Holly Chen z5359932

COMP3331 Assignment Report

Program Design

Language used is Python 3, written in TigerVNC. I implemented all the features: login and logout, blocking user for 10 seconds, successful login for multiple clients, implementation of /activeuser, /msgto, /creategroup, /joingroup, /groupmsg, displayed message, and /p2pvideo.

Application Layer Message Format

To send messages from client to server or server to client using TCP, I encoded the message strings into bytes, then decode it once it has been received. For sending messages (information) from client to server using UDP for the p2pvideo, I encoded the message in latin-1, since the video files are written in latin-1. I then decoded the messages in latin-1 when it reached the server side. In this message, I used a header to show the filename and sequence number of packet so that when it arrives on the server side, I could use the filename to write to the file and use the sequence number to make sure that all packets are assembled in the right order, then written to the file correctly. Since the packet size including header should contain maximum 1024 bytes as designed in my implementation, I used 950 bytes for the data in the packet, and 74 for the header.

How the system works

After running the server using python3 server.py <server port> <number of consecutive failed attempts>, the number of consecutive failed attempts will be stored on the server side. Then, run the clients using python3 client.py <server IP> <server port> <client udp server port> in different terminals and in different working directories. I have prepared the two working directories client1 and client2, which could be used for testing.

Authentication

After the client is running, the client will be prompted to input their username and password details. If the username and password details are in the credentials.txt file, then the client will be logged in successfully. If not, then the client will be prompted to try again, until the number of tries exceeds the allowed number of consecutive failed attempts specified on the server side. If it does exceed this amount, then the user will be blocked from logging in with the username. The client will terminate. After 10 seconds, the user can retry again and log back in. In that time, the client can log in with a different username without being blocked.

Commands

After logging in, the client will be prompted to input a command. Commands include: /msgto, /activeuser, /creategroup, /joingroup, /groupmsg, /logout, and /p2pvideo. To use any of these commands except /logout and /activeuser, there needs to be other active users that are connected to the server. /logout disconnects the client from the server, while /activeuser shows the active user information if there are any other active users, or prints a message that there are no other active users. To use /p2pvideo, the client must first call /activeuser so that the client knows who is active and who is not. To use /groupmsg, the client must be joined in that group already. To use /joingroup, the client must be invited to the group by another client who used /creategroup. After calling a command, the corresponding response will happen, and the client is then prompted to type in another command, until the client logs out. When the client logs out, the server gets rid of the client in the active userlog.

Design Tradeoffs Considered

A design tradeoff considered was the use of time.sleep in my client.py for p2pvideo sending. I implemented this since the packets were not receiving on the server side without it. This is

a design tradeoff, since it takes longer for the server side to receive the packets than without time.sleep. It is especially noticeable if the video file size is relatively large.

Improvements

Improve the implementation of p2pvideo by not using time.sleep, using retransmission instead, by communicating NAKS back to the client side for p2pvideo if a packet has been lost.

Code References

Referenced the assignment starter code given for both the server and client python file. Additionally, referenced the multithreading code in the programming tutorial week 7 to do the multithreading in my client side.