COMP/ELEC 429/556 - Introduction to Computer Networks
Project 1 - Ping-Pong Client/Server

Handed out: January 20

Due: February 10 at 11:55pm

I. Introduction

The primary goal of this project is to let students get some experience with the design and implementation of networked applications and protocols, including how to format data and use sockets. Through this project, students will also gain hands-on experience with measuring network performance, concurrent software design, and I/O handling.

In this project, you are to develop a ping-pong client program and server program. The ping-pong client and server together are used to measure the network's performance in transmitting messages of various sizes.

II. The Ping-Pong Client and Server (100 points)

The ping-pong client program should take 4 command line parameters, in the following order:

- *hostname*

  The host where the server is running. You should support connecting to a server by domain name (current list of CLEAR servers: ccnc-{01,02,05,06,07,08,09}.arc.rice.edu).

- *port*

  The port on which the server is running (on CLEAR, the usable range is 18000 <= port <= 18200).

- *size*

  The size in bytes of each message to send (18 <= size <= 65,535). The minimum and maximum size is related the design of the message format. See below.

- *count*

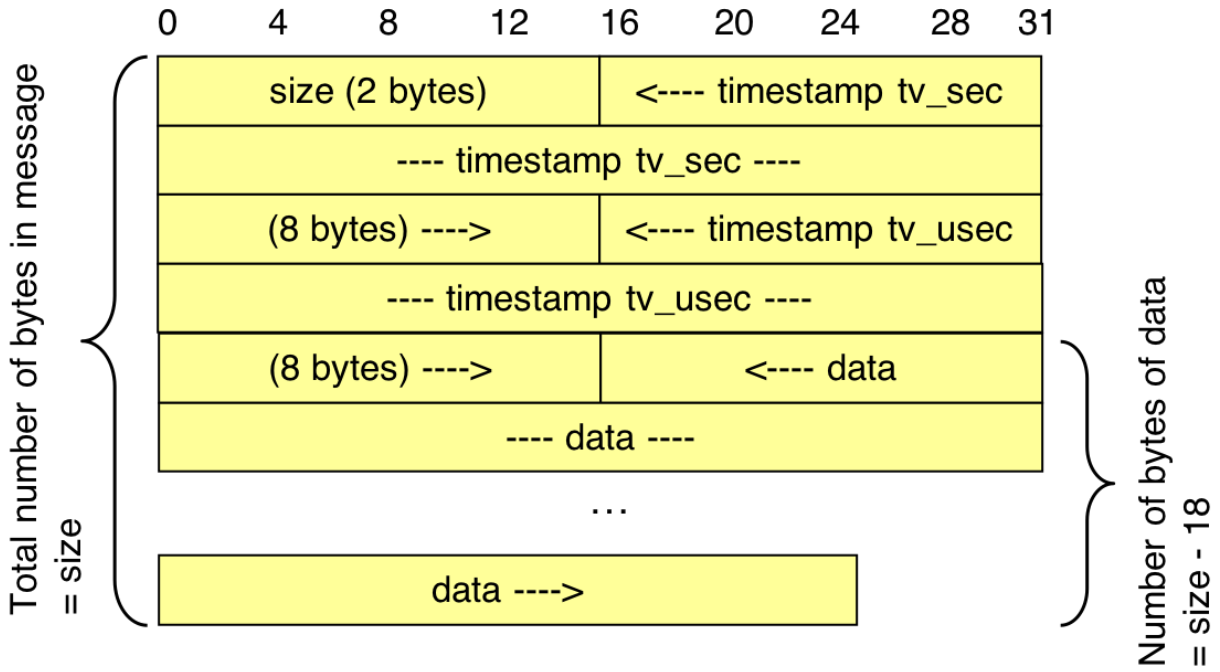
  The number of message exchanges to perform (1 <= count <= 10,000).

The ping-pong server program should take 1 command line parameter:

- *port*

  The port on which the server should run (on CLEAR, the usable range is 18000 <= port <= 18200).

The ping/pong message should be formatted as follows:



The size and the two timestamp fields are numeric values and therefore should be stored in network-byte order in the message. The data portion can contain arbitrary byte values, the default zeros in the allocated memory for the message is fine for the purpose of this project. The size lets the receiver of the message know how much data there is to read from the socket to completely receive the message.

The ping-pong server should be run first and wait for incoming SOCK_STREAM connections. The ping-pong client should initiate a SOCK_STREAM connection with the server. Once the connection is established, the client should send a ping message to the server. Upon receiving the complete ping message, the server should reply with that same message without modifying its content, a.k.a. a pong message, immediately. The client should receive the complete pong message. The client should measure the latency of each such message exchange (details below). The client should repeat such message exchange for "count" number of times, then close the connection. The client should printf() to the screen the average latency of the exchanges (expressed in millisecond, with up to 3 decimal places of precision) and then terminate.

In order to measure latency, the client should use gettimeofday() to obtain the current time before sending a ping message and put it as a timestamp in the ping message. gettimeofday() fills in a struct timeval, which consists of two 64-bit fields, tv_sec and tv_usec, the second and microsecond components of the time since 00:00:00 UTC, January 1, 1970. See "man gettimeofday" for details. Subsequently, when a pong message is received, the timestamp embedded in the pong message can be compared to the current time (again, use gettimeofday()) to compute the latency experienced by this ping-pong exchange.

Your server must be able to handle multiple concurrent connections from ping-pong clients.

III. Measurement - Mandatory for COMP/ELEC 556; Optional for COMP/ELEC 429 (15 points)

The ping-pong client/server measures the total latency of transferring the specified amount of data twice, once from the client to the server, and once from the server to the client. The data transfer latency measured can be decomposed into two components: (1) data size and bandwidth dependent transmission delay (which is approximated by data size divided by network link bandwidth), and (2) data size and bandwidth independent delay caused by the time it takes for data to physically propagate over the length of network links, the time overhead of making software function calls, etc.; this bandwidth independent delay exists no matter how high the underlying network link bandwidth is.

For this part of the assignment, design and write up a method for measuring this bandwidth independent delay across two different CLEAR servers as accurately as possible using only the ping-pong client/server software you have developed; no change to the software is allowed. Because you are limited by the ping-pong client/server's design, you will not be able to obtain a perfect answer; instead, the goal is to estimate the bandwidth independent delay as closely as possible. You should then

perform the corresponding measurements according to your method using your ping-pong client/server, and report your estimate of this bandwidth independent delay. Finally, use this estimated bandwidth independent delay to help you estimate the true bandwidth of the network link and report your answer. You must clearly show your measurement results and calculations in your write-up.

The grading of this part will depend on the method design and the underlying reasoning to support your method, as well as your actual measurements and calculations. That means your grade heavily depends on the write-up. Logical reasoning and precise wording in the write-up are expected. Your write-up document should be a PDF file named "part2.pdf".

IV. Administrivia

Failure to adhere to the following rules may result in a zero grade for your project.

You may form a group of up to 4 students to work on this project. If you need assistance finding group-mates, email the instructor as soon as possible.

You are allowed to use the sample C programs and Makefile provided by the instructor as starting points for your project.

To facilitate the grading process you must put your work under a directory called project1, then create a tar archive of it (i.e. tar cvzf project1.tar.gz project1). Submit the tar archive to Canvas. One submission per group is enough, unless there is a special situation such as members of a group want to submit on different days due to slip day usage differences.

In the project1 directory, in addition to any software code files, there must be a file entitled README which states the names of the students who worked together on the project, and documents how you tested your client and server, any known problems, and anything the graders should know. Also, include your "part2.pdf" document as appropriate. You must use *make* to build your project. There must be a Makefile in the project directory, and running make in the project directory must build both the client and the server. The grading will be done on CLEAR, so make sure that your code works on CLEAR. If your code does not compile with make on CLEAR, you will receive a zero grade.

You must write your programs in C and use the Linux networking API.