German University in Cairo
Faculty of Media Engineering and Technology
Amr El Mougy
Yasmeen Abdrabou
Hana Medhat
Mariam Samy

DMET 602 Networks and Media Lab
Spring Term 2018
Project I

1. Introduction

The objective of this project is to give you experience in developing a network application based on the client/server architecture. Using sockets API, you are required to design a chat application that is composed of a network of two interconnected chat servers. Once users are connected to one of the chat servers, they can converse with other users connected to the other chat server.

2. System Model

The system consists of two chat servers connected together. Each chat server can handle multiple clients and is connected to the other server. Half the list of clients is maintained at each server.

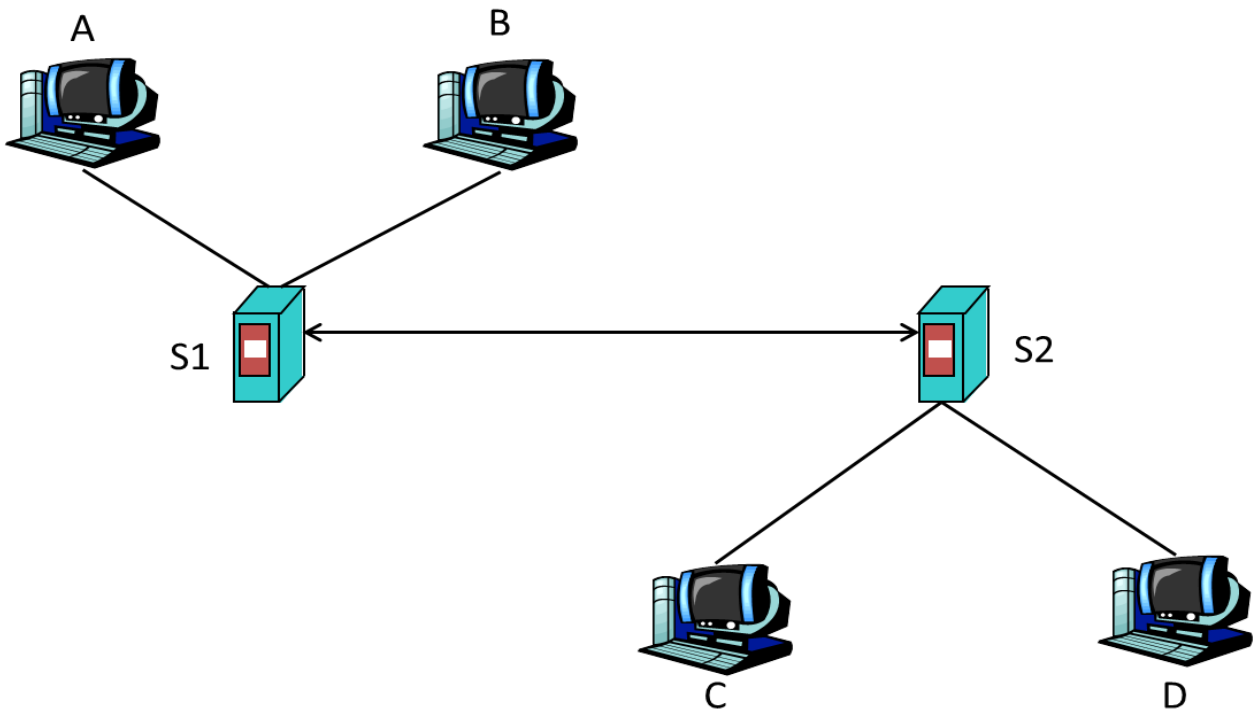
You should consider two different scenarios:

- Two users connected to the same server chatting with each other. In this case the message will be directed through their local chat server, no routing is needed.
- Two users connected to two different servers chatting with each other. In this case, the server of the initiator client must forward the message to the second chat server.

2.1 Example

Consider the network below in Figure 1 consisting of 2 chat servers. In case A wants to send a chat message to B, it will send its message to server S1. S1 will check the destination of this message (B) and compares it with the list of clients it has. Since B and A are connected to the same server (S1), thus the message will be forwarded through the server to B directly. Now let's consider the case where A wants to chat with C. In this case A will send its message to server S1. S1 will check the destination of this message (C) and compares it with the list of clients it has. Since C will not be found at the list of S1, thus the server S1 will forward the message to the server directly connected to it which is S2. S2 will then forward the message to B.

German University in Cairo
Faculty of Media Engineering and Technology
Amr El Mougy
Yasmeen Abdrabou
Hana Medhat
Mariam Samy



3. Client-Server Functionalities

To be able to understand the application here are the functionalities of the client and the server:

3.1 Client Functionalities

The client should be able to perform the following functions:

1. **Join(name)**: The client uses this message to log on to the server (i.e., initiate a chat session).The client must send this message first, before sending any other messages. The member name is any name the client chooses to be identified by.
2. **GetMemberList():** This message asks for a list of all members on the network.
3. **Chat (Source, Destination, TTL, Message):** The chat message that the user will send to another user.

Any chat message consists of three header fields and a body:

- *Source:* The *id* of the sender.
- *Destination*: The *id* of the receiver.
- *Time To Live* (*TTL*): TTL is a counter which is decremented at each chat server, when it reaches zero the message should be discarded and an error message should be sent to the sender. The purpose of TTL is to prevent the message from looping infinitely in the network. The default value for the TTL is set to 2.

4. **Quit:** This message is used to log off.

3.2 Server Functionalities

The server should be able to perform the following functions:

1. **JoinResponse:** This is the response to the join request of the client whether accepted or not. (Note: the client is only rejected when he registers with a used name).
2. **MemberListResponse**: This sends the list of all members to the client upon request.
3. **Route (Message, Destination):** This method sends the message to the targeted destination. The server should check whether the destination is directly connected to it or not. In the case where the destination is directly connected to it, then it will directly forward the message. However if the destination is not found in the list of clients connected to the server, then it will send the message to the other chat server.

4. Graphical User Interface

The project should be implemented with a graphical user interface allowing users to use your application with all the functions specified above.

5. Submission Phases

You are required to develop the project through three major milestones.

5.1 Milestone I

Submission Date: 1st week labs “3-7 /2”

In this part you are required to develop a simple client/server application that runs as follows: A server application will run on a machine waiting for a client to connect on port 6000. A client running on the same machine will connect to the server. Text messages should be exchanged between them until the client sends "BYE" or "QUIT". In this milestone a simple command line interface will be sufficient.

5.2 Milestone II

Submission Date: 2nd week labs “10-14 /2”

German University in Cairo
Faculty of Media Engineering and Technology
Amr El Mougy
Yasmeen Abdrabou
Hana Medhat
Mariam Samy

In this part you are required to modify your program so that the server will be able to handle simultaneous service requests in parallel. To be able to do that , the server should be multi-threaded, where in the main thread the server listens to incoming connections and when a connection is requested, it sets up a TCP connection and services the request in a separate thread. In this milestone different clients connected to one server can chat together.

5.3 Milestone III

Submission Date: 3rd week labs "17-21 /2"

In this part you will complete the missing parts of the project. A second server will be implemented in the same way as the first one. A defined set of clients will be connected to each of the two servers. One client connected to one server should be able to chat with a client connected to the second server.

5.4 Milestone IV

Submission Date: 4th week labs "24-28 /2" EVALUATION WEEK

A GUI should be implemented to test the final project.

6. Grading

- This is an individual project where cheating cases will be treated severely.
- You have to submit each milestone in it's lab and take a task on it.
- One lab will be dedicated to the final project evaluate

7. Recommended Reading

Socket programming with TCP: Section 2.7 Computer Networking, James Kurose and Keith Ross.