CS594 IP Haritha Munagala
Internet Draft Susham Yerabolu
Intended status: IRC Project Specification May 17, 2018
Expires: November 2018

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 November, 2018.

Copyright Notice

Abstract

This memo describes the IRC client and server system for the CS 594 Internetworking Protocols class at Portland State University.

1. Introduction

This specification is a simple Internet Relay chat (IRC) application used by multiple users to communicate with each other via text messages in common “virtual rooms”. Client-server model is used to implement this application. Client will connect to the single central server and all the communication happens through the server. Clients can create a chat room, join the chat room or multiple chat rooms simultaneously. If a client sends a message in chat rooms it will be forwarded to all the clients who are part of that chat room. Also the client can anytime leave the chat room. In addition to these, the application has features of one to one chat i.e. private and file transfer between individual clients. The reliable communication is implemented using TCP protocol.

1.1 Server

The server forms the backbone of IRC, providing a point to which clients may connect to talk to each other. Server provides all the functionalities that are necessary for application to work properly such as storing the information about the clients, delivering the messages to clients efficiently and reliably. The important requirement from the server is for it to be available at all the times.

1.2 Client

A client is the one who wants to send the message to group of people or an individual. Each client has ‘name’ which is unique and used to identify the client. Clients have the option of personal chat or group chat to communicate over network. The Client must register on the server by providing its details. The server will use these details to send and receive the message to and from the client.

1.3 Chat Room

A Chat Room is a named group of more than one clients connected to the server. Multiple clients can send and receive messages in the chat room. A message sent in chat room is broadcasted to all other clients in chat room. Clients can view the list of available chat rooms and can join any of them. For a chat room to be active it must have at least one client. The chat room will be terminated when there are no clients left in it. The chat room is identified and referenced by a unique name. The unique name is used to send all the message that are directed to a particular room. The channel is created implicitly when the first client joins it, and the channel ceases to exist when the last client leaves it. While channel exists, any client can reference the channel using the name of the channel.

2.Message Format

Message format will consist of opcode and length header fields. The opcode field will determine what type of message its containing - whether its a join request/leave request and the message client length field will specify how many bytes of payload follow the header in the message. The server to which a client is connected is required to parse the complete message, returning any appropriate errors. If the server encounters a fatal error while parsing a message, an error must be sent back to the client.

2.1.Operation codes:

```
IRC_OPCODE_ERR = 01
IRC_OPCODE_KEEPALIVE = 02
IRC_OPCODE_HELLO = 03
IRC_OPCODE_LIST_ROOMS = 04
IRC_OPCODE_LIST_ROOMS_RESP = 05
IRC_OPCODE_LIST_USERS_RESP = 06
IRC_OPCODE_JOIN_ROOM = 07
IRC_OPCODE_LEAVE_ROOM = 08
IRC_OPCODE_SEND_MSG = 09
IRC_OPCODE_TELL_MSG = 10
IRC_OPCODE_SEND_PRIV_MSG = 11
IRC_OPCODE_TELL_PRIV_MSG = 12
```

3.Error codes

They specifies the type of error that occurred.

```
IRC_ERR_UNKNOWN = 50
IRC_ERR_ILLEGAL_OPCODE = 51
IRC_ERR_ILLEGAL_LENGTH = 52
IRC_ERR_WRONG_VERSION = 53
IRC_ERR_NAME_EXISTS = 54
IRC_ERR_ILLEGAL_NAME = 55
IRC_ERR_ILLEGAL_MESSAGE = 56
IRC_ERR_TOO_MANY_USERS = 57
```

IRC_ERR_TOO_MANY_ROOMS = 58

4.Error handling

Both server and client MUST detect when the socket connection linking them is terminated, either when actively sending traffic or by keeping track of the heartbeat messages. 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.