Programing Project #3.0 - Ping

Overview

In this project you will implement a simplified version of the **ping** utility. The goal is to provide a better understanding of how the network layer behaves by hand coding and processing network layer packets. To create the program you will have to use the "*raw*" socket interface. Unlike the TCP and UDP sockets you used for the first two assignments the raw socket API does minimal processing of the buffer before passing it to the link layer. This means you must build all the network and transport layer headers by hand. Unfortunately because of the way raw sockets must be implemented in the kernel most multi-user operating systems require elevated privileges to access the raw socket API. This means that you will not be able to use isengard for this assignment. You may use your own computer or if you prefer you can install a LinuxVM in on one of the lab computers in the CTLM and program from that (if you need help with this process please let me know).

The ping process is fairly straightforward. Your program will build and send an ICMP echo_request packet and then wait for the target to reply with an ICMP echo_reply. By computing the time between the send and receipt of the reply you can compute the RTT. By repeating this process several times in a row you can get a rough estimate of the expected RTT and amount of jitter in the network.

Program Outline

Your program will take one command-line parameter and should report the minimum, maximum and average RTT.

```
$> ping imagine.mines.edu
min = 1.2ms max = 3.0ms ave=2ms
```

In broad terms your program will need to do the following:

1. Read the target hostname from the command line.
2. Convert the target hostname to an IP address (this time you can use the DNS library routines).
3. Build the packet
   - 3.1. Fill in the IP header.
   - 3.2. Fill in the ICMP echo_request header.
   - 3.3. There is no "body" or data in the packet.
4. Create the raw socket using the IPNET family and ICMP protocol type.
5. Do the following five times
   - 5.1. Set a timer for 15 seconds (use the alarm signal).
   - 5.2. Send the packet and make note of the time.
   - 5.3. Wait for the response.
   - 5.4. If the response is received confirm it is yours and then compute the RTT.
   - 5.5. If the timer goes off (i.e., you didn't get a response) print a message.
6. Report the min, max and average RTT.

Submission

This assignment is due before the end of the day Sunday, November 17th. You may use whatever programing environment/language you choose but as with pervious assignments I will be most able to help (and most forgiving when grading) if you use C or C++.

Your submission must compile and run on isengard. I know you will not be able to test running the program on isengard because of the need for root privileges, but you should be able to confirm that it will compile and that you have not used a library that is non-existent.

To submit, create a directory named with your username. Place the code, a makefile and a README.txt into the directory, tar and compress the directory (name the tar file with your username.tgz) and submit the tarfile to blackboard.