CSCI 5531: Advanced Operating Systems
Project-1 [KSH]
A Distributed Shopping System

Due Date: Mar 13th 2022

Spring 2022

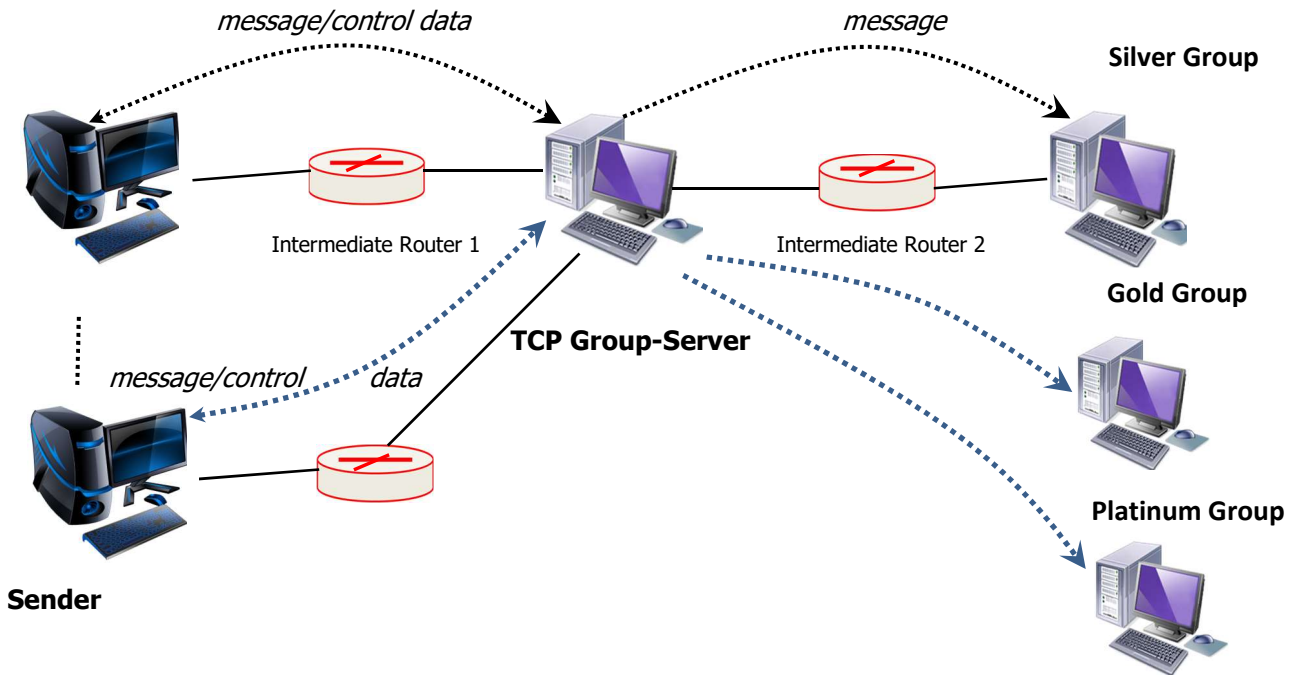
100 pts

1. Objective

The objective of this first programming project is to learn Inter-process communication through TCP iterative client, TCP concurrent middle-server, and UDP iterative client-server interaction using socket interface in Java programming language. After completing the project, you will have an basic/intermediate understanding of the steps required to develop a distributed application.

2. Project Specification

In this programming project, you are required to implement a distributed group-based shopping system involving TCP and UDP protocols in Java language. The sender process takes user-entered "TCP-Group server" IP and port from using keyboard. Next, it sends login credentials along with current member credit point to the TCP-Group server. The TCP-group server displays the message and verifies the credentials. If it is correct then the member's credit point is updated and a success message is sent back to the client. The TCP-Group server is a concurrent server. Upon successful login, the Group server communicates with dedicated Silve/Gold/Platinum server to conduct the shopping process. The entire process diagram and more specific project description are given below:



Detail specifications:

- The sender process establishes a TCP connection with the middle server. Your sender program needs to take at least *two arguments* that specify the *IP address* and *port number* of the Middle server that it will try to connect.
- After a successful connection, your sender program will first prompt a welcome message that asks the user to key in a username and password (keyboard). This credential will then be sent to the middle server. After receiving the username & pwd from your sender, your middle server will send an *acknowledgment message* back to the sender.
- You have to use (*10000 + last 4 digits* of your student-id number) to avoid requesting same port by multiple students. Group server gets middle server port number + 1.
- Your middle server process is responsible for verifying the user *authentication* and the group server details based on points.
- Upon valid credentials, the middle server will connect the client with proper group server.
- Group servers have their own item list with item price and quantity. Client will receive the list from group server and purchase items based on available credit points.
- After shopping, credit points remaining will be updated along with items quantities.
- **userList.txt [its saved in middle server machine]**

Username	Password	Group	Points
anna	a86H6T0c	Platinum	5250
barbara	G6M7p8az	Silver	985
cathie	Pd82bG57	Platinum	5244
dohas	jO79bNs1	Gold	2577
eli	uCh781fY	Gold	3579
farah	Cfw61RqV	Silver	522
shaz	Kuz07YLv	Platinum	7844

- Finally, when the shopping is over and client wants to terminate, client process will send a CLOSE message to the proper group server. On receipt of a CLOSE message, the group server process should terminate itself gracefully.

3. Programming Notes:

I would strongly suggest that everyone begin by writing the sender and middle server processes first, i.e., just getting the two of them to interoperate corrections. Then modify the relay server and write the receiver-process.

You will need to know your machine's IP address, when one process connects to another. You can telnet to your own machine and see the dotted decimal address displayed by the telnet program. You can also use the UNIX *ipconfig* or *ping (e.g., ping uhcl.sscb.edu)* commands to figure out the IP address of the machine you are working on. You can also ask your local system administrator for the info.

Many of you will be running the sender, relay, and receiver on the same UNIX/Windows 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 you are using sockets for communication, these processes can run on the same machine or different machines without modification. 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 though it is prohibited.

You should program your 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!). One should be able to 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.

Note that we will be ***using our Delta lab machines (Unix/Windows platform)*** in the lab to run and verify your codes. Since the goal of the programming project is for you to learn socket programming (not to build a hardened application), you may also assume that the relay and the receiver will only be receiving messages from a single sender or relay, respectively. You are required to work on this project alone. Any clarifications and revisions to the assignment will be posted on the course UHCL BB page.

3.1 File names:

Make sure you follow the file name guideline given below for your project:

lastNameP1Sender.java, lastNameP1MidServer.java, lastNameP1GroupServer.java, userList.txt, and other relevant files.

4. Points Distribution:

Bits and pieces	Points
Client Program	20
Middle Server	30
Receiver Program	30
Program Style (Coding style, comments etc.)	10
Documentation	10

5. Submission Instructions:

This project requires submission of *soft copy* of java codes and a *detailed documentation* describing each program and your approach of solution.

5.1. Due Sunday, March 13th 2022, 11:59 pm

The soft copy should consist of the sender, mid server, group servers and any other API file(s) along with the detailed documentation of the programs. Documentation should include your problem-solving approach, flow charts of the programs, class diagrams, discussion of data structures, algorithms, user define functions/methods, and screen shots of outputs. These must be submitted to course BB system.

6. Late Penalty:

You have to submit your project on or before the due date to avoid any late penalty. **A late penalty of *15% per day* will be imposed after the *due date (March 13th, 2022)***. After one week from the due date, you will not be allowed to submit the project under any circumstances.

Please **DO NOT MODIFY** the project code in any way after the deadline of soft copy submission. The hard copy and the soft copy must be same.

Good Luck!!