longer length, by selecting Edit→Advanced Options→Packet Options and enter a value of 2000 in the Packet Size field and then press OK. Then press the Resume button. Finally, send a set of datagrams with a longer length, by selecting Edit→Advanced Options→Packet Options and enter a value of 3500 in the Packet Size field and then press OK. Then press the Resume button. Stop Wireshark tracing.

- If you are using a **Unix/OSX** platform, enter three traceroute commands to a destination of your choice, first with a packet length of 56 bytes, then with a length of 2000 bytes, and lastly with a length of 3500 bytes.
- Stop Wireshark tracing.

Comment: In choosing a target destination for your traceroute, you should choose an educational institution besides WPI (unless you are attempting this lab from an off-campus location).

Part II: A look at the captured trace

To help the relevant packets stand out, you will probably want to apply a filter as we have done in previous labs. One option would be to set `ip.addr==[your IP address]`. In your trace, you should be able to see the series of ICMP Echo Request (in the case of Windows machine) or the UDP segment (in the case of Unix) sent by your computer and the ICMP TTL-exceeded messages returned to your computer by the intermediate routers. In the questions below, we'll assume you are using a Windows machine; the corresponding questions for the case of a Unix/OSX machine should be clear. When answering a question you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout to explain your answer. To print a packet, use File→Print, choose Selected packet only, choose Packet summary line, and select the minimum amount of packet detail that you need to answer the question.

1. Select the first ICMP Echo Request message sent by your computer, expand the Internet Protocol part of the packet in the packet details window, and print this.

2. Within the IP packet header, what is the value in the upper layer protocol field?

3. How many bytes are in the IP header? How many bytes are in the payload of the IP datagram? Explain how you determined the number of payload bytes.

4. Has this IP datagram been fragmented? Explain how you determined whether or not the datagram has been fragmented[4].

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 your 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 your computer) below this first ICMP. Use the down arrow on your keyboard to move through the ICMP messages sent by your computer (Note: in the following, we only concern the messages with source addresses equal to that of your computer).

5. Which fields in the IP datagram always change from one datagram to the next within this series of ICMP messages sent by your computer?

6. Which of the fields must stay constant? Which fields must change? Why?

7. Describe the pattern you see in the values in the Identification field of the IP datagram

Next (with the packets still sorted by source address) find the series of ICMP TTL-exceeded replies sent to your computer by the nearest (first hop) router.

8. What is the value in the Identification field and the TTL field?

9. Do these values remain unchanged for all of the ICMP TTL-exceeded replies sent to your computer by the nearest (first hop) router? Why?

[4]It may be helpful to change the settings in Options → Protocols → IPv4 → "Reassemble fragmented IPv4 datagrams". With this setting enabled, Wireshark will lump all the IPv4 fragments together. With this setting disabled, you can see the contents of each IPv4 fragment separately.

Part III: Fragmentation

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

10. Find the first ICMP Echo Request message that was sent by your computer after you changed the Packet Size in pingplotter to be 2000. Has that message been fragmented across more than one IP datagram?

11. Print out the first fragment of the fragmented IP datagram. 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? How long is this IP datagram?

12. Print out the second fragment of the fragmented 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?

13. What fields change in the IP header between the first and second fragment?

14. Now find the first ICMP Echo Request message that was sent by your computer after you changed the Packet Size in pingplotter to be 3500. How many fragments were created from the original datagram? What fields change in the IP header among the fragments?

What to hand in: answers to questions 1-14, including evidence for how you got the answers.