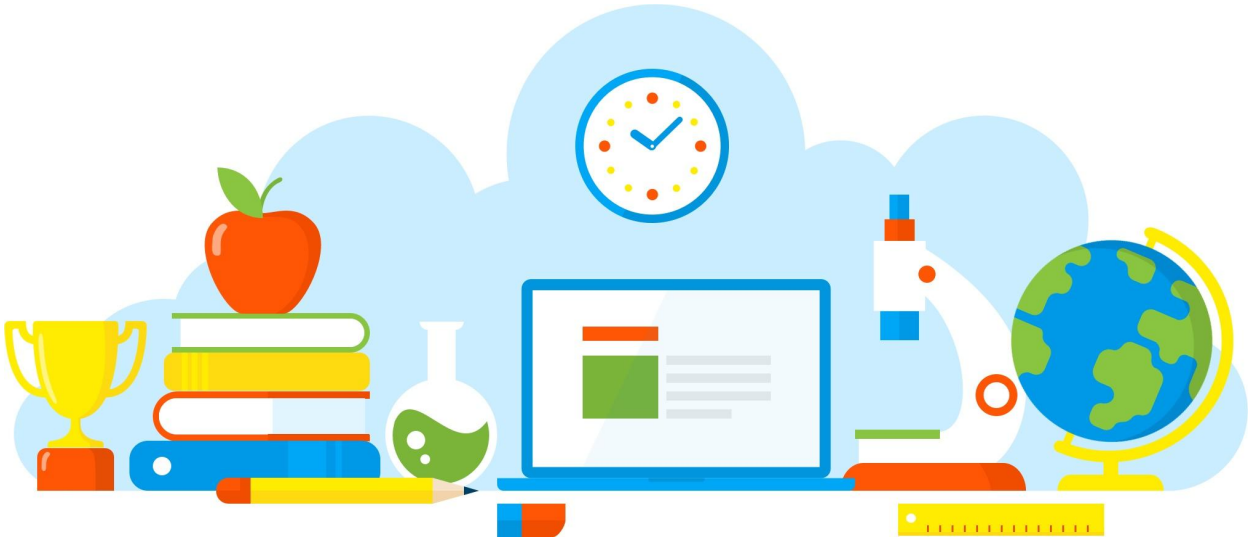

Project Report

Lorem ipsum dolor sit amet, consectetuer adipiscing elit

Zijian Yue
15/04/2021

Introduction

The application will be designed to support a range of functions typically found on videoconferencing, including authentication, posting text message to all participants or one particular participant, uploading video streams.

The limitation of the program

1.The program will not allow one user to log on to the server on different machines or terminals.

2. In this application, it is assumed that a user will always input the correct username.

3. Inconsistent format might occur a file is received when a user is typing/issuing a command. (spec).

4. It is assumed that clients are on the earth and have a bandwidth of internet larger than 2Mbps.

5. The program will not be able to handle the case where the client sends files with the same name.

6.Note: due to the transmission security measures, it takes UPD about 1 to 2 minutes to transfer example2.MP4(4.2MB).

The Architecture and workflow of the system

Server: The server of the application is designed to be able to interact with multiple clients simultaneously. This design is achieved by a trivial implementation that the server will start a new thread for that client if one client is authenticated by the server. Within that thread, the server will provide six functions(MSG, DLT, EDT, RDM, ATU, OUT) for the client to call. As for the situation of racing between the clients, the program will use python thread lock as a solution.

Client: The client of the application is designed to be able to achieve two goals. First, it will be able to connect and interchange information with the server. Second, it needs to be able to receive and send UDP packages to other client programs. So the client's workflow is as such if a client passes the server's authentication, it will initiate two threads. One of the threads is used to communicate with the server. The other one is a UDP receiver that keeps listening to the possible UDP transmission from other clients. It will also write the transmitted file to the client's current working directory until the client logs out.

Specific designs

1. The server will use bunches of list of lists to collect three fields of information.

I. The failed attempt of users, II. The active users, III. The message sent by users.

2. All the specific time in the program are obtained by the python time module.

3.The server uses python thread lock to lock some necessary procedure(eg. Update the list of received messages) to make sure only one thread is fixing the data storage and write the logfile at the time.

4.P2P secure measures

1. Before the sender sends the file to the receiver, it will keep sending a handshaking request package to the receiver until the server eventually received it. Then the receiver will create a list for the sender to store the content and send an ack for handshaking back to the sender (please check the comments in client.py).

2. After the sender received the handshaking ack, it will start to send packages using GO-BY-N design, where the sender will send 100 packages every time to the receiver. After the sender sends 100 packages, it will wait for 1 second to receive the newest ack from the receiver. Then the sender will use the new ack as the base case to send another 100 packages to the receiver.

3. After the sender sends all the packages to the receiver, it will keep sending a signal to the receiver to inform the receiver that the transmission has been done until it receives an ack from the receiver. After the receiver received the signal, it will write the stored bytes to a file and then send it to the signal back to the sender, so the sender could acknowledge that the transmission has been a success.

4.The size of the UDP package is set to be 1027 bytes(includes the header) which can be seen as a secure size of the UDP packet. The window size is set to 100 packages.

Type of application message format

This section will provide a list of formats of the command used in TCP transmission and UPD transmission.

For TCP, This 'UTF-8' encoded format is used by clients to send to the server.

<username>;<command>;<arg of the command n>;<arg of the command n + 1>;...

The chunks are separated by semi-colons, the first chunk is always the username, and the second chunk is always the command. The chunks after chunk two are the arguments of the specific command(please check the comments in the code).

The TCP server has no fixed format for sending information back to the client. Please check the comments in the code.

For UDP, This bytes format is used in the interaction between the clients.

<byte0> + <bytes1 ~ 2> + <bytes 3 ~>

Byte 0 is used to determine the package type (please check the comments in the code).

Byte 1 and byte 2 are used to determine the ack values or the sender's listening port.

Bytes after byte3(include byte 3) are used to contain 1024 bytes of data.

This gives a total size of 1 + 2 + 1024 = 1027 bytes of a package size.

Possible improvements

1. Some part of the program can be re-written in a more pythonic way, improving both readabilities and efficiency of the program. This can be achieved after the programmer(author in this case) have done more training on python programming.

2. The program can be re-designed to a more object-oriented centred program. By doing so, the portability and maintainability of the program will be improved. This can be achieved after the programmer(author in this case) has done more training on object-oriented design.

3. Currently, the P2P implementation is using a trivial variation of the GO-BY-N design. This can be improved to increase the speed of P2P transmission.

4. The data should be stored in a database, not in the memory or a text file. Use SQL to store the data.

5. This application cannot detect malware and virus, it might result in damaging the computer, It indicates that the python virus detection component should be used.

Design tradeoffs

1. As this is an individual assignment, some easy implementation(idea) is used to implement the program, which will reduce the difficulties of programming and debugging. However, such implementation will decrease the program's efficiency. The machine will also need to allocate more memory to run the program.

2.For the P2P implementation, due to the usage of UDP protocol which can not be considered a secure way for transferring information. Therefore some security measures had to be taken by the program(application). These security measures will guarantee that the file will eventually be transmitted to the destination. However, these measures also slow down the transmission of the content.

3. To prevent the log file's corruption, some of the data are stored in memory and provide security to the program; however, the machine will need to allocate more memory, making the application less scalable.

Declaration of the usage of external resources

The program(both client and server) contains no external source code. However, it is worth noting that some programming techniques(such as python multi-threading TCP server design, the use of t_lock to solve the racing problem, the method of reading binary content from a file) were learned from YouTube, StackOverflow as well as Geeks for Geeks.