CSE 3461 Project - Spring 2020

In project 1, students will implement an instant messaging (IM) platform. In project 2 they will implement a file sharing platform.

For these 2 projects, students are free to use any programming language that they feel comfortable using. The programs must run on STDLINUX. You will use STREAM in the first lab and DGRAM in the second lab.

Bonus (20 points): The bonus points are to implement a graphical user interface for the client side to connect to the back-end application. In this case, the users can type their messages (IM platform) or request the file through the GUI. You will get a maximum of 20 points total for the two projects. You will have to show the grader your GUI during his office hours.

Project 1: Instant Messaging Platform (100 points) – Use STREAM

List of features:

1. Multi-user IM platform: Server should be able to handle multiple connections from multiple clients.
2. Client registry: Server should register the clients with <username> and <password>. New clients should be able to register by sending a command including their desired <username> and <password>. If the <username> already exists, the server should send an appropriate error message. Server keeps track of the registered users by adding them to a **file** or database.
3. Online clients: Each client should be able to see other online clients.

The Client will be called ‘client’ and the server will be called ‘server’
The client must implement the following commands:

1. Register <username> <password> - if the user exists, the server responds saying user already exists.
2. Login <username> <password> - the server handles it properly if the password/username is wrong.
3. List – this lists all the users active on the server
4. Message <username> <msg> - this will send the message to the user specified.
5. Logout <username> - logs user out.

Project 2: File Sharing Platform (100 points) – use DGRAM

List of features:

1. Client sends the name of file that it wants to share and its connection information to the server.
2. The client then opens a socket and wait for requests from other peers for a file that it holds.
3. Server keeps track of the file names and corresponding connection information.
4. Upon a request for a file, server replies with the connection information of the client which holds the file.
5. The client will be called ftpc and the server will be called ftps

The client must implement the following commands:

- **Register** – this is when the client will tell the server that it is 'alive' (you can add a login if you want, but it is not required)
- **Share** "files" – this is when the client tells the server the list of files that it has available to share
- **List** – the user should be able to enter a 'list' command and see all the files that the server has access to
- **search** 'filename' – the client should be able to search for a specific file. You don't have to implement wildcard search
- **download** "filename" – the client tells the server it wants to download a file, the server tells it the location of that file, and then the client tries to download from that location. You should implement the ability to tell the user if the download location is not found, or does not respond. Ideally, the server will reply with multiple locations where the client can download a file, and if the first location does not respond, the client should try the second, etc.