CS-494P/594-001 FALL 2023

Internetworking Protocols: Final Project

INTERNET RELAY CHAT

Author #1: Snehil Shrivastava / snehils@pdx.edu

Author #2: Manisha Katta / manishak@pdx.edu

Author #3: Nida Mariam Sheikh Aslam / nidama@pdx.edu

Status of this memo:

By submitting this Internet-Draft, each author represents that any applicable patent or IPR claims of which they are aware have been or will be disclosed, and any of which they become aware will be disclosed, per RFC 3668.

This Internet-Draft is submitted in full conformance with BCP 78 and BCP 79 provisions. This document may not be modified, and derivative works of it may not be created except to publish it as an RFC and to translate it into languages other than English.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and working groups. Note that other groups may also distribute them as Internet-Drafts. Internet-Drafts are valid for six months and may be updated, replaced, or obsoleted by other documents. It is inappropriate to use Internet-Drafts as reference material or to cite them as works in progress.

The list of current Internet-Drafts can be accessed at

http://www.ietf.org/ietf/1id-abstracts.txt.

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

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

Abstract:

In this project, we will implement the internet relay chat. A single IRC server is created, and people connect to it using an IRC client. This protocol allows users to set a username on the server and communicate in private or group chats over various IRC channels. This is our first chatbot project, and we want to start small, so the focus is the chat functionality. The ambitious future of this project is chat and voice functionality with secure messaging.

Table of Contents

1. Introduction **4**
1.1 Server 5
1.2 Client 5
1.3 Room 6
2. Conventions used in this document **6**
3. Basic Information **7**
2. Specifications **7**
2.1 Mode of Communications 7
2.2 Character Codes 8
2.3 Messages 8
2.4 Replies 8
3. Message Infrastructure **9**
3.1. Create room 9
3.2. List available rooms 9
3.3. Join a room 10
3.4. Leave a room 10
3.4. Switch room 11
3.5 Send a Personal Message 11
3.6 Quit 12
3.7 Help 12
4. IRC Concepts **13**
4.1. One-to-one communication 13
4.2. One-to-many communication 13
4.3. One-to-all communication 13
5. Error Handling **14**
6. "Extra" Features Supported **15**
6.1 Private or Ephemeral Messaging 15
6.2 Cloud connected server 15
7. Conclusion and Future Work **16**
8. Security Considerations **16**
9. IANA Considerations **16**
10. Normative References **17**
11. Acknowledgments **17**

1. Introduction

The IRC Application project is dedicated to enhancing the chat experience by implementing an efficient server-client system. Within this framework, a central server assumes the role of managing connections, message routing, and chat room administration, ensuring a well-organized and uninterrupted interaction.

This project aims to provide users with a user-friendly platform where they can effortlessly join various chat rooms, create their own, discover available rooms, and engage in conversations. This platform operates like a well-organized social gathering, where hosts can invite guests, check room availability, and identify participants.

An important feature of this project is its capacity to accommodate multiple users simultaneously, facilitating group discussions and collaborative efforts. Users can send messages to specific chat rooms, akin to group messaging, and even engage in multiple conversations concurrently, tailoring their communication to diverse needs.

This system is also designed to handle user exits gracefully, ensuring a smooth departure from chats or the overall system. In the event of unexpected issues, such as chat disruptions, the system will maintain its stability, preventing widespread interruptions.

In addition to these features, the IRC Application project will also implement file transfer capabilities, further enhancing the utility and versatility of the IRC platform.

1.1 Server

In the context of IRC (Internet Relay Chat), a server plays a central role in facilitating real-time text-based communication among users. These servers are distributed around the world and are responsible for hosting chat rooms or channels. When users connect to an IRC server, they gain access to a vast network of chat rooms, each dedicated to different topics or interests. These servers act as the backbone of the IRC network, ensuring that messages are relayed swiftly and efficiently between clients. They also manage user connections, authentication, and the creation and administration of chat rooms. Essentially, IRC servers serve as the foundational infrastructure that enables the global community of users to engage in conversations, share information, and build communities in the virtual realm.

1.2 Client

The client serves as the user's interface to the network. In the IRC network all users connect to a single central server. Each client can be identified by a name, which serves as a unique identifier for the client within the network, allowing users to distinguish and interact with one another. Clients are responsible for sending and receiving messages, participating in chat rooms, and engaging in one-on-one or group conversations. They play a vital role in facilitating the exchange of information and communication within the IRC network, making it a versatile platform for online discussions and community building.

1.3 Room

A chat room, often referred to as a "room," is a virtual space where users gather to engage in real-time text-based conversations on a particular topic or subject of interest. These chat rooms serve as hubs of communication, allowing individuals from around the world to come together and discuss common interests, share information, or simply enjoy conversations. Within a chat room, participants can exchange messages that are visible to all members of the channel, fostering group discussions and a sense of community. IRC chat rooms can vary widely in size, topic, and user base, making them versatile platforms for both casual and structured conversations on the internet.

2. Conventions used in this document

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT","SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in thisdocument are to be interpreted as described in RFC 2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lowercase uses of these words are not to be interpreted as carrying significance described in RFC 2119.

In this document, the characters ">>" preceding an indented line(s) indicate a statement using the keywords listed above. This convention aids reviewers in quickly identifying or finding the portions of this RFC covered by these keywords.

3. Basic Information

All the communication in this protocol happens through TCP/IP, and the server is always ready to receive connections on port 9000. Clients connect to this port and stay connected to the server persistently. Through this open channel, clients can send messages and requests to the server, and the server can reply in the same manner. It's worth noting that this messaging protocol is designed to be asynchronous, meaning clients can send messages to the server whenever, and the server can send messages back at any time.

Either the server or the client can decide to end the connection at any point for any reason. In such cases, an error message may be sent to let the other party know why the connection was terminated.

The server may choose to limit the number of users and rooms based on the host system's capabilities. Error codes are in place to inform connecting clients if there's a high demand on the server due to many users or groups accessing it. This helps manage the server load effectively.

2. Specifications

2.1 Mode of Communications

In IRC, users communicate through public channels for group chats, private messages for one-on-one conversations, commands for various actions, notices for important messages, and CTCP messages for special interactions. These modes facilitate real-time interactions with varying degrees of privacy and functionality on the IRC network.

2.2 Character Codes

A "character code" usually refers to the character encoding used to represent text and messages within the IRC network. Character encoding determines how characters, symbols, and text are encoded and transmitted in a standardized format so that different devices and software can interpret and display them correctly.IRC typically uses the UTF-8 character encoding for text communication. UTF-8 is a widely supported encoding that can represent a broad range of characters, including those from various languages and special symbols.

2.3 Messages

IRC messages are the basic units of communication in Internet Relay Chat. They include public room messages for example, in a room called **"cyan"** a message might look like this: **[cyan] Nida: Hello!** private messages for example, a private message might look like this: **[personal message] Nida: Hi,Snehil!**, commands (starting with hash) For example, to join a room, you might use the command **#join <roomname>**, server messages, CTCP messages for client-to-client interactions, and notices for important or system messages. These messages are typically plain text and facilitate real-time text-based communication among users in chat rooms and private conversations on the IRC network.

2.4 Replies

Replies are standardized numeric codes or messages sent by the server in response to client commands or actions. These codes indicate specific outcomes, events, errors, or information, helping clients understand the results of their actions and facilitating communication between clients and the server.For example, when a user called **"Snehil"** successfully joins a room

called **"cyan"**, the server may send a reply such as **[cyan] Snehil joined the room.**

3. Message Infrastructure

3.1. Create room

Initially, there are no rooms available. To create a room for the IRc project, connect to the IRC server using an IRC client. Once it is connected, create a room by using the **#join <roomname>** command. For Example, to create a room called **"EB-92"**, the following command has to be executed, **#join EB-92.** This created room will now be visible to all the other users to join and start communicating.

```
List of commands:
1.#list:                        To list all the rooms
2.#quit:                        To quit
3.#help:                        To list all the commands
4.#leave:                       To leave the room
5.#join <roomname>:             To join or create the room
6.#switch <roomname>:           To switch room
7.#personal <name> <message>:   To send personal message
#list
No rooms are available to join
#join cyan
cyan created
```

3.2. List available rooms

Users can view a list of all the currently available rooms created by other users and the users present in the room by using the **#list** command.

```
#list
List of available rooms:

Room: cyan
Members:
- snehil
- nida

Room: EB-92
Members:
- nida
```

3.3. Join a room

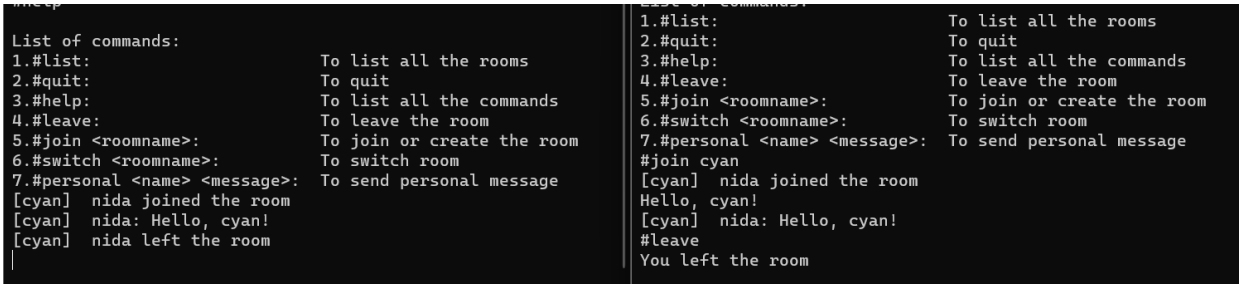
To join a specific room from the list, users can utilize the **#join <roomname>** command.For example, if the user "Manisha" wants to join the room "EB-92" then they have enter the command #join EB-92. In response, other users in the room will get the message **[cyan] Manisha joined the room.**

```
Enter your name: nida
Connected to the server!

List of commands:
1.#list:                        To list all the rooms
2.#quit:                        To quit
3.#help:                        To list all the commands
4.#leave:                       To leave the room
5.#join <roomname>:             To join or create the room
6.#switch <roomname>:           To switch room
7.#personal <name> <message>:   To send personal message
#join cyan
[cyan]  nida joined the room
```

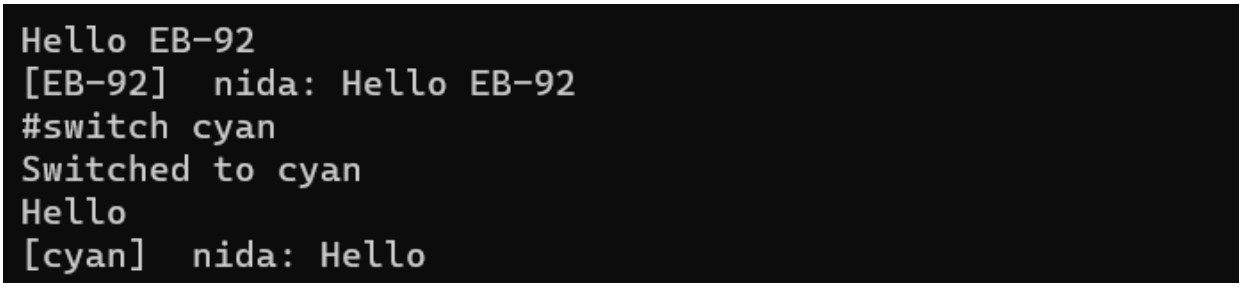
3.4. Leave a room

Users have the flexibility to exit a room at any time using the following **#leave**. For example, to leave the room #EB-92, the following command has to be executed, **#leave**. The room remains accessible even after a user leaves and if other users are present in the room. However, if the room becomes empty, it ceases to exist.



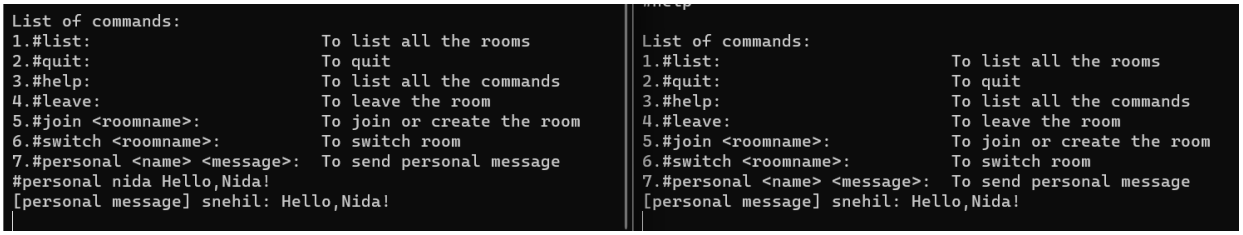
3.4. Switch room

Users have the flexibility to switch to another room at any time using the following **#switch <roomname>**. For example, the user "Nida" is a part of rooms "EB-92" and "EB-80". If the user is currently in "EB-92" it can use the command #switch EB-80 and the user will be switched to room "EB-80".



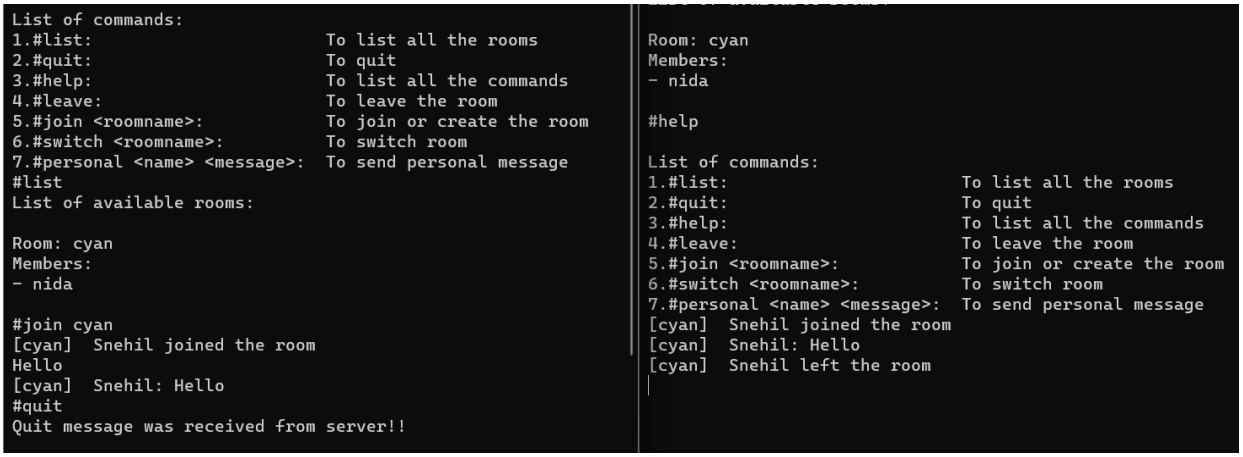
3.5 Send a Personal Message

The users can send personal messages to another user using the command **#personal <name> <message>**. For example, The user "Manisha" wants to send a personal message to the user "Snehil", they can do so by typing #personal Snehil Hello,Snehil!. Snehil will receive the message in the following format: **[personal message] Manisha: Hello,Snehil!**



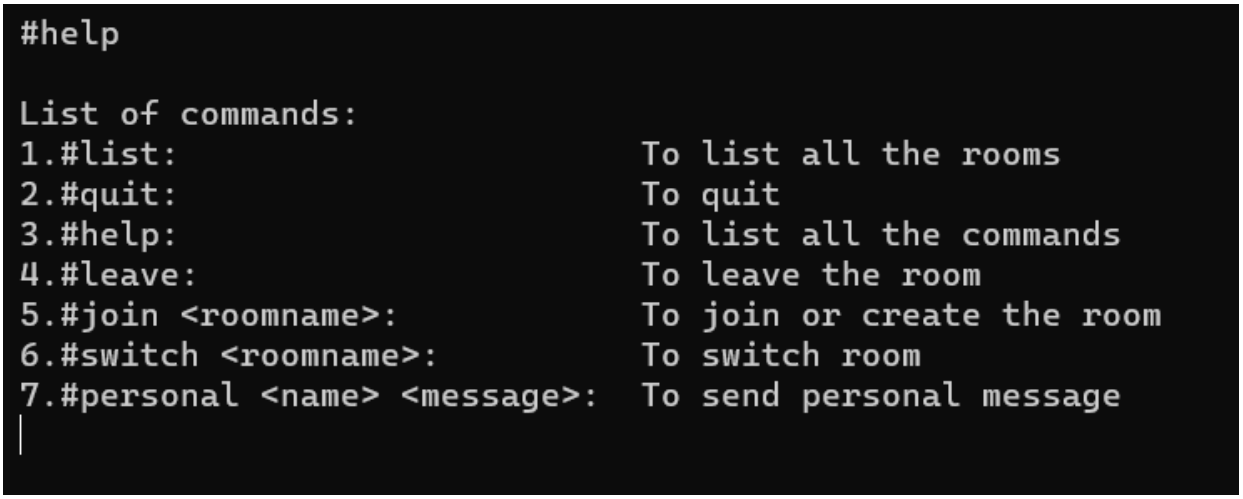
3.6 Quit

The user can quit the server by using the command **#quit**.



3.7 Help

If the user wants to view the catalog of options available to them they can do so by using **#help** command.



4. IRC Concepts

4.1. One-to-one communication

One-to-one communication is useful for discussing private matters, sharing sensitive information, or simply having a more focused conversation without the noise of a public channel. It's important to note that the exact commands and features for one-to-one communication may vary depending on the IRC client you're using, so you should refer to your client's documentation for specific details. Additionally, the ability to send private messages may be subject to network-specific settings and permissions. It allows users to have private conversations by using the **#personal <name> <message>** command <name> is the target user's nickname and <message> is the message you want to send.

4.2. One-to-many communication

One-to-many communication occurs when a room is created and multiple users join that room to engage in group discussions, where messages are visible to all participants. It allows for collaboration, information sharing, and group interactions.

4.3. One-to-all communication

A client can send a broadcast message that will be delivered to each and every client and server. Broadcast messages in IRC are messages that are sent to all clients and servers on the IRC network. This can be useful for sending important announcements or alerts to a large number of users at once.

5. Error Handling

- Only authorized users (those who are part of a particular room) can send messages within that room. If a user who is not a member of a specific room attempts to send messages to it, the server should reject these messages. This enforcement ensures that only authorized users can participate in discussions within a particular chat room.
- When a user attempts to leave a chat room (referred to as a "channel") they are not currently a member of, the IRC server should respond with an appropriate error message. This error message, such as "You are not part of any room," serves to inform the user that they cannot leave a room they are not a part of. This is an essential mechanism to prevent unnecessary confusion and ensure that users can only exit chat rooms in which they are valid members.
- If a client attempts to communicate with a user who does not exist (e.g., due to a mistyped or disconnected user), the server should handle this situation by delivering an appropriate error message. This error message will typically inform the sender that the target user does not exist, maintaining effective and relevant communication within the network.
- In the event of a client crash or disconnection in IRC, a well-implemented server should detect this situation. This may involve the use of periodic "heartbeat" messages sent by the client to confirm its active status. If the server detects a client crash or disconnect, it should take appropriate actions. For example, the server can remove the

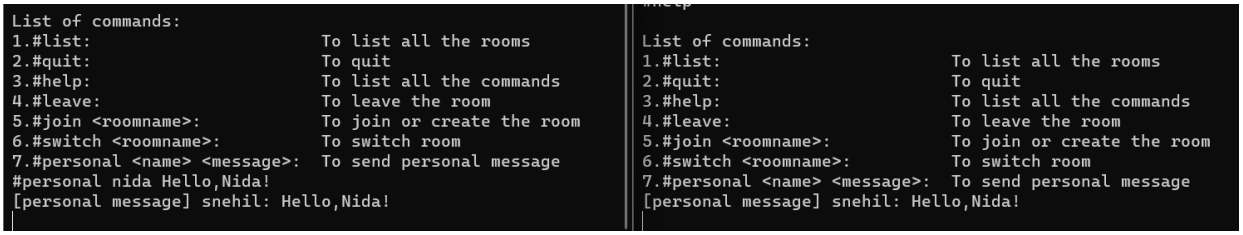
disconnected client from any rooms they were previously part of and notify other users within those rooms about the client's departure.

- Similarly, if the IRC server experiences a crash or becomes unresponsive, clients connected to that server should be gracefully logged out to prevent further communication issues. Users should be notified with a message like "Server not responding." This enables users to understand the situation and attempt to reconnect to an alternative server if available, ensuring the continuity of their chat sessions.

6. "Extra" Features Supported

6.1 Private or Ephemeral Messaging

The users can send personal messages to another user using the command **#personal <name> <message>**. For example, The user "Manisha" wants to send a personal message to the user "Snehil", they can do so by typing #personal Snehil Hello,Snehil!. Snehil will receive the message in the following format: **[personal message] Manisha: Hello,Snehil!**



6.2 Cloud connected server

In the GCP deployment of an Internet Relay Chat (IRC) system, a virtual machine (VM) instance hosts the IRC server, leveraging Google Cloud's infrastructure. The VM allows for scalable and flexible resource allocation based on demand. Security measures,

monitoring tools, and GCP-specific features contribute to a robust and reliable IRC system.

7. Conclusion and Future Work

This document has provided an overview of IRC, from its single server architecture to client interactions and error handling. While IRC has served as a foundational communication tool for decades, it is essential to recognize that the digital landscape is ever-evolving. In addition to these features, the IRC Application project will also implement file transfer capabilities, further enhancing the utility and versatility of the IRC platform.

8. Security Considerations

Messages sent through this system lack safeguards against inspection, tampering, or potential forgery. The server has visibility into all messages transmitted via this service. Even messages labeled as 'Private' could be susceptible to interception by a third party with the ability to capture network traffic. Users seeking secure communication through this system are advised to employ or develop their own user-to-user encryption protocol.

9. IANA Considerations

None.

10. Normative 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. Acknowledgments

This document was prepared using Google Docs.