EECE4638: Homework I

Socket Programming

July 12, 2015

1 Introduction

The purpose of this homework is to get you familiar with the concept of socket programming and the way applications communicate over the network. You will also make observations into the behavior of the network by running experiments using the code that you will write in this exercise.

To start, download the material (code zip file) for this homework from the blackboard and extract it in your computer. You will see a few *cpp* files which are the base code that you will use and extend for this homework. The *Socket* class is the parent class that uses *"sys/socket.h"* from the system API and implements a series of easy to use C++ functions. These functions are inherited in *ServerSocket* and *ClientSocket* to implement client and server classes. There are two main files *simple_client_main.cpp* and *simple_server_main.cpp* that have the main functions. To build the code use the *make* tool inside the folder and the executables will be generated. To add new code files, edit Makefile (try to follow the syntax in case you are not familiar with writing such scripts).

After running the task below, prepare a short report and explain your observations. Submit your report along with your code by July 23 3.00PM.

2 Tasks

- Extend this code to implement an echo server (the server will reply to the client with the exact same message it has received)? We call this "direct echo" task.
- Add a new client code and name it *destination_client.cpp* similar to the first client. The first client will play the role of the source client as before except the server will have to relay the message from the source to the destination client. The destination must echo the message back to the server which will in turn relay the echo message to the source. We call this "relayed echo" task.
- For both of direct echo and relayed echo tasks, write code to measure the time it takes for the message to return to source after transmission.

Hint: you can use *"sys/time.h"* from system API. You can specifically use *gettimeofday* function to measure times in microseconds.

- Run your client(s) and server code on (2- for direct and 3- for relayed echo tasks) different computers in the COE network and record the echo times. At the same time, run *ping* command from the source to the destination. Compare the time values returned by ping with the ones from your program. What observations you can make? Which values are greater? Can you explain why? Hint: You *ifconfig* command in your Linux terminal to find out the IP address of your ethernet interface on each computer. You will specifically need the IP of the machine for your server code.
- *ServerSocket* and *ClientSocket* provide the platform for **Stream Sockets** which use TCP protocol (consult textbook for details). Write code for UDP protocol as well. Repeat the above experiments with the UDP and compare with TCP and ping results. Explain your observations.