COMP3331 Assignment

z5253838

Manh Truong Nguyen

The program works fine on vlab machine or java version "1.8.0_231" Java(TM) SE Runtime Environment (build 1.8.0_231-b11).

I mainly refer from java document and stackoverflow, there are one or two small functions that I borrow.

The program architecture

The server has two parts Client listener and Client Handler:

- Client listener run in a while loop, wait for the client to connect. When the connection is establishment, start a new parallel thread to handle the client (Client Handler).
- Client Handler run in a different thread and created whenever a client connects to the server. Client Handler work as a state machine, when It receive protocol from client, it checks the protocol header which triggers the change of the state for handling data.

The Client Handler has 4 main states:

- "auth": receive username, password then verify it and send response back to the client. In order to keep track of login time and prevent login after 3 failures. I create a hashmap which have phone number as the key and AuthObject as the value, AuthObject will hold information about login time (number of login times, last time login).
- "upload": receive data from client and display logs.
- "download": Generate a new tempID.
- "Beacon": Display the beacon which sent by the client
- "Exit": logout.

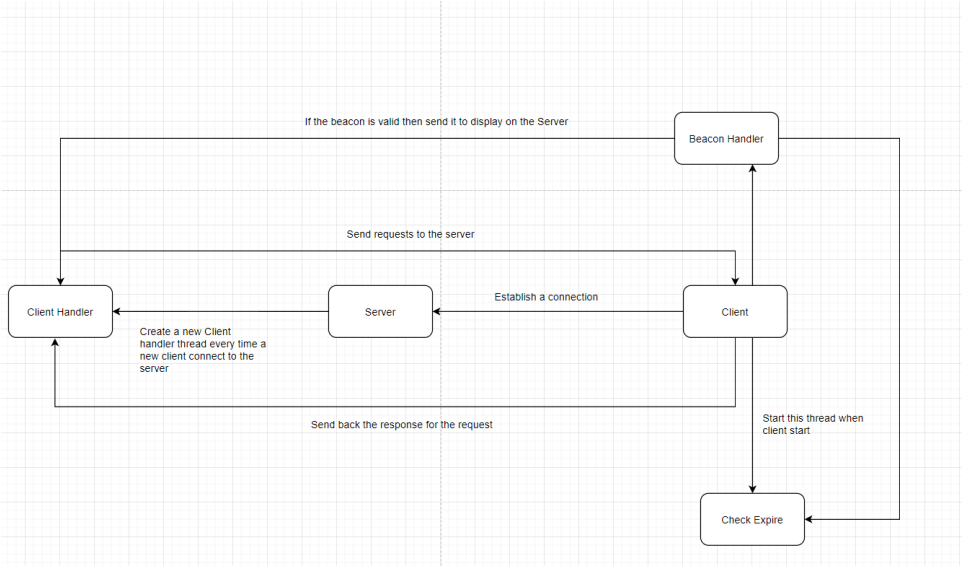
The client is a state machine which is like Client Handler ("auth", "upload","download","Beacon","Exit"), it takes command from user and change it state to handle the command.

- "auth": Send username and password to the server then wait and display the response from the server.
- "upload": change the state of the Check expire thread.
- "download": send the request to the server and wait for new tempID from server
- "Beacon": send beacon to other clients.
- "Exit": logout.

When a client start running, it triggers Beacon handler thread and Check expire (file) thread to run. Beacon handler thread is used to handle the protocol from other clients and verifies the beacon

they send. If the beacon is valid, it displays the beacon, write a file and send a protocol to the server (then server displays it).

Check Expire is also a state machine. Its job is to keep track of which beacon is expired and deletes it from the file. Its state is changed by the client and Beacon Handler. For example, when Beacon Handler need to write to the file, it changes the state to "writing" or when the client needs to upload the file, they change the state to upload. Then check expire will do all the work. By doing this, it controls when to open and close the file, avoid conflict between read and write to file.



This figure shows how threads communicate.

Application protocol:

Server to client or client to server protocol is MessageFormat object which is passed between A server and a client. It contains 1 header field and 1 generic data field. The header informs the server how to handle the data (it's the state of the machine). The client sends an array of string to the server as the data. However, the server only responds with a string.

Client to Client protocol is a string which contain beacon's information (time stamps, client's ID).

Design and Tradeoff and Potential Improvements

Even though, the protocol has generic data's field but the server only can receive string array data. If a client wants to send a string to server, it must send an array string. This doesn't make sense in term of architecture (Why do we need to pass an array string while we only need a string?). However, string array is the most generic type we can use. We can improve this by sending in binary and convert it back to an object (but for this program, it's not necessary) to make it more generic. Furthermore, to encrypt the data for secure reasons.

We're using txt file as the data base for the program, therefore, I need to implement a check expire thread to keep track of which beacon expires and make sure the program in the right state (avoid read and write as the same time). If we implement a proper database, some databases provide checking

service and keep track which expired, therefore, no need the check expire thread, make the program faster.

The program has a simple software architecture. This could be improved by implementing a better architecture for the program to keep the code clean and avoid threading problems.