

Programming Assignment 1

Assigned: 02/05/2009

Due: 02/12/2009, 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 server program must work with *any* port, but when you are *testing* your server program you should only use the ports you have been allocated to avoid collision with others.

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

3 The server program

The command line syntax for a minimal server is given below. The server will take the port number as an argument. If the port number is not specified, the server will listen to `SERVER_PORT` instead.

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

- The cookies should be generated randomly, in the range from 1 to 1000 (inclusive).
- After successful communications (i.e., all messages are exchanged correctly according to the protocol), the server **MUST** print the client's login ID, last name, the two cookies it generates, client's IP address and port number, and the comment. All the information must be in a single line. An example:
`cs417001 Huang 564 383 128.8.126.208:48542 See you later!`
- *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 assignment 0, are ones that have an incorrect magic string, incorrect message type or too many fields, etc. Remember, you also have to check whether the sum of cookies in client's `BYE` message is correct.
- 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.
- No sketch code will be provided this time. You can use any resources (e.g., books, Internet, etc) for documentation only. Note that copying code, even from the Internet or books, is *not* allowed.
- Your server must be able to handle multiple simultaneous connections.
- 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.
- The TA will answer general questions/confusions only, and is not supposed to debug for you. The “this is my code, what could be the problem” type of questions will be ignored.

5 Submission

- Please submit your code as described in the forum.
- What to turn in: All of your source code along with a Makefile. Your Makefile must produce an executable named `server` after a single `make` command is applied.