

Programming Assignment 1

Assigned: Sep. 18

Due: Sep. 25, 11:59:59 PM.

1 Introduction

In this assignment you will write the server program which will communicate using sockets with the client program in the previous assignment.

2 The Protocol

Your server will run on the `linuxlab` cluster machines and will listen on a TCP socket bound to a port as described below. Note that you cannot bind to a port below 1024 without having superuser (`root`) access.

Given that your class account login id is `cs4170xx`, the ports you should use are `10xx0-10xx9` (inclusive). Thus, if your login id is `cs417060`, you will use port range `10600-10609`. Your final project must work with *any* port, but when *you* are testing your project you should only use the ports you have been allocated to avoid collision with others.

The rest of the protocol is as described in Project 0.

3 The server program

The command line syntax for a minimal server is given below. The server will take the host name and port as arguments (you can re-use the argument parsing code in the client).

```
server [<hostname>[ <port>]]
```

- The cookie should be generated using the formula:
 $(a + b + c + d) \times 13 \pmod{1111}$, where $0 \leq a, b, c, d \leq 255$ and `a.b.c.d` is the IP address of the client.
- After the successful communication, the server MUST print the cookie it generates along with the client's login id, first name, IP address and port number. All this information should be in a single line. An example is:
`555 cs417000 Kan-Leung from 128.8.126.208:48542`
- *Note well:* your server should not accept spurious input from the clients.
- We will test your server with non-conforming clients; the server should print out an error message containing the client's IP address and port number also in a single line, as such:
`**Error** from 128.8.126.133:48522`
and immediately `close` the connection when it finds a bad message from the client. Bad messages, as per project 0, are ones that have an incorrect magic string, incorrect message type or too many fields. Remember, the cookie sent in the STATUS message has to match the cookie in the CLIENT_BYE message for a communication to be successful.
- Do NOT print out any other debugging messages. They are useful for you, but not for your TA to grade.
- All output should be printed to `stdout`.

4 Requirements

- We will provide the source code for a conforming client for those who did not get the client to work.
- You will be building on these programs for subsequent stages of the term project, so it is in your own best interest to make them maintainable.
- Your code must be `-Wall` clean on `gcc`. Do not ask the TA for help on (or post to the newsgroup) code that is not `-Wall` clean unless getting rid of the warning is what the problem is in the first place.

5 Project Submission

- Please submit your code to the Submit Server (<https://submit.cs.umd.edu/>).
- You should upload a zip file which contains (at least) the following files:
 - `server.c`
 - `common.h`