CS594
Inter-networking protocol
Portland States University

THE PROJECT: Internet Relay Chat Application Protocol

Dec 6, 2019
Expires: March 6,2020

Status of this Memo

This paper has been modified for the class purposes. Note that it is not allowed to be published over the Internet without permission. About distribution, just the group member can distribute the document under the leader supervision. The validation of This document will be cancel after three months.

Table Of Contents

1. Introduction:

This paper is a description of our work in programming Internet Relay Chat protocol IRC, that can handle the basic communication between a bunch of clients using practical server.

Basically, it is new method that can successfully handle the connected clients to the server and produce some basic options such as creating room, sending messages, listing users in addition to listing all users in specific room. On the other hand, the server job is to handle all the connected clients and proceed their requests successfully.

2. Conventions used in this document:

In this document, any word with a capital latter or between parentheses or apostrophe is from the code that we listed below this specification. The words room and channel will be used alternately.

3. Main Important facts:

The clients can connect to the server over the TCP/IP protocol, which is the type of the socket we have been used here. The port is ‘1234’ while the host have this IP ‘127.0.0.1’.

4. Semantic label:

- The message length should be 1024.

- Contain any valid character on the format of ASCII.

- Must be at least 1 character.

- The length header for the message should not exceed the length header condition.

- When the clients want to send a command, this command should start with "/" before the command itself, for example: /join and the channel name. to enter or creating channel.

- The channel will be specifying by # before it such as #flowers: that means there is a channels its name is flower.
- There should be an error messaging will appear either in the client's terminal or the server terminal if any of the roles are broken.

5. The Server:

5.1 Handling Connection:

First, we used SELECT library to handle the multiple incoming connection gracefully. Then the server will bind and listen for the incoming client's sockets. In addition, when the clients do the connection, the sever need to accept all that connections and store them in the list sockets.

5.2 Handling messages:

After receiving messages from the client, the server will respond to that message by the appropriate respond depending on the functions that will handle them. In this protocol, the server HAS three main functions to handle the messages are:

‘def messages(self, c_sockets)’
‘def cmd_message(self, cmd, chan_name, s)’
‘def send_message_to(self, chnl, msg, s)’

The first one will handle the message and the header. The second will deal with the commands. While the last one will handle the messaging distribution between the users in some channel.

If there is something went wrong, the exception statements on these three will either close the connection or give an error message to the user.

5.2.1 Joining and creating the channels:

First, If it is join to a channel, the server will check if the channel is already exist or not. If not, the server will create one and save it to the dictionary. Also, it checks if the user not in that channel to add it there.
Then it will response to the Client that he is successfully did the join. If he is already in the channel, the response message will tell him that he is already there.

5.2.2 listing:

The main function for this command is to list the users that connected to the server. However, it will create a list to fetch the users from the results of FOR statements that will abstract all the value of a ‘clients_info’ dictionary that we have used to save all the sockets of the clients as keys and the information as values. After that, it will save the users in the list and then send these users to the client as response message. Basically, it will list all the users that connected to the server. In the help of ‘cmd_message’ function.

5.2.3 listing the Users in Specific Channels:

In this case, the server will check the message if the channel is mentioned, if not then the server will send a response says that the channels is not exist and ask him to write that channel. Otherwise, it will fetch all the users in that channels using FOR loop then it will send response message to the client's terminal with these users, in the help of a 'cmd_message' function.

5.2.4 listing channels:

This command will list all the existing channels that have been created from some users in the server, in the help of a 'cmd_message' function.

5.2.5 send message to the channel:

Unlike the other commands, the send-to-message command will be done in the help of 'send_message_to' function. Firstly, When the user wants to send message to some room, the server will make sure first that the user in the room and then will distribute that message to all users' terminals in that channels.

5.2.6 leaving:

This command for the users who want to leave the channel so the server will remove the users in that room and send to him message says that you left channel. If the user already not in that channel, the server will send a message says you not in that channel using the 'cmd_message' function.

5.2.7 quitting the server:

When the server receives a quit command, it should remove the user from the channels if it is still existing. Also, to remove the sockets for closing the connection.

5.3 handling distribution:

The server will handle the messages between the clients. consequently, when the use writes the message to send it in some channel the server will check if this user inside that channel or not. Then the server will check if the clients is not the sender from the users in the channel, it will send and forwards the messages for the all the users in that channel.
Basically, the user should type the channel name before he tries to send the message then the server will distribute that message to all the users with the username of the sender.

5.4 Dealing with client's sockets:

After establish the connection of the new clients, then the server will check if the connection is bad or there is something wrong with the connection. In that case, the server will close and remove the clients from the server. Also, the server check if the client is new, then it will add his information to the clients list. Here, we used the SELECT statement lists to distinguish between the sockets condition.

When the client sends a command such as join, list, usernchan, listchannel, leave or quit, the server will check if the command message is in perfect format then it will send the command to the command function 'cmd_message' to produce it. However, if the command is 'quit', it will remove the user then remove the sockets and finally close the connection.

6. The Clients:

Eventually, the clients will set up the socket connection with assigning the port to ‘1234’ and the IP to ‘127.0.0.1’. The main job for the client is to make sure that the connection in good situation and get connected perfectly to the server beside forwarding commands. basically, when the user is connected successfully, the client will send a message to the user terminal or webpage to ask for the username. Then it will print a message to list for the new user all the provided commands that our server has and how he can use them. Before the sever starts dealing with the users’ commands, it used the SELECT statement in the client’s side for differentiate if the client is receiving or sending messages.

6.1 Listing message:

In this case, the clients will wait for the user to enter a command using ‘input()’ function. After that the to the user terminal. In case of error occurs, it will raise an error exception.

6.2 Joining message:

If the command was ‘join’, it will sent by the clients to the server to join the channel, if no one is exist it will create one. Also, there is an error raise message to handle any error will occur.

6.3 Leaving message:

If the message is to leave the channel, the clients will send to the server so that the user will be removed from the channels list. Also, there is an error message to handle any error will occur.

6.4 Quitting message:

if the user wants to quit the connection, the clients will tell the server about quitting command and then the socket will be close. There is a message error handling function to handle the errors.

6.5 Listing channels:

if the message was a ‘list channel’ command, the clients will direct the message to the server to activate this command.

6.6 listing User in The Channel:

when the user enters ‘usernchan’ command the clients will forward it to the server so the user can get all the user. also

6.7 Sending messages to a Channel:

by entering ‘sendmsgto’, the server will use this command to send the user message to the users in that channel. and there will be some error handling message will appear if there was any problem.

7. Security and Privacy:

This server didn't guarantee any secure or privacy for the messages between parties and it didn't support the end-to-end encryption principle.

8. Conclusion:

Mainly, this is a simple relay chat protocol that handle multiple communication using one server.

- Working group: Ahmed Almutairi, Hanin Alshalan.