1. Measuring Round Trip Times With Ping

In the first two exercises, you will use the ping utility to send echo requests to a number of different hosts. The ping utility is one of the more useful utilities for testing a network. It allows you to measure the time that it takes for a packet to travel through the Internet to a remote host and back. The ping utility works by sending a short message, called an *echo-request*, to a host using the Internet Control Message Protocol (ICMP). A host that supports ICMP (and most do) and receives an echo-request message simply replies by sending an *echo-response* back to the originating host.

In many of the exercises, you will be referring to hosts via their DNS names rather than their IP addresses.

For more information about ping, look at the man page on ping and the specifications for ICMP, located in RFC 792. Section 7.13.4 of the course notes describes ICMP as well.

```
% man ping
```

To use the ping command on your machine, run a command such as:

```
% ping www.google.com
```

A. Round Trip Times:

In the following two questions, you are asked to use the ping utility to measure the round trip times to several hosts on the Internet.

For the following hosts, send 10 packets, each with a length of 56 data bytes. The hosts are:

www.ee.hawaii.edu
www.csail.mit.edu
www.berkeley.edu
www.uwa.edu.au

Question 1: Indicate what percentage of packets sent resulted in a successful response. For the packets from which you received a response, write down the minimum, average, and maximum round trip times in milliseconds. Note that ping reports these times to you if you tell it how many packets to send on the command line.

Question 2: Explain the differences in minimum round trip time to each of these hosts.

Question 3: Now send pings with 56, 512 and 1024 byte packets to the 4 hosts above. Write down the minimum, average, and maximum round trip times in milliseconds for each of the 12 pings. Why do you think the minimum round-trip times to the same hosts different when using 56, 512, and 1024 byte packets?

B. Unanswered Pings:

For the following hosts, send 100 packets that have a length of 56 data bytes. Indicate what percentage of the packets resulted in a successful response.

www.wits.ac.za	(University of the Witwatersrand, Johannesburg)
www.microsoft.com	

Question 4: For some of the hosts, you may not have received any responses for the packets you sent. What might be a reason as to why you might have not gotten a response? (Be sure to check the hosts in a web browser.)

2. Understanding Internet routes using traceroute

As the name implies, traceroute essentially allows you to trace the entire route from your machine to a remote machine. The remote machine can be specified either as a name or as an IP address.

We include a sample output of an execution of traceroute and explain the salient features. The command:

```
% traceroute www.google.com
```

tries to determine the path from the source machine to www.google.com. The man page for traceroute (% man traceroute) contains explanations for the remaining fields on each line.

```
% traceroute www.google.com
traceroute to www.google.com (142.250.191.68), 30 hops max, 60 byte packets
 1  _gateway (128.171.61.1)  1.821 ms  1.768 ms  1.737 ms
 2  vl-3060-manoa7050-1.uhnet.net (128.171.185.120)  1.064 ms  1.036 ms  1.318 ms
 3  xe-1-0-0-669-coconut-re0.uhnet.net (128.171.213.13)  2.488 ms  2.461 ms  2.822 ms
 4  xe-0-0-6-53-ohelo-re0.uhnet.net (205.166.205.26)  2.102 ms  2.374 ms  2.347 ms
 5  hundredge-0-0-0-24.3505.core1.seat.net.internet2.edu (64.57.21.49)  75.922 ms  75.895 ms  76.259 ms
 6  * * *
 7  72.14.203.202 (72.14.203.202)  68.998 ms  70.802 ms  70.739 ms
 8  * * *
 9  142.251.55.198 (142.251.55.198)  71.093 ms  71.060 ms 74.125.243.177 (74.125.243.177)  70.996 ms
10  108.170.245.123 (108.170.245.123)  71.966 ms  71.934 ms  71.903 ms
11  142.251.64.250 (142.251.64.250)  70.862 ms 216.239.50.20 (216.239.50.20)  72.357 ms *
12  72.14.239.235 (72.14.239.235)  71.258 ms 142.250.56.35 (142.250.56.35)  70.139 ms  70.106 ms
13  66.249.94.28 (66.249.94.28)  70.412 ms  71.333 ms 142.250.234.140 (142.250.234.140)  70.979 ms
14  108.170.242.81 (108.170.242.81)  69.863 ms 108.170.242.241 (108.170.242.241)  73.296 ms  71.201 ms
15  142.251.224.31 (142.251.224.31)  70.851 ms  70.387 ms 142.251.224.33 (142.251.224.33)  70.355 ms
16  nuq04s43-in-f4.1e100.net (142.250.191.68)  70.324 ms  71.517 ms  69.588 ms
```

A. Basics:

Question 5:

In at most 50 words, explain how traceroute discovers a path to a remote host. The man page might be useful in answering this question.

B. Routine Asymmetries:

For this exercise, we will traceroute in both directions using the traceroute server at http://www.slac.stanford.edu/cgi-bin/nph-traceroute.pl. I've already run the reverse traceroute to a machine on campus with this result (if you try to run that page on campus on your laptop you may be flagged as an attacker):

```
Thu Jan 12 20:32:55 2023: execute(traceroute -m 30 -q 1 -w 1 128.171.61.144
140)=traceroute from 134.79.197.146(www.slac.stanford.edu) to
128.171.61.144(mahina.eng.hawaii.edu) for 134.79.138.4
traceroute to 128.171.61.144 (128.171.61.144), 30 hops max, 140 byte packets
 1  rtr-serv01-02-serv01-dmz-webserv.slac.stanford.edu (134.79.197.131)  0.589 ms
 2  rtr-core1-p2p-serv01-01.slac.stanford.edu (134.79.253.249)  0.509 ms
 3  rtr-fwcore2-trust-p2p-core2.slac.stanford.edu (134.79.254.146)  1.220 ms
 4  rtr-core2-p2p-fwcore2-untrust.slac.stanford.edu (134.79.254.149)  1.508 ms
 5  rtr-border2-7k-p2p-core1.slac.stanford.edu (134.79.252.181)  1.357 ms
 6  slac50s-cr6-ip-p2p-border2-7k.slac.stanford.edu (192.68.191.233)  1.391 ms
 7  slac50s-cr6--sunn-bb-c.igp.es.net (134.55.57.144)  1.895 ms
 8  aarnet-2-is-jmb-778.sttlwa.pacificwave.net (207.231.245.4)  19.955 ms
 9  aarnet-2-is-jmb-778.sttlwa.pacificwave.net (207.231.245.4)  19.973 ms
10  et-2-0-0.pe1.a.hnl.aarnet.net.au (113.197.15.200)  69.761 ms
11  et-1-2-0-802-ohelo-re0.uhnet.net (205.166.205.134)  68.652 ms
12  xe-1-1-0-63-coconut-re0.uhnet.net (205.166.205.37)  66.205 ms
13  xe-1-1-0-63-coconut-re0.uhnet.net (205.166.205.37)  66.022 ms
14  vl-3060-ex4550-holmesl3bldg.uhnet.net (128.171.185.121)  67.745 ms
15  *
16  *
17  *
18  *
19  *
20  *
21  *
22  *
23  *
24  *
25  *
26  *
27  *
28  *
29  *
30  *
traceroute -m 30 -q 1 -w 1 128.171.61.144 140 (traceroute.pl process #=1/max=14) took
2secs. Total script traceroute.pl time=3secs.
```

Now run this on your machine:

% traceroute www1.slac.stanford.edu

Question 6: Describe anything unusual about the output. Are the same routers traversed in both directions? If not, why might this happen?

3. Using Wireshark

1. Start up your favorite web browser, which will display your selected homepage.
2. Start up the Wireshark software. You will initially see a window similar to that shown in Figure 2. Wireshark has not yet begun capturing packets.
3. To begin packet capture, select the Capture pull down menu and select Interfaces. This will cause the "Wireshark: Capture Interfaces" window to be displayed (on a PC) or you can choose Options on a Mac. You should see a list of interfaces, as shown in Figures 4a (Windows) and 4b (Mac).
4. You'll see a list of the interfaces on your computer as well as a count of the packets that have been observed on that interface so far. On a Windows machine, click on Start for the interface on which you want to begin packet capture (in the case in Figure 4a, the Gigabit network Connection). On a Windows machine, select the interface and click Start on the bottom of the window). Packet capture will now begin - Wireshark is now capturing all packets being sent/received from/by your computer!
5. Once you begin packet capture, a window similar to that shown in Figure 3 will appear. This window shows the packets being captured. By selecting Capture pulldown menu and selecting Stop, or by click on the red Stop square, you can stop packet capture. But don't stop packet capture yet. Let's capture some interesting packets first. To do so, we'll need to generate some network traffic. Let's do so using a web browser, which will use the HTTP protocol that we will study in detail in class to download content from a website.
6. While Wireshark is running, enter the URL:
   http://gaia.cs.umass.edu/wireshark-labs/INTRO-wireshark-file1.html
   and have that page displayed in your browser. In order to display this page, your browser will contact the HTTP server at gaia.cs.umass.edu and exchange HTTP messages with the server in order to download this page, as discussed in section 2.2 of the text. The Ethernet or WiFi frames containing these HTTP messages (as well as all other frames passing through your Ethernet or WiFi adapter) will be captured by Wireshark.
7. After your browser has displayed the INTRO-wireshark-file1.html page (it is a simple one line of congratulations), stop Wireshark packet capture by selecting stop in the Wireshark capture window. The main Wireshark window should now look similar to Figure 3. You now have live packet data that contains all protocol messages exchanged between your computer and other network entities! The HTTP message exchanges with the gaia.cs.umass.edu web server should appear somewhere in the listing of packets captured. But there will be many other types of packets displayed as well (see, e.g., the

many different protocol types shown in the Protocol column in Figure 3). Even though the only action you took was to download a web page, there were evidently many other protocols running on your computer that are unseen by the user. We'll learn much more about these protocols as we progress through the text! For now, you should just be aware that there is often much more going on than "meet's the eye"!

8. Type in "http" (without the quotes, and in lower case – all protocol names are in lower case in Wireshark, and make sure to press your enter/return key) into the display filter specification window at the top of the main Wireshark window. Then select Apply (to the right of where you entered "http") or just hit return. This will cause only HTTP message to be displayed in the packet-listing window. Figure 5 below shows a screenshot after the http filter has been applied to the packet capture window shown earlier in Figure 3. Note also that in the Selected packet details window, we've chosen to show detailed content for the Hypertext Transfer Protocol application message that was found within the TCP segment, that was inside the IPv4 datagram that was inside the Ethernet II (WiFi) frame. Focusing on content at a specific message, segment, datagram and frame level lets us focus on just what we want to look at (in this case HTTP messages).

9. Find the HTTP GET message that was sent from your computer to the gaia.cs.umass.edu HTTP server. (Look for an HTTP GET message in the "listing of captured packets" portion of the Wireshark window (see Figures 3 and 5) that shows "GET" followed by the gaia.cs.umass.edu URL that you entered. When you select the HTTP GET message, the Ethernet frame, IP datagram, TCP segment, and HTTP message header information will be displayed in the packet-header window. By clicking on '+' and '-' and right-pointing and down-pointing arrowheads to the left side of the packet details window, minimize the amount of Frame, Ethernet, Internet Protocol, and Transmission Control Protocol information displayed. Maximize the amount information displayed about the HTTP protocol. Your Wireshark display should now look roughly as shown in Figure 5. (Note, in particular, the minimized amount of protocol information for all protocols except HTTP, and the maximized amount of protocol information for HTTP in the packet-header window).

10. Exit Wireshark