CS494 Junyi Feng
Internet-Draft Portland State University
Intended status: IRC Class Project Specification May 23, 2022
Expires: November 2022

Internet Relay Chat Specification Draft

Status of this Memo

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."

Abstract

This specification draft describes the communication protocol for an IRC-style client/server system for the Internetworking Protocols class at CS494, Portland State University.

Table of Contents

1. Introduction

This specification describes a simple Internet Relay Chat(IRC) protocol by which clients can communicate with each other. This system employs a central server which 'relays' messages that are sent to it to other connected users.

2. Conventions used

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

3. Basic Information

All communication described in this protocol takes place over TCP/IP, with the server listening for connections on port 8888. Clients connect to this port and maintain this persistent connection to the server. The client can send messages and requests to the server over this open channel, and the server can reply via the same. This messaging protocol is inherently asynchronous - the client is free to send messages to the server at any time, and the server may asynchronously send messages back to the client.

4. Message Infrastructure

When a client connects to the server, the client should enter a username to join IRC. When the server successfully receives the client username, the client is prompted with a menu of commands.

```
Available commands:
/create [chatroom name]: create a new chatroom
/join [chatroom name]: join a chatroom
/msg [chatroom name] <msg>: send a public message to a
chatroom
/pm [username] <msg>: send a private message to an user
/leave [chatroom name]: leave a chatroom
/lsur [chatroom name]: print all members of a chatroom
/lsrm: print all chatrooms
/menu: print all available commands
/quit: quit the IRC
```

All commands must start with the character ‘/’ and there must be a space between the command and the arguments.

5. Client Messages

5.1 Connect to server

Enter ‘py client.py’ in the client’s terminal to connect to the server and enter the username.

5.2 Create a chat room

Enter '/create [chatroom name]' in the client's terminal to create a new chatroom. An Error message('Error. Room already exists') will be displayed if the name of the chatroom exists.

5.3 List all rooms

Enter '/lsrm' in the client's terminal to print the chatroom list.

5.4 Join a room

Enter '/join [chatroom name]' in the client's terminal to join a chatroom. An Error message('Error. Can not find room') will be displayed if the name of the chatroom does not exist.

5.5 Leave a room

Enter '/leave [chatroom name]' in the client's terminal to leave a chatroom. An Error message('Error. Can not find room') will be displayed if the name of the chatroom does not exist.

5.6 List members of a room

Enter '/lsur [chatroom name]' in the client's terminal to print the user list of a chatroom. An Error message('Error. Can not find room') will be displayed if the name of the chatroom does not exist.

5.7 Send distinct messages to multiple (selected) rooms

Enter '/msg [chatroom name] <msg>' in the client's terminal to send a message to a chatroom. An Error message('Error.

Can not find room’) will be displayed if the name of the chatroom does not exist.

5.8 Disconnect from clients

Enter ‘/quit’ in the client’s terminal to disconnect the server.

5.9 Print commands menu

Enter ‘/menu’ in the client’s terminal to print the commands menu.

5.10 Handle server crashes

- An error message(‘Port is not available’) will be prompted if the port is not available.
- An error message(‘Server closed’) will be prompted if the server shuts down.
- An error message(‘Lost connection’) will be prompted if the client loses connection with the server.

6. Server Messages

6.1 Show the client joining IRC

A message will be displayed in the server’s terminal if a client connects to the server successfully.

6.2 Show the client create a chatroom

A message will be displayed in the server’s terminal if a client creates a chatroom successfully.

6.3 Show client leaving

A message will be displayed in the server’s terminal if a client disconnects from the server successfully.

7. Error Handling

Both server and client MUST detect when the socket connection linking them is terminated. If the server detects that the client connection has been lost, the server MUST remove the client from all rooms to which they are joined. If the client detects that the connection to the server has been lost, it MUST consider itself disconnected and MAY choose to reconnect.

8. Conclusion

This specification provides a general messaging framework for multiple clients to communicate with each other through a central relay server.

9. Security Considerations

Messages sent using this system have no protection against inspection, tampering, or outright forgery. The server is able to see certain messages that are sent through the use of this service. The 'private message' may be easily intercepted by a 3rd party that is able to capture network traffic. Users wishing to use this system for secure communication should use/implement their own user-to-user encryption protocol.

10. Reference

IRC_Grading_Sheet(1).pdf

11. Acknowledgments

This document was prepared using Google Docs.

Authors' Addresses

Junyi Feng

jfeng@pdx.edu