CS 594 Sukanya Kothapally
Internet Draft Portland State University
Interworking Protocols – IRC Project Specification December 4, 2019

Internet Relay Chat

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. 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 its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work 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

This Internet-Draft will expire on December 31, 2020.

Copyright Notice,

Abstract

Most of the communication networks are based on the client-server architecture. Internet Relay Chat (IRC) is one of the examples that is using this architecture. This document describes about the working of this application and IRC protocol.

Table of Contents

1. Introduction

Internet Relay Chat is one of the client-server networking models and is an application layer protocol where clients can communicate in group with each other. When IRC client logs into this application, the user can join any public chat room that is currently on that server. The means of communication to a group of users connected to one chat room is through channels. Thus, clients communicate with the chat server

through channels and server echoes the conversations on each channel and transfer the message to every other client connected to that chat room. The client can even send private messages as well as group messages to other clients.

2. Conventions used in this document

Values within the code piece presented in this document, and in the text of this document that are dynamically allocated or hard-coded will be enclosed within < >.

3. Basic Information:

The main components of IRC protocol are client, server, channel.

3.1 Client:

The client of IRC is the computer program that can be installed by the user in their systems. Each user is distinguished by a unique nickname whose maximum length is eight characters. The communication between client and server is asynchronous in nature as client can send message at any time and server can reply at some other time. IRC message has three main components, the prefix which is optional, the command and the command parameters and they are separated by one space character.

3.2 Server:

Each server is uniquely defined by a name. Each server knows every other server. The server, when it receives message identifies its source using the prefix.

3.3 Channel:

The channel is basically a group of users that gets messages intended to that channel. The name and its current members are describing factors of a channel. The channels on the network can be known using command LIST. There are various modes of a channel where channels with mode +s and +p (secret and private channels) are displayed. The channel modes define the properties of each channel and modes could be manipulated by the channel members.

4. Commands:

4.1 Client:

Connection Registration: To register a connection as well as to disconnect properly with an IRC server. In acknowledge the client receives WELCOME <CLIENTNAME/> indicating the registration of the connection and now the client is known to the server.

4.1.1:

CREATEROOM - For creating room, CREATE command is used

Command: createroom <ROOMNAME/>

The server responds with message “Room <ROOMNAME/>created” to the client who created the room. At server, “Room <ROOMNAME/> created by <CLIENTNAME/>” will be seen.

- When client tries to create room that already exist, then error will be thrown as “Room already exits”

4.1.2:

JOINROOM – A Client can join in any of the rooms created.

Command: joinroom <ROOMNAME/>

The server responds with message “joined to room <ROOMNAME/>” to the client who joined and with message “<CLIENTNAME/> joined the room <ROOMNAME/>” to clients already in the room. At server, “<CLIENTNAME/> joined room <ROOMNAME/>” will be seen.

- When client tries to join any non-existing rooms, the error will be thrown to client as “Room <ROOMNAME/> doesn’t exist”.

4.1.3:

MESSAGEROOM – The Client can send message to group or room which will be broadcasted to all clients in that room. A client can even send different messages to different rooms using <ROOMNAME/> in command in which he has joined in.

Command: messageroom <ROOMNAME/> <MESSAGE/>

The server responds with message “Message sent” to client and at server the message “<CLIENTNAME/> broadcasted the message” will be seen.

- If client tries to broadcast to message to room in which he is not member of, then error will be thrown to client as “You are not member of this room”.

4.1.4:

LISTMEMBERS – A client can list members of any room.

Command: listmembers <ROOMNAME/>

The server responds with <CLIENTNAMES/> who are members of that particular room.

- When client tries to list members of any non-existing room, the error will be thrown to client as “Room <ROOMNAME/> doesn’t exist”.

4.1.5:

LISTROOMS: A client can request for list of existing rooms.

Command: listrooms

The server responds with list of <ROOMNAMES/>.

- In case if proper command is not provided, the error “Invalid Command! Please enter proper command” will be thrown.

4.1.6:

PRIVATE: A client can send private messages to other client.

Command: private <CLIENTNAME/> <MESSAGE/>

The client who sent private message to other client receives message as “Message Delivered”.

- If the other client whom the client is sending private message does not exist, the error “Client <CLIENTNAME/> does not exist.

4.1.7:

LEAVEROOM: The client can leave any room that they are the members of.

Command: leaveroom <ROOMNAME/>

The server responds with message “you left the room <ROOMNAME/>” to the client who left the room and with message “<CLIENTNAME/> left the room <ROOMNAME/>” to the other client members of that room.

- When client tries to leave the room in which is not member of, then error “You are not member of room <ROOMNAME/>” will be thrown.

4.1.8:

QUIT: A client can disconnect from the server with quit command.

Command: quit

Response: The client will be removed from all the rooms he is member of and the client receives message “See you later! <CLIENTNAME/>” and at server “<CLENTNAME/> Left” message will be seen.

4.2 Server:

QUIT: The server can disconnect from all the clients.

Command: quit

Response: All the clients connected to the server will receive message “Server OFF! Try later”.

4.3 Crashes:

The server crash is handled gracefully by through exceptions and all clients connected to server will receive message “Server crashed. Existing to handle server crash gracefully”. When the client crashes, the Server will receive message “Client Disconnected!”

5. Error Handling:

The errors might occur when either client or server connections are lost or when socket link connecting them might have been terminated. The server and client must be able to detect the error by continuous

track of heartbeat messages. When server detects that connection to client is lost, it MUST remove all the client from every chat room it has joined in. Likewise, if client detects it has lost connection with the server it MUST know by itself it has been disconnected and MAY try to reconnect to the server.

- In case if proper command is not provided, the error "Invalid Command. Please give correct command" will be thrown in all above cases.

6. Security Considerations

There are few security issues in IRC as messages sent through connections are not usually encrypted and higher risks of being target to hackers. A careful security policy is important to ensure it not susceptible to any attacks. One more concern is server can see all the messages. Even private message mode can be easily hacked by some third party.

7. Conclusions & Future Work:

This specification gives brief idea about various modules in IRC like server, client and channel where multiple clients can communicate with each other in the form of text by connecting to a server via channels. The servers can connect to each other to create a network. There are several client server and channel related commands to establish connection, to terminate the connection to check connection properties.There are some security concerns as there is no encryption of messages being sent and hence the user can modify and design his protocol that would encrypt the message or to achieve some other purpose of user.

8. IANA Considerations

None.

9. References:

9.1 Normative References

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

[1] Crocker, D. and Overell, P.(Editors), "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, Internet Mail Consortium and Demon Internet Ltd., November 1997.

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

9.2 Additional References

1) irchelp.org

2) GeeksforGeeks.com

3) StackOverflow.com

10. Acknowledgments

RFC 1459 has been taken as reference to prepare this document.