ASSIGNMENT COMP3331 REPORT

Program Design and tradeoffs:

Language: Using pyhton3.7 CSE environment

Outline:

Client has many utility function to take different part of work, setup udp sever, login, logout ,download tempid , send beacon, upload contact log.

Sever use thread and check the message type in the dictionary to determine work.

Use json to transmit dictionaries data and use many global variables.

Initial Part:

Server:

Server will be run ip get by os and port that type in and will block user if three wrong password for the time that type.

Have some global dictionaries, list, variable.

blockingdic will store the user that is blocking. Key is the username and value blocked time.

blockdic will store the user that login failure. key is username and value is the number of login failure

blocktime is the time the user will be blocked. It will be update at the start of server.

loginuser is the list that the user has log in.

tempidlist is the list that existed tempid.

It will get server port (sys.argv[1]) and blocking time (sys.argv[2]) from command line.

And set global blocktime as the blocking time which get from command line. (sys.argv[2])

When server create, it will create a new empty tempIDs.txt file for future use.

It will start a new thread for new one connected and print the new connection message on server terminal `('Connected to :', addr[0], ':', addr[1])`. Then each thread will check the payload and check messtype to know which request it want. Then response it, waiting for next request. If someone disconnect, print some message on terminal.

Client:

Client will create a TCP connection to sever IP and port. And create a UDP server at UDP port.

Have a global dic:

tempid store the tempid from server. key is tempid, createtime, expiredtime and store relevant information from server.

At first, it will create TCP connection to server and get the username and password from terminal to login. When user has login successful, it will start a UDP server on client ip from os and start a new thread for bind beacon from other client. And create a new empty z5223797_contactlog.txt whether have a older one And waiting for terminal next command. For different command, it will go to different utility function for better repair and style. If command is unreadable, will print error message.

TCP part:

Login part:

Client has a unity function loginserver which need username and password and clientSocket. Client will transmit dictionaries via json and send to server.

```
sentence = {
    'messagetype':'credentials',
    'username': username,
    'password': password
}
```

According the reply message from server print different information.

Server will check message type and do credentials check. And reply the different message according to the username whether matching password in the local file credentials.txt.

Sever reply	Client Display	Description
“Y”	“> Welcome to the BlueTrace Simulator”	Succeed log in. Add this user to loginuser.
“N”	“> Invalid Passsword. Please try again”	Username does not match password. Server start count the number of times the user login failure.
“NB”	“> Invalid Passsword. Your account has been blocked. Please try again later”	the number of times the user login failure is equal 3. Blocking this user for <blocking time>.
“B”	“> Your account is blocked due to multiple login failures. Please try again later”	Username has been block by server. Wait the time exceed the time been block plus <blocking time>.

Download_tempID part:

```
sentence = {
    'messagetype':'downloadtempid',
    'username': username
}
```

Client has a unity function download which need username and clientSocket which send dictionaries to the server. And client will print the current tempID when it reveiced and update the gloabal dictionary tempid.

Server will check message type and generate random 20 digit as tempID for this user. And record the create time and expired time in tempIDs.txt. In format for example (space split)

“38958736185989944548 04/08/2020 19:05:37 04/08/2020 19:20:37” on each line.

And send to client in format like(“,”split)

“38958736185989944548,04/08/2020 19:05:37,04/08/2020 19:20:37”

There will be (and should be) multiple tempIDs associated with one username since duration of the tempID is 15 minutes only and username is forever. Print username and tempid on the terminal. And make sure they won't have same tempid at same time via list.

Upload contact_log part:

```
sentence = {
    'messagetype':'uploadcontactlog',
    'username': username,
    'log': log
}
```


Client has a unity function upload which need username and clientSocket which send dictionaries to the server. If contact log is empty or user has not get a tempID, it will print error message. Client will open the <z5223796_contactlog.txt> and send the information to server in dictionaries Log. Log is a dictionaries which key is tempid and value is time dictionaries. Time has two key createtime and expiredtime which value is the string of the time.

Server will check message type, print the received contact log from client on terminal and put contact log in a dictionaries then matching the tempID.txt find phone number of user and print it. If a user in contact log appear many times, only print once phone number in contact checking stage.

Logout part:

```
sentence = {
    'messagetype':'logout',
    'username': username
}
```

Client has a unity function logout which need username and clientSocket which send dictionaries to the server. And will print a exit message then close the connect and exit program.

Server will check message type, remove user from loginuser list. Print user logout message on terminal.

P2P(UDP) part：

Client will have three unity function udpserver, revbeacon, sendbeacon. To do the P2P part work.

UDP server part:

When client has successful login, UDP server will be run on udp port which get from begin. And for each new client connected in, UDP server will open a thread revbeacon for them to revbeacon information from other client.

Revbeacon part:

If beacon information been sent to the UDP server, it will print the received message on terminal and check whether valid according to the time. If valid, print valid message and store in z5223796_contactlog.txt in format looks like

“3895873618598994454 8 04/08/2020 19:05:37 04/08/2020 19:20:37”

And this beacon information will be delete after 3 minutes by timer thread helper function dellog. Dellog will delete the specified string in z5223796_contactlog.txt.

If is invalid, the server will print invalid message and discard this beacon message.

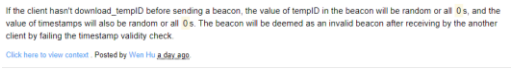
To be noticed, log file should be empty (the file does not exist) when the client starts as Wen Hu said.

Sendbeacon part:

Client has a unity function sendbeacon which need the udpip and udport that type in. And will send the beacon information to this udpip and udport in format like

“1,tempid,creatime,expiredtime” and print this message.1 the version of protocol.

If no tempid, it will send tempid all 0 and timestamp all 0 as WenHu said.



Problem:

If client use ctrl + c to stop program, it will not delete the received beacon when it exceed 3 minutes and server cannot know which user disconnect and also update the userlist.

When we typing command and we received beacon message it may looks mess. Miss ‘> ’after receviced beacon message.

If IP is not type right will raise error.

Improvement and extensions:

Improve data structdt and fixed format on request and response. Like more reasonable tempid dictionaries and response header. [response state][payload]

Use tempid as the key to do other command will be safer and closer really app in fact.

Return a tempID when user login is more reasonable for app using.

Maybe add login user check is a good way to prevent the hasker user.

Download tempid and save them as a file so they can use it when reopen the app.

Refence:

Get idea from sample code thread server TCP python from https://www.geeksforgeeks.org/socket-programming-multi-threading-python/

Sample output: