CS 494 B. Altermatt
Internet-Draft M.Semchuk
Intended status: IRC Class Project Specification T.Fontleroy
Expires: August 1, 2021 Portland State University
February 1, 2021

Internet Relay Chat Class Project

Abstract

This specification defines an IRC client/server protocol used for text-based conferencing. This project is designed for the Internetworking Protocols class at Portland State University.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

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/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet Draft will expire on August 1, 2021.

Copyright Notice

Table of Contents

1. Introduction

This specification describes an IRC (internet relay chat) protocol for use with text-based conferencing. This protocol was developed using TCP/IP network protocols. The purpose of the IRC protocol is to facilitate the transfer of text messages between clients via a centralized server.

1.1. Servers

The server provides a point that clients may connect to in order to send messages to one another. The server will listen for connections on port 5050.

1.2. Clients

A client is anything connecting to a server that is not another server. Clients must have a unique nickname between one (1) and twelve (12) characters. Client names may not have spaces and may not begin with the ‘#’ character.

1.3. Channels

Channels are named groups of one or more clients who all receive messages addressed to that channel. A channel is created when the first client joins it, and a channel is deleted when the last client leaves it.

Channel names are strings beginning with a ‘#’ character, of a length up to 200 characters with no spaces.

2. Message Infrastructure

Generic Message Format is used to create simple text chat. The messages may or not may get a reply, as IRC is asynchronous communication type. The format allowed will be ASCII with UTF-8 encoding. The messages will be headed with a name of the room preceded by a hashtag (#), indicating to which room the message is being sent.

2.1. Label Semantics

The way that creating, sending and receiving, entering and leaving, as well as error handling will be handled with headers. Each header will be defined in a hexadecimal format. The message will be parsed by the server to figure out what to do with the incoming message. The string after will contain the rest of the data covered in the next section (3)

3. Client Messages

3.1. Server Connection

When a client will join a server, the client will send a server a header containing the name of itself (the client) in order for the server to create a room for it. The name must be valid ASCII, and not contain special characters like (#) as it is reserved for the channels.

3.1.1. Field Definitions

The string after will contain the name of the client attempting to connect. The same named clients will not be allowed to connect.

3.2. Server Disconnection

When a client voluntarily disconnects from the server, it will send a message telling the server that it is leaving and to close the connection. The server will

close the connection to the client and not send any new messages to the client after this point.
This will indicate that the client is disconnecting and no longer able to receive messages.

3.3. Listing Channels

A message with a header for listing channels will be sent to the server. The server will get a list of the current channels, and send back to the requesting client by name.
This will indicate that the server needs to get the active rooms, create a list and send it to the server requesting the information (contained in the following string data).

3.4. Joining and Creating Channels

Each user will be allowed to create a channel, and join a channel. This will be done by entering the name of the channel the user wants to join.
Upon leaving the server, any user created channel will be removed from the list of channels and no data will be saved.

3.4.1. Creating Channels

The user will send a up to 20 character (20 bytes) name along with the request, where the server will make sure no other names match. If not, the server creates a room and returns a Ok signal indicating creation of the room.

3.5. Leaving Channels

If a user wishes to leave a channel, they will have to type in the name of the server they wish to leave. The header will be: IRC_OPCODE_LEAVE_ROOM = 0x10000008. If the room name is invalid in the string then the server will send a code back to the client requesting, notifying that the name is invalid. Otherwise it will

send an acknowledgement after removing them from the room.

3.6. Listing Channel Members

Every few seconds a client process will request to update the list of channel members. This will send a label of: IRC_OPCODE_LIST_USERS_RESP = 0x10000006. A string with the clients name will indicate to whom the data will be sent back.

3.7. Sending Messages to Channels

3.7.1. Usage

Each client will be on a loop to send a “alive” message to indicate that the current connection is active. The server will be expecting this signal, and if none is received, then the connection will be considered terminated and the client is disconnected from the server.

Each Client sending a message must use the header to indicate the message and to which room the message is being sent to. A hashtag (#) is used to separate the rooms from clients. The header will be handled by the server. The server parses the header and then decides what needs to be done.

3.7.2. Field definitions

Identifiers are used in every operation and have a response as well to ensure proper connection. The message field will have a “client_name” and “room_to” field, which will be a field that the user entered at login to keep the user authenticated and make sure that the sending user is known and a valid user.

3.8. Sending Messages to Users

3.8.1. Usage

Each Client sending a message must use the header to indicate the message and to which room the message is being sent to. The header will be handled by the server. The server parses the header and then decides to which user the message is going to.

3.8.2. Field Definitions

The field will have a “client_name” and “send_to” to indicate the user's name, and a field to indicate to which other user the message is going to.

3.9. Code of conduct

Users will be expected to adhere to a basic code of conduct. Harassment, discrimination, and trolling will not be tolerated and is grounds for disconnection and/or IP address blocking. Examples of such behavior include insults, sexual harassment, threats, encouragment to harm others, disclosure of personal data (doxing), or stalking.

4. Server Messages

4.1. Client Disconnection

When a client sends a message that it will disconnect, the server sends an Ok signal to the client, closing the connection. This is to ensure that any new messages to the client are replied with an error message.

5. Error Handling

Each corresponding command has an error message to indicate failure. If an error does not fit any of the corresponding fields, then the error message will let the user know that it needs to use one of the given commands.

5.1. Usage

Messages that are not in a fail list structure, will be met with a general fault, and either the server will be notified if it is a client side fault, or the server will send an error message to the client to indicate failure.

5.2. Field definitions

“/” : is a standard message to the default room
“/list” : will list the active channels
“/leave” : will allow a user to leave the channels specified.
“/send” : will allow the user to send a message to a specific room(s).
“/members” : will allow the user to list to users in a given room
“/quit” : allows the user to shut down the current session.

6. Security Considerations

Messages sent using this system have no protection against inspection, tampering or outright forgery. The server sees all messages that are sent through the use of this service. 'Private' messaging 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.

7. IANA Considerations

None