Socket Programming: Secure Connection

Assigned: Oct. 4, 2021
Due: Oct. 18, 2021 @11:30 PM
Version: 0.1.2

You work for the SSSGO (Super-Secret Shady Government Organization) and the secret agents in the field need a secure cannel to check in to HQ. For this assignment, you'll write a client (Agent) that will use sockets to communicate over TCP with a server that you will also write. Your client and server will exchange the sequence of messages shown in Fig. 1. This will allow us to authenticate each Agent in the field and establish a secure channel for them to communicate.

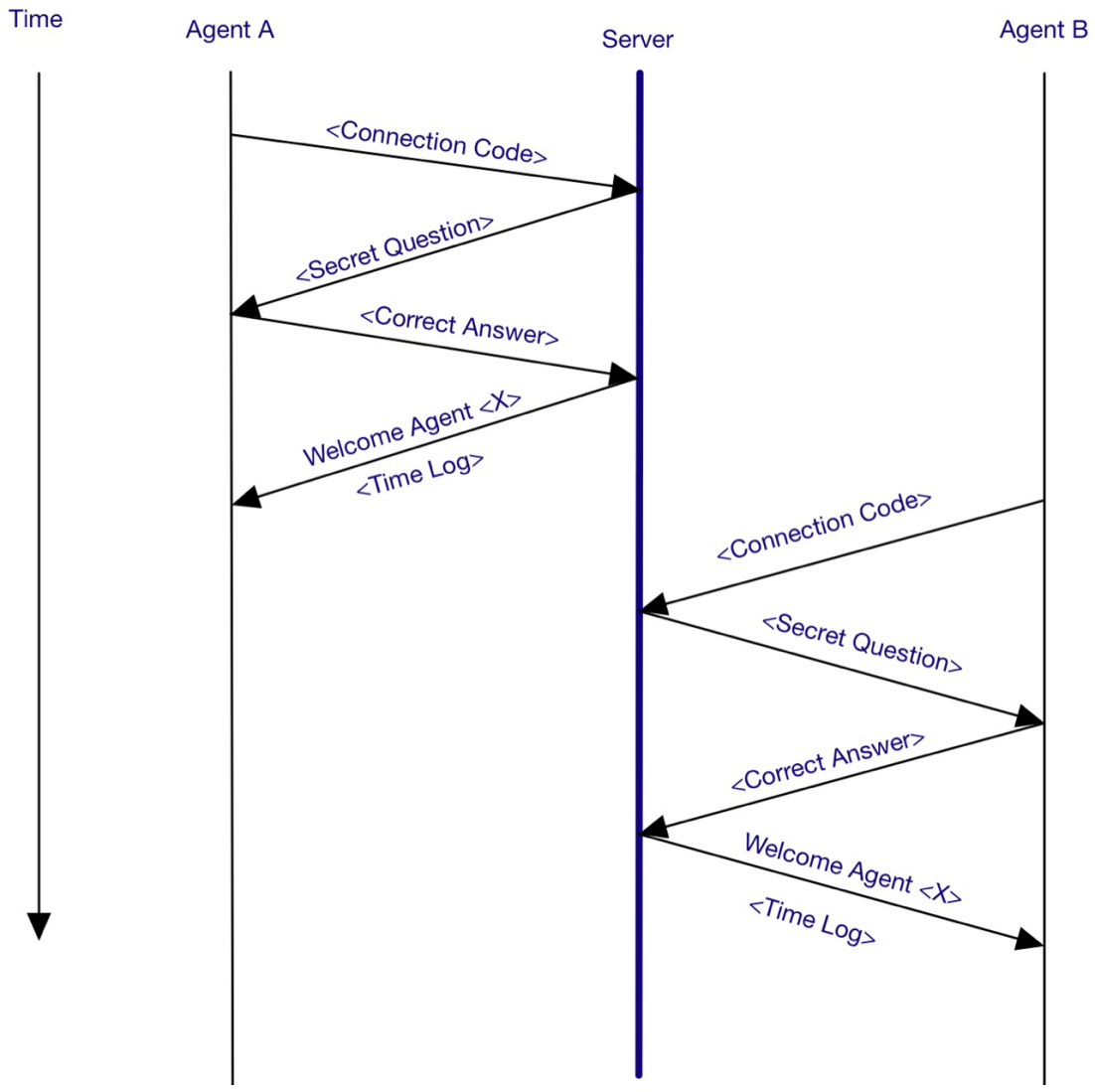

Figure 1: Message sequence exchanged between Agents and server

Description:

In this assignment, you'll write two clients (Agents) that will communicate with a server that you will also write. On start-up, the client should be prompted to enter their connection code.

Your client (Agent) will then open a TCP socket to your server and send a message (a string of characters) to your server containing their <Connection Code>. The first five characters of the connection code is a predefined set, while the last four is the <Agent Code#>.

<Connection Code> = <Predefined Charterers> + <Agent Code#>

Below are the list of the predefined characters and agent code numbers you should use in your project:

Predefined Characters: [AJK78, KTV90, NEL55, DFG28]

Agent Code Numbers:

1. Agent A – (2975)
2. Agent B – (6144)

Possible Connection Codes for Agent B:

1. AJK786144
2. KTV906144
3. NEL556144
4. DFG286144

Anyone of these codes should successfully identify Agent B.

If the connection code is unknow to the server it should close the connection to that client (Agent) immediately. If the connection code is correct, then the Server should move to the second layer of security. This is where the server sends the agent one of five secret questions (as a string). If the agent produces the correct answer to the secret question given, the server should send this message to the Agent: "Welcome Agent X Time Logged - <Date and Time>" where X is A or B

E.g., Welcome Agent B Time Logged - 2020-07-7 17:45:48.537986

If the answer is incorrect the server should send no further response and close the connection to that client immediately.

Secret Questions and Answers:

1. I saw a purple Kangaroo yesterday, did you? -> Only after the sun went down
2. What did Eve say when she ate the fruit? -> Nothing
3. What do you call a fish wearing a bowtie? -> Sofishticated
4. What did the ocean say to the beach? -> Nothing, it just waved
5. Why did God save men but not fallen angels? -> Good Question

Server Implementation

Your server should keep track of all possible connection codes for each agent. The sever should also store the secret questions and answers.

The Agent will initialize the communication with their connection code. The server will check if the connection code supplied matches any of the possible connection codes. If the connection code is incorrect the server must close the connection with that client.

After the server checks the connection code received and finds it to be legitimate it should randomly select a secret question and send it to the client. The server will then wait for an answer from the client. Upon receiving the answer from the client, the server must check it against the answer to the specific question given. If the answer is correct the server must send the welcome string with the time and date the Agent checked in: "Welcome Agent X Time Logged - <Date and Time>."

Additional details:

You have been given three python files with code you need to complete. Please not that you are not to rename any of the functions already defined (or file names) as this may affect your marks since your code will be partially marked by another script.

Note that a short design document is required. You should program each process to print an informative statement whenever it takes an action (e.g., sends or receives a message, detects termination of input, etc.), so that you can see that your processes are working correctly (or not!). This also allows the grader to also determine from this output if your processes are working correctly. You should hand in screen shots (or file content, if your process is writing to a file) of these informative messages.

Programming Notes

Here are a few tips/thoughts to help you with the assignment

- You must choose a server port number greater than 1023 (to be safe, choose a server port number larger than 5000).
- Video tutorial that might be helpful: (https://www.youtube.com/watch?v=u4kr7EFxAKk)
- You **must** write your programs in Python3. Details about the socket module can be found at http://docs.python.org/3/library/socket.html.
- I would strongly suggest that everyone begin by writing one client and one server first, i.e., just getting the two of them to interoperate corrections. Then your code for the two client case.
- You will need to know your machine's IP address, when one process connects to another. You can telnet to your own machine and seeing the dotted decimal address displayed by the telnet program. You can also use the UNIX `nslookup` command or you can ask your local system administrator for the info. Or, under Windows, see the ipconfig utility.
- Many of you will be running the clients and senders on the same UNIX machine (e.g., by starting up the receiver and running it in the background, then starting up the relay in the background,

and then starting up the sender. This is fine; since your using sockets for communication these processes can run on the same machine or different machines without modification. Recall the use of the ampersand to start a process in the background. If you need to kill a process after you have started it, you can use the UNIX `kill` command. Use the UNIX `ps` command to find the process id of your server

- Make sure you close every socket that you use in your program. If you abort your program, the socket may still hang around and the next time you try and bind a new socket to the port ID you previously used (but never closed), you may get an error. Also, please be aware that port ID's, when bound to sockets, are system-wide values and thus other students may be using the port number you are trying to use.

Rules

- **The project is designed to be solved independently**.
- **You may not share submitted code with anyone.** You may discuss the assignment requirements or your solutions away from a computer and without sharing code, but you should not discuss the detailed nature of your solution. If you develop any tests for this assignment, you may share your test code with anyone in the class. Please **do not put any code from this project** in a public repository.

What to turn in

1. You will hand in the code for the client and server implementations along with screen shots of a terminal window, verifying that your programs actually carry out the computation that is specified.
2. A separate (typed) document of a page or so (in **pdf** format), describing the overall program design, a description of "how it works," and design tradeoffs considered and made. Also describe possible improvements and extensions to your program (and sketch how they might be made).
3. A separate description of the tests you ran on your program to convince yourself that it is indeed correct. Also describe any cases for which your program is known not to work correctly.
4. The log files produced by your programs. (Server Log File.txt and Agent Chat Log.txt)
5. Everything should be submitted in a **single** Zip file with your id number as the name of the file.

Grading

- Program listing
  - Works correctly as specified in assignment sheet – 20 points
  - Contains relevant in-line documentation – 5 points
- Design document
  - Description – 5 points
  - Thoroughness of test cases – 5 points