COMP3331 Computer Networks and Applications

Assignment Report

Victor Tian; z5245580; Nov 17, 2021

Table of Contents

General

This project implements a simple instant communication software. It contains two major sections, the server, and the client. There will only be one active server for each instance of the software and there may be an unlimited number of clients active on the server. This allows users to communicate via the server, block and unblock a particular user, and start private messaging. However, tests on significant number of clients needs to be done in the future. Multithreading have been implemented to ensure sufficient communication.

Execution

- The execution of the program should be as follows:
  - Server : python3.7 server.py server_port block_duration timeout
  - Client : python3.7 client.py server_port
- Dependencies and Libraires:
  - server.py :
    - socket
    - threading
    - sys
    - datetime
    - time
    - signal
  - client.py :
    - socket
    - threading
    - sys
    - threading
    - readline
    - time
- The server must start before the start of any clients. Once the server starts, it listens to any connections at the specified server_port. If a client connects to the server_port, the server will prompt the user to enter credentials and validates them. Once validated, a secure connection would be created, and the server would process

specified operation as in the given example. The server would log a user out if the user has been inactive for over the specified timeout period and the server would block the user from logging in if the user had multiple failure logins.

Program Structure

- Server Side:
  - server.py: contains all server functions
  - credentials.txt: contains all username and password pairs for authentication
- Client Side
  - Client.py: contains all client side functions

Application Layer Design

- Message will be communicated in simple text format. Transmitting messages in text format provides code simplicity on both the server and the client. However, sending messages in text would scarifies message security as the message might be read by hacker during transmission.
- The loginBlockedList, peerList, activityList, messageToSend, and userBlockedList, which keeps track of different types of user data for different functionalities, would be stored dynamically in the server with no backups and data would be lost once the server have been restarted, detailed storage format can be seen in the corresponding comments in server.py or client.py.

Design Trade-off

- Timeout functionalities have been implemented on the server for simplicity. Hence, to implement the timeout functionality for private connections, the client would then require sending a special flag to the server when the client sends a private message or stopped private messaging.
- Multithreads have been used on both the server side and the client side. The server uses multithreading to handle multiple clients at the same time. The client uses multithreading to ensure the I/O will not be blocked by any of the processes, i.e., the system must wait for a user input before printing out the received messages.
- Messages have been implemented to be transmitted in the simple text format. This allows the most design and code simplicity and would make the code easier to maintain. However, this might result in potential security issue as sensitive data and messages will be transmitted in text format and hackers might intercept and read them during transmission.
- The system will ask if a user is willing to accept private messaging with client. We assume that the request to the user from client about whether he wants to start p2p with client happens through p2p communication socket right after p2p begins. The user would return ‘y’ or ‘n’ as soon as he can. The client will not send message to the user by p2p until he receive the confirmation from user about accepting p2p communication and the user will not receive any message or notification after the request popped out before the user replies.

Possible Improvement

- Json might be used to transmit data instead of plain text. Furthermore, data might be encrypted before transmission to provide maximum data integrity and security.

- Data might be made persistent, i.e., userBlockedList, which keeps track of if a userA have been blocked by any other users, might be stored in a file instead of dynamic memory of the program and can be persistent after the shutdown of the server.

Known Bugs

- The entire project has been written on my local computer under python3.9.1 conditions and have been tested on UNSW VLAB environment by command line through terminal window under python 3.7.
- There are no significant known bugs of the program
- A user might be unable to run the server in a particular port. This is because the port is currently occupied by other processes and the user may change to another port to solve this problem.

Reference

- The entire assignment code has been developed based on the provided starter code by cs3331 teaching team
- The safe_print function which ensures printing incoming messages to a client will not break any threads such as the input and p2p connections was referenced from https://stackoverflow.com/questions/2082387/reading-input-from-raw-input-without-having-the-prompt-overwritten-by-other-th/4653306#4653306