COMP3331/9331 Computer Networks and Applications

Report of assignment

Yuan Gao

z5239220

Introduction

> There are six python files used to implement the client and server programs of a discussion forum application
>
> ```
> Client.py
> Server.py
> Command.py
> FileManagement.py
> FTclient.py
> FTserver.py
> ```
>
> The Operations of forum application including:
>
> Store user name and password, Register a new user, Create Thread, Post Message, Delete Message, Edit Message,
>
> List Threads, Read Thread, Upload file, Download file, Remove Thread, Exit, Shutdown
>
> The running envrionment:
>
> python 3.7

Program design

> - `Client.py` and `Server.py`
>
>> These two python files are main file of this forum application. `Server.py` provies the Server service and `Client.py` provies the Client service.
>>
>> This forum application uses TCP protocol. I seprate two part to transfer data. Firstly, clients need to be authenticated. When server recive correct command, client can log in forum (create thier subthread) and usign the forum operations.
>>
>> In Server, I use `list` to store subthread ID number. Every ID number indicates a client user. The main thread can recive message from every client user. Sever can send meesge to every client user.

> > In Client, there are two threads. One is used to receive data, the other is used to send data.
>
> - `FTclient.py` and `FTserver.py`
>
> > These two python files are designed by transfer files. I try to build function in the `Server.py` and `Client.py` but failed. Therefore, I build extra files to achieve `UPD` and `DWN` command. I also ues TCP protocol to complete transmission function. It is worth mentioning that those function occupied `5745` port.
>
> - `FileManagement.py`
>
> > This file is used to implement forum management, which include registering a new user, creating new thread, reading the forum thread, write massage and edit massage.
>
> - `Command.py`
>
> > This file is used to implement forum operations. This file encapsulates the 11 commands of this forum. We can use the class interface in `Server.py`

Application layer message format

> I use TCP protocol to complete this application. At first, the message formats just the simiple `encode()` and `decode()`. However, a piece of message is difficult to process some specific instructions. Therefore, I use `json` method encapsulates message. The benefit of `json` is that it can easily handle the content of the message.
>
> For this part, I made a serious mistake at the beginning. Receiving and sending need to be performed alternately and cannot receive or send data twice or more consecutively. Continuous sending and receiving will cause the program to crash.

How to work

> `FTclient.py`, `FTserver.py` and `Client.py` should be in the same folder.
>
> `FileManagement.py`, `Command.py`, `FTclient.py`, `FTserver.py` , `credentials.txt` and `Server.py` should be in the same folder.
>
> For the Server:
>
> `python3 Server.py portNumber adminPassword`
>
> For the Client:
>
> `python3 Client.py IPaddress protNumber`

Progress and improvement

> - Show '*' instead of character when entering password
> - I can add a progress bar during file transfer which let user know the file transfer progress.
> - Solve the problem of file transfer occupying additional ports

> - Add thread lock. Ensure that there are no conflicts in file modifications

References

> The `FTclient.py` and `FTserver.py` refers to the URL:
>
> https://blog.csdn.net/qq_31063727/article/details/89740190?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-7.channel_param&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-7.channel_param