PORTLAND STATE UNIVERSITY

CS 594 - Project Description

Internetworking Protocols

Title - Internet Relay Chat

Done by: Sri Lakshmi Polavarapu

Sahithi Mothe

Apurupa Bodha

STATUS OF THE DEMO

By submitting this Internet Relay Chat project, each author of this project represents that any patent who is applicable where or other IPR claims of which he or she will be getting to know or will be communicated and any of which he or she is aware of the project will be disclosed.
The Internet Engineering Task Force (IETF), its areas, and its working groups publish working documents known as Internet-Drafts. Keep in mind that working documents may be dispersed as Internet drafts by other groups as well.

Internet drafts can be modified, replaced, or provided redundant by other documents at any time. They are only intended to be used for a maximum of six months. Internet drafts should not be referenced or used as a source of information other than as "work in progress."

The list of Internet-Draft Shadow Directories can be accessed at

http://www.ietf.org/shadow.html.

GITHUB LINK: https://github.com/SriLakshmiPolavarapu/Internet-Relay-Chat

ABSTRACT

This project mainly focuses on development of Internet Relay Chat (IRC) where the final output provides reliable communication between users. IRC (Internet Relay Chat) is a standing protocol for real-time communication for an Internet Working protocol course in Portland State University. Here we used Python as our programming language for this project.. We want to start small because this is our first term, so we have taken chat functionality.

TABLE OF CONTENTS

1 Introduction 3

2 Basic Information 4

3 Message Information 4

4 Client-Side Message 5

5 Server-Side Message 6

6 Error Handling 7

7 Future Work 7

8 Conclusion 8

9 Security 8

10 References 8

11 Acknowledgement 9

1. INTRODUCTION

Internet Interoperability Protocol (IRC) allows users to connect to public servers and be able to chat or chat. The server acts as a central hub connecting clients. Here, users joining the server can connect to multiple rooms or create rooms and join multiple clients, each client can post their messages for clients in the room information. Users can even personally chat with other clients. Users can move from one room to another by listing all available rooms and joining the rooms they want. Users can also leave the room, whenever they want.

2. BASIC INFORMATION OF INTERNET RELAY CHAT

This project is based on the Transmission Control Protocol (TCP) using client-server architecture. Here the server will be in a "listening" state, constantly listening to the client, to make it listen continuously we have used threads. In this Internet Relay Chat, when a user joins the server, the message "Please enter your name" appears and the user has to enter his nickname to continue chatting with other users. Clients establish a connection with the server by connection to the ports. By using this open channel client can easily communicate with server, send messages, and receive messages. The message system here is asynchronous, because client and server work simultaneously. The client can terminate the connection at any time by displaying an error message.

3. MESSAGE INFORMATION

The maximum size parameter for each message sent by the client and server is 1024 bytes. The message is decoded and encoded using ASCII code characters.

Code example: message = client.recv(1024).decode('ascii')

4. CLIENT SIDE MESSAGES

When a client joins a room a message will be posted in the room stating that the user's name has joined the room.

Code example: client.send(nickname.encode('ascii')). When the client finally gets an error, the error below occurs as shown.

Code example: print("An error occurred"). When the user leaves the room or the server an error will be displayed and no message will be displayed because the client just needs to close you. You can easily log out by opening the window.

Here are the list of commands displayed in the client terminal:

4.1 Greetings, client

The user will enter his username.

4.2 List every room

Usage: the user will input the command "$list."

Reaction: gives list of rooms

4.3 Create and join a room - To use, enter the command "$join"

In response, a room with the specified name will be created.

4.4 Leave a Room: To use this command, type "$quit"

Reaction: The client will leave the room he is currently in.

4.5 Send a personal message:

To use this command, type "$personal nickname message"

4.6 Get help

Usage : User should the command $help

Response - client will display all the list of the commands

5. SERVER SIDE MESSAGES

When the user leaves or disconnects from the communication chat room, the server will receive a message as shown below

Code example: post(f'{nickname} Left The chat '.encode('ascii'))

Also when a customer joins, it sends a message and sends the content of the chat between customers in different rooms for research or future use.

Code example: print(f"User's nickname: {nickname}!!!")

post(f'{nickname} Joined the room'.encode('ascii'))

client.send('Connect to the server.'. encoding ('ascii'))

When the server is on, the server will be in listening mode and will send

messages as shown below:

Code Example: print("Server is listening.")

6. ERROR HANDLING

We used only try except blocks for errors and client-side processing, and also we have used try blocks on the client side to handle data entering and leaving the server.

7. FUTURE WORK

In the future we will be trying to add a files sharing system and also try to add emojis when a user is trying to message a different user. We will also try to implement a secure messaging platform which is implemented on the cloud.

8. CONCLUSION

This specification provides a communications protocol for various clients to communicate from a central server to a server using TCPconnection. If there is no change to this specification, users can build their own systems based on the shipping documents described here. This specification is the first version of Internet Relay Chat (IRC). On this basis, the implementation plan was created, which is the framework through which many people can communicate with each other through the server at same time. In easy words, Internet Relay Chat provides a message communication framework for various clients to communicate with each other via a central server communication.

9. SECURITY

These messages cannot be analyzed, interacted with, or transmitted directly from the Internet. Because all messages are encrypted during transmission. Only at the end i.e. client or server side. The message is decrypted for processing. This is done using the symmetric encryption algorithm and only those who have the key can decrypt the message

10. REFERENCES

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997

11. ACKNOWLEDGEMENT

This document has been prepared by Sri Lakshmi Polavarapu, Sahithi Mothe, Apurupa Bodha