Project Title: A ClassChat System

October 27, 2020

Instructor: Xu Yuan
Deadline: November 21, 2020

The objective of this project is to design and develop an online chat system, named ClassChat, to be used for communications and discussions among students in a class. The ClassChat is to offer a software platform including the function of enabling students to chat with Instructor and other students for any necessary disccusions, i.e., homework problem. We give guidelines for the system design of ClassChat.

1 Client–Server Communication using TCP/IP

The first step is to build a server and a client that can communicate with each other.

1.1 Server

A server should be developed as a central controlling point, which can offer resource and services per clients' request. Here, the server should have a core function of interconnecting the communication of two clients. That is, if a client A wishes to initial a chat with another client b, Both A and B should connect to server and the server can help forward messages or requests between A and B. To implement a server, the following steps have to be implemented:

- Create a socket for communication
- Bind the local port and connection address
- Configure TCP protocol with port number
- Listen for client connection
- Accept connection from client
- Send Acknowledgment
- Receive message from client
- Send message to client

1.2 Client

To implement a client, the following steps have to be implemented:

- Create a socket for communication
- Configure TCP protocol with IP address of server and port number
- Connect with server through socket
- Wait for acknowledgement from server
- Send message to the server
- Receive message from server

The sketch of client-server communication through socket programming using TCP/IP is shown in Fig .1.

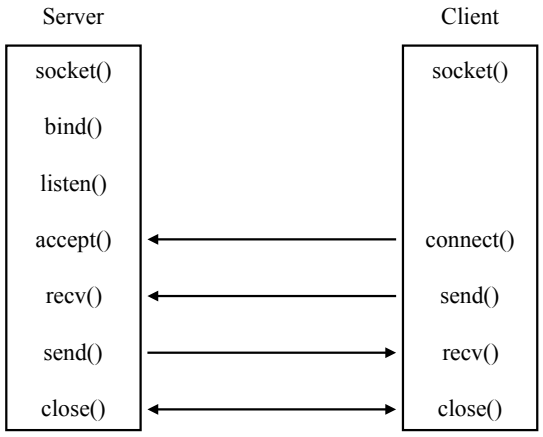


Figure 1: Client-server communication using TCP/IP

1.3 Requirement

Now, you can implement the communications of a client and a server. In the implementation, GUI is not required but highly encouraged. It is fine if you just use command line to let client and server to communicate with each other. This system can be written with any language, but I strongly recommend Python and Java since they offer convenient tools for socket programming. This task takes 30 points.

2 Advanced Client

After you have implemented the simple client-server communication system, now you can add more function that a client can both send and receive message at the same time with less CPU workload. I/O multiplexing can be used in this task, in which you can use system callback function to activate a client's application. That is, a client will be activated if the socket receives data from the server or keyboard input from the user.

Hint: try to use select(), poll() and epoll() in your client.

This part takes 20 points.

3 Multi-Thread Communication Server

Now we would like to improve our client-server communication system by developing a network server to handle multiple concurrent problem. The goal of this task is to allow multiple students to discuss class topics or homework problem at the same time.

There are 3 common ways to implement such function in a server:

- Use socketserver model.
- Thread + socket
- I/O multiplexing

You can select any one method to implement your server. Notably, in the first method, Python has provided *socketserver* package to simplify the process of building a network server. Til now, your server should be able to support connections with multiple clients at the same time. This part takes 20 points.

4 Client-Client Communication

Now, we are ready to implement the client to client communication in ClassChat. Your ClassChat should have three core functions:

- Client management.
- Receive message from a sending client.
- Forward message to a receiving client.

The client management is very important at a server. The server should be able to capture the exception that if a receiver is not in the system. For example, A wishes to chat with B, but B is not in the system. Such practice issue should be considered to make your system much robust. Fig. 2 illustrates the main function of ClassChat server. Clients can communicate with each other by passing messages through Class-Chat server. Fig. 3 shows a demo of ClassChat. In this figure, Alice and Bob first create connections with the server. Then, Alice sends a message to Bob, the server receives this message and forwards it to Bob.

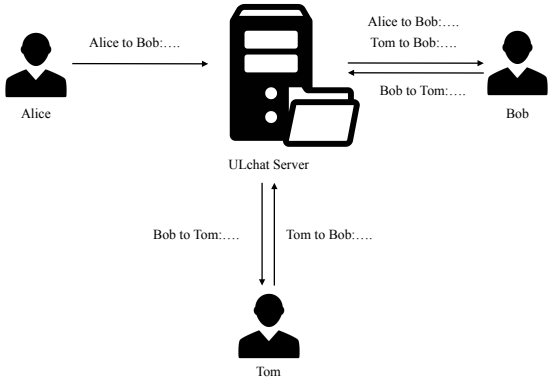
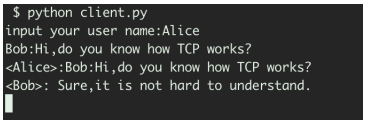
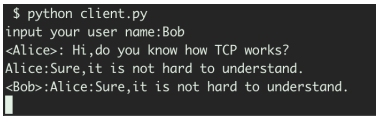


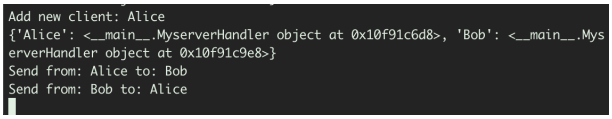
Figure 2: An Flowchart of ClassChat.



(a) Client window for Alice



(b) Client window for Bob



(c) Server window

Figure 3: ClassChat demo.

In your client, you should be able to capture sender information, receiver information, and text messages. JSON is a standard format that can be used in network transmission. In this project, a simple example like:

```
{
    "status": "1",
    "sender": "Alice",
    "receiver": "Bob",
    "text": "Hi, do you know how TCP works?",
}
```

This task takes 30 points.

```
input your user name:Alice
<Dr.Yuan>: I will give another 20 points for those who solves this problem
<Bob>: that's great
<Tom>: yeah!!!
```

Figure 4: An example of group chatting

5 Bonus

The following tasks aim to enhance ClassChat's function. You can get 10 points bonus for each one.

5.1 Group Chatting

This group chatting is important for group discussion or information announcement. The core function here is that each group member can send and receive messages in the group chat window, while these messages can be visible to all group members.

For example, Instructor would like to create a group including all students, so all important information can be broadcast in the group and received by all students. And also, if any student has some question, they can rise in the group. Instructor and all students can saw this question and give replies. An example is shown in Fig. 4.

5.2 File Transfer

We can improve ClassChat's function by supporting file transferring between clients. That is, one client can transfer a file to another client by using ClassChat.

5.3 Off-line Message

Now, we enable ClassChat to have the function of receiving offline message. For example, Instructor assigns a project to the class through ClassChat, but some students may not be online (not connect to server). In order to recover this offline message, we should create a powerful server that can save the message for off-line clients. Once these clients connected to the server, the stored message can be forwarded to them.

5.4 Encryption/Decryption Between Client and Servere

ClassChat server IP address and port are exposed in the network. The transmitted messages have the potential risk of being captured by Wireshark. So we would like to enhance ClassChat's security by adding encryption and decryption to the transmitted messages. Please come up with a security model for ClassChat. Hint: use the public and private key to generate and transmit session key.

6 Requirement

The total points of this project is 100 + 40 (Bonus). Please add makefile and readme file to explain how to run your code. Please add screen shot for your demo for each section and give a technique report to explain your implementation.