CIS527 Programming Assignment 1

Professor Jinhua Guo

Fall 2003

Due Tuesday, October 7, 2003

1. Introduction

The programming assignments will require heavy use of the Berkeley UNIX Networking facilities (often called **The BSD Socket Interface**).

In this assignment, you will use the socket interface to implement a simple, Internet chat room services. Your implementation will consist of both a client and a server application. The client can allow users to login, send messages to other active users, and receive messages from other active users, logout, and quit the chat room. The server verifies the identity of the users, forwards the message to the destination, and monitors clients' activities.

2. The Assignment 1

You will write two programs, a server and a client. The server creates a socket in the Internet domain bound to port SERVER_PORT (a constant you should define in both programs, you may use last 4 digits of your UM-ID or your SSN). The server receives requests through this socket, acts on those requests, and returns the results to the requester. The client will also create a socket in the Internet domain, send requests to the SERVER_PORT of a computer specified on the **command-line**, and receive responses through this socket from a server.

For this assignment, you just need to allow one active client connect to the server. In the next assignment (bonus), you must allow multiple clients connect the server at the same time. You will need implement the following command on the client side and the corresponding function on the server side.

1. login [UserID] [Password]

   Identify yourself to the remote talk server. If the UserID is not specified, the client will prompt the user for it. If the Password is not specified, the client will prompt the user for it. If the UserID and the Password match, the server will send a confirmation message to the client; otherwise, the server will send an error message to the client.

2. send

Send a message to the remote server. When a server receives a message, it will reply the client with the same message preceded by the UserID.

3. New

Create a new user account. The length of both the UserID and Password should be between 3 and 8 characters. The users' information should be kept in a permanent media, such as a file.

4. logout

Logout from the remote server. A user is not allowed to send any message after logout.

5. quit

Terminate the talk session with the remote server. The client exits when it receives the confirmation message from the server.

3. Programming Environment

You can use either C/C++ or Java to implement the assignments. The assignments will be tested on the Sun workstations in 1180EC. For easy grading, please don't use any GUI interface.

4. Requirements

The following items are required for full-credit:

- implement all five commands: login, send, new, logout, and quit
- the users information should be maintained by the server. You must have the following users (lower case) in your system:

UserID	Password
john	john
david	david
jennifer	jennifer
mary	mary

- a user can only be able to send the message to the server after its identity has been verified by the server.
- both the server and client should be able to run on any Sun workstations.
- the client should be able to connect to the server running on any machine. Therefore, the server IP address should be a command line parameter.
- the server should output all client activities on the screen.

- when the previous client exits, the server should allow the next client connect.
- your source codes must be commented
- include a **README** file in your submission.
- include a **Makefile** in your submission.

Note 1, in your README file, the following information should be included: the functions that have been implemented, the instructions about how to run your program, known bugs, and sample outputs.

Note 2, your Makefile might be the exact same as the sample Makefile if your file names are the same as those of the sample codes.

5. Grading (150 points)

- Correctness and Robustness (120 points)
  - You will lose at least 10 points for any bugs that cause the system crash.
  - You will lose 5 points for any other bugs.
- Comments and style (10 points)
- README and Makefile (10 points)
- Design and Performance (10 points)
  - Limit the communication between the client and the server. If something can be done locally at the client side, you should not ask the server to do it. For example, checking the validness of a user command should be done at the client side.
  - Don't make the client a dummy terminal.

6. Sample outputs

Sample client outputs, the words in **bold** are the user inputs.

```
cluster1:~/cis527/p1% clientTalk 141.215.10.123
Internet chat room: computer networks group
Here, list commands available for the users, and instructions.
Chat>send
Permission denied, you must login first.
chat>login
UserID: john
Password: john
SERVER# John joined the chat room
chat>send
Please input your message:
Hello, Everyone
SERVER#John: Hello, Everyone
chat>send
Please input your message:
```

```
Good Bye
SERVER#John: Good Bye
chat> quit
SERVER#John has been disconnected from the server.
cluster1:~/cis527/p1%
```

7. Submission Instruction

(1) Copy all your files (only source codes, data file, README, and Makefile, no object file and executable file) in a directory. The directory should be named like lastname_firstnameinitial_p1. For example, if you name is John Smith, your directory name should **smith_j_p1**.

(2) Generate a tar file of the directory using the following command.

Enter the parent directory of your current directory and type

```
% tar cvf lastname_firstnameinitial_p1.tar lastname_firstnameinitial_p1
```

For example

```
%tar cvf smith_j_p1.tar smith_j_p1
```

Note, only the tar file will be accepted. You are fully responsible for generating the right tar file.

(3) Email the tar file before 11:59PM, October 7, 2003 to me jinhua@engin.umd.umich.edu

8. Hints

Start early. These programs will not be very long, but they may be difficult to write, and they will certainly be difficult to debug.

Download the sample codes, compile and run them, try to understand every detail, and then make the changes.

Design your own communication protocols between the client and the server.

Watch the course webpage (http://www.engin.umd.umich.edu/~jinhua/fall03/cis527) for additional tips and information concerning the assignment.