Internet Relay Chat Protocol

Status of This Memo:

This memo is a specification for an IRC protocol for the CS 594 Internetworking Protocols course at Portland State University in the RFC style.

Abstract:

This document is a specification for an Internet Relay Chat Protocol. This protocol is loosely modeled after the protocol specified in the RFC 1495, with a reduced and simplified message set. The goal of this protocol is to provide basic chat room functionality for multiple clients connected to a single server, with the ability to create, join, message, etc. within ‘rooms’ inside the chat interface.

Table of Contents

[1. Introduction 2](#_Toc484172140)

[1.1 Servers 2](#_Toc484172141)

[1.2 Clients 2](#_Toc484172142)

[1.3 Rooms 3](#_Toc484172143)

[2. Conventions and Basic Information 3](#_Toc484172144)

[2.1 Naming 3](#_Toc484172145)

[3. Messages 4](#_Toc484172146)

[3.1 Client Messages 4](#_Toc484172147)

[3.1.1 CREATE 4](#_Toc484172148)

[3.1.2 JOIN 4](#_Toc484172149)

[3.1.3 SEND 4](#_Toc484172150)

[3.1.4 LEAVE 4](#_Toc484172151)

[3.1.5 QUIT 5](#_Toc484172152)

[3.1.6 LIST 5](#_Toc484172153)

[3.1.7 NICKS 5](#_Toc484172154)

[3.1.8 NNICK 5](#_Toc484172155)

[3.1.9 WHO 5](#_Toc484172156)

[3.1.10 COMMANDS 6](#_Toc484172157)

[3.1.11 PVTMSG 6](#_Toc484172158)

[3.2 Server Messages 6](#_Toc484172159)

[3.2.1 ERR\_ROOMEXISTS 6](#_Toc484172160)

[3.2.2 ERR\_ROOMNOTEXIST 6](#_Toc484172161)

[3.2.3 ERR\_NOTMEMBER 6](#_Toc484172162)

[3.2.4 ERR\_INVALIDCOMMANDFORMAT 6](#_Toc484172163)

[3.2.5 ERR\_INVALIDNICK 6](#_Toc484172164)

[3.2.6 Welcome Message 7](#_Toc484172165)

[3.2.7 LIST Response 7](#_Toc484172166)

[3.2.8 WHO Response 7](#_Toc484172167)

[3.2.9 NNICK Error 7](#_Toc484172168)

[3.2.10 Client Leaving 7](#_Toc484172169)

[3.2.11 Invalid Command Format 7](#_Toc484172170)

[3.2.12 Command Not Found 7](#_Toc484172171)

[4. Conclusion 7](#_Toc484172172)

[5. References 7](#_Toc484172173)

# Introduction

The IRC protocol contained in this document is based in large part on the original RFC 1459 from 1993. This version is designed for a single server connected to multiple clients. The clients are able to create nicknames, create and join rooms, and communicate with other users via those rooms or private messages. A message sent to a room that a client is a member of is relayed through the server to all of the other members of that same room.

## Servers

There is a single server in this IRC implementation. All messages from all clients are relayed through this one server. The port for this server is specified as Port 6000. The server tracks all connected clients and their nicknames, created rooms, and the relaying of message both through rooms and private messages. The server will notice if clients become inactive, consider the connection terminated, and cleanup all entries related to that client.

## Clients

Multiple clients are supported. Each client must connect to Port 6000 on the server. Once connected to the server a client will set a unique nickname for that IRC session. Then a client can join or create rooms and begin sending messages in those rooms or private messages. The first message a client should send after connecting to the server is NNICK to set a nickname for that IRC session. A client will detect if the connection to the server has failed or the server has become unresponsive and will terminate the session at that point.

## Rooms

Rooms within the IRC interface are created by the clients. A client must join at least one room before sending messages, with the exception of private messages. Any message sent by any client within a room will be relayed to all other clients connected to that room. Room names must be unique within a single IRC session. A room will be considered active until the IRC session terminates.

# Conventions and Basic Information

This IRC protocol closely follows the message formatting specified in RFC 1459 and RFC 2812. A message has between one and three fields, <command>, <parameters>(optional), and <parameters>(optional) separated by the ‘%’ symbol. The command field will contain the action that the client would like to perform (CREATE, JOIN, SEND, etc.). Commands will be listed in all capital letters. The parameters field will contain any other necessary information, for example the name of the room that the client would like to join. <crlf> denotes the carriage return/linefeed character, ASCII 0xA, and all messages will terminate with that character. “spaces” are defined as ASCII 0x20; tab (0x9) and newline (0xA) are not considered “spaces”.

Examples:

<message> ::= <command><space>’%’<parameters><space>’%’<parameters><crlf>

<parameters> ::= [ <room name> | ‘%’ <message content>]

CREATE %OregonFishing - denotes a client would like to create the room “OregonFishing”.

SEND %OregonFishing %How is the fishing today? – denotes the client sending the message “How is the fishing today?” to the group OregonFishing.

### Naming

The following are the naming conventions that are adhered to in this protocol. Nicknames will be ASCII characters ‘a’ – ‘z’, ‘A’ – ‘Z’, or ‘0’ – ‘9’; no other characters are permitted. Nicknames will have a maximum of 10 characters.

# Messages

## Client Messages

### CREATE

Command: CREATE

Parameters: <room name>

<CREATE> ::= ‘CREATE’<space>[’%’<room name>]<crlf>

Example: CREATE %OregonBiking

The CREATE command creates a new chat room and automatically adds the creator to that room. The room name must be unique. If the room name parameter matches an existing room name the server will reply with ERR\_ROOMEXISTS and will not add the new room. The client can then proceed as they wish.

### JOIN

Command: JOIN

Parameters: <room name>

<JOIN> ::= ’JOIN’<space>[’%’<room name>]<crlf>

Example: JOIN %OregonBiking

The JOIN command is a request by the client to join an existing room. If the room exists then the server adds the client to the list for that room and notifies the members that the client has joined the room. If the room does not exist the server will reply with ERR\_ROOMNOTEXIST and then allow the client to proceed as they wish.

### SEND

Command: SEND

Parameters: <room name><message>

<SEND> ::= ‘SEND’<space>[’%’<room name>]<space>[’%’<message>]<crlf>

Example: SEND #OregonBiking :What great biking weather today!

The SEND command is a message from the client to the server, and the server will relay the message to all of the other members of that room. If the room does not exist the server will reply with ERR\_ROOMNOTEXIST. If the client is not a member of that room the server will reply with ERR\_NOTMEMBER. The total message content must be less than 1024 characters. The end of message is defined to be the <crlf>.

### LEAVE

Command: LEAVE

Parameters: <room name>

<LEAVE> ::= ’LEAVE’<space>[’%’<room name>]<crlf>

Example: LEAVE %OregonBiking

The LEAVE command is a request by the client to leave a particular room. The server will delete that client from the room list and inform the other members of that room of the departure. If the client is not a member of that room then the server will reply ERR\_NOTMEMBER.

### QUIT

Command: QUIT

Parameters: none

Example: QUIT<crlf>

The QUIT command is a request by the client to withdraw from the IRC session. The server will terminate the connection and notify the members of that client’s rooms of his departure.

### LIST

Command: LIST

Parameters: None

Example: LIST<crlf>

The LIST command is a request by the client to view a list of active rooms in the chat space. The server will reply with a list of active rooms.

### NICKS

Command: NICKS

Parameters: None

Example: NICKS<crlf>

The NICKS command is a request by the user to view all active members of the chat space. The server will reply with the nicknames of all active members.

### NNICK

Command: NNICK

Parameters: <nickname>

<NNICK> ::= ‘NNICK’<space>[‘%’<nickname>]<crlf>

Example: NNICK %lblakely

This NNICK is the first command that a user MUST use after a connection has been made with the server; the NNICK is a request for a new nickname for this chat session. The server will reply with WELCOME.

### WHO

Command: WHO

Parameters <room>

<WHO> ::= WHO<space>‘%’<room><crlf>

Example: WHO %OregonBikers

The WHO command is a command to list the users in the given room. The standard reply is ‘<username><space>’/’<space><username> <space>’/’<space><username> ‘are in room’ <room>’. If the room does not exist the error message ‘ERR\_ROOMNOTEXIST’ will be given.

### COMMANDS

Command: COMMANDS

Parameters: None

<COMMANDS> ::= ‘COMMANDS’<crlf>

This command will cause the server to send the client a list of commands for the IRC.

### PVTMSG

Command: PVTMSG

Parameters: <nickname>, <message>

<PVTMSG> ::= ‘PVTMSG’<space>’%’<nickname><space>’%’<message><crlf>

Example: PVTMSG %logan %Hello!

The PVTMSG command is used to send a private message to another user. Any user can send a message to any other user. If the user does not exist the server will reply ‘ERR\_NICKNOTEXIST’.

## Server Messages

The server will generally only respond to messages sent to the client, and relay the chat messages across the chat. Those replies will be acknowledgements of client actions or error messages. The server will assume client whose socket is readable but has sent no data has disconnected from the IRC and the server will delete that client’s information and inform the remaining clients of the departure.

### ERR\_ROOMEXISTS

ERR\_ROOMEXISTS is an error message generated by the server when a client attempts to create a room that already exists.

### ERR\_ROOMNOTEXIST

ERR\_ROOMNOTEXIST is an error message generated by the server when a client attempts to join a room that does not exist or when a client attempts to send a message to a room that does not exist.

### ERR\_NOTMEMBER

ERR\_NOTMEMBER is an error message generated by the server when a client attempts to send a message or leave a room where they are not a member.

### ERR\_INVALIDCOMMANDFORMAT

ERR\_INVALIDCOMMANDFORMAT is an error message generated by the server when a client sends a command with invalid parameters

### ERR\_INVALIDNICK

ERR\_INVALIDNICK is an error message generated by the server when a client attempts to set a nickname longer than 10 characters

### Welcome Message

The Welcome Message is a message sent to the client upon first joining the IRC

### LIST Response

In response to the LIST command, the server responds with a list of all existing chatrooms.

### WHO Response

In response to the WHO command, the server responds with a list of users in the specified chatroom

### NNICK Error

“You must create a user name before doing anything else.” Is sent if a user attempts to execute a command other than NNICK as their first action in the IRC.

### Client Leaving

‘<nickname> has left <room>’ is sent when a client has left a room for any reason.

### Invalid Command Format

‘ERR\_INVALIDCOMMANDFORMAT’ is sent if the client sends a message that does not conform to the message guidelines.

### Command Not Found

‘ERR\_COMMANDNOTFOUND’ is sent if the client uses a command that is not in the command list.

# Conclusion

This document provides the initial framework for a simplified IRC protocol.

# References

Oikarinen & Reed, “Internet Relay Chat Protocol”, RFC 1459, May 1993.