CS 594 – INTERNET WORKING PROTOCOLS

Name : Saiteja Garlapati PSU ID : 985031204
Topic : IRC Class Project Specification
Teammates : Sriharshitha Ayyalasomayajula, Srija Gaddam

Project : Internet Relay Chat (IRC) Class Project

Status of this memo

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 rendered obsolete 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."

i) The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

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

Abstract

This memo describes the communication protocol for a client/server system based IRC project for the Internetworking Protocols. The purpose of this RFC is to outline the experimental protocols for programming projects used to develop Internet Relay Chat (IRC) servers and client applications. The intended result is to mimic some very basic IRC functionality in the protocol. The protocol defines the responsibilities of the server and the client as well as how they will communicate. The protocol is heavily weighted towards the server.

TABLE OF CONTENTS

1. Introduction....3
2. Conventions used in this document....3
3. Basic Information....3
4. Registration....3
5. Messages....3
   5.1 Generic Message Format....3
      5.1.1 Field Definitions.... 4
      5.1.2 Intents....4
   5.2 Error Messages....4
      5.2.1 Usage....4
      5.2.2 Field Definitions....4
      5.2.3 Errors....4
6. Naming Conventions....5
7. Client Intents....5
   7.1 Connecting to the server....5
      7.1.1 Usage....5
   7.2 Listing....6
      7.2.1 Usage....6
      7.2.2 Response....6
   7.3 Joining and Creating Rooms....6
      7.3.1 Usage....6
   7.4 Leaving a Room....6
      7.4.1 Usage....6
   7.5 Sending Messages....6
      7.5.1 Usage....7
      7.5.2 Field Definitions....7
      7.5.3 Message Types....7
   7.6 File Transfer....7
      7.6.1 Usage: set-file-transfer-key....7
      7.6.2 Usage: send-file....7
   7.7 Getting Help....8
      7.7.1 Usage....8
   7.8 Disconnecting from Server....8
      7.8.1 Usage: socket.close()....8
      7.8.2 Usage: quit-irc....8
8. Server Messages....8
   8.1 Listing Response....8
   8.2 Forwarding Messages to Clients....8
   8.3 Notifications....9
9. Error Handling....9
10. Extra Features....9
   10.1 Private Messaging....9
   10.2 Secure Messaging....9
   10.3 File Transfer....9
11. Security Considerations....9
12. Conclusion & Normative References....10

1. Introduction

This is a protocol specification for a simple Internet Relay Chat (IRC) Protocol. This system deploys a single central server process that allows clients to communicate with each other by forwarding or “relaying” the messages to intended recipient.

Users can join one or more existing chat rooms, create new chat rooms, send and receive messages from other users sent to the chat room to which they are subscribed.

Users can also send private messages and secure messages to other users, send broadcast messages to all other users, and send files to each other.

2. Conventions used in this document

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].
In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying significance described in RFC 2119.

3. Basic Information

The communication in this IRC protocol happens over TCP/IP. The server process constantly keeps listening at port 1234 for any incoming requests for connection from clients. The clients and server communicate through messages. The connection is persistent and stays open until either the client or server chooses to close it. The server can serve multiple requests, from multiple clients at a time with no adverse effect on any of the client processes.

4. Registration

Each user MUST register with the IRC server with a unique username. Only registered users will be able to use IRC services. If the chosen username already exists, the server notifies the client and asks to choose another username. The username MUST be at least 1 character and at most 20 characters.

5. Messages

5.1 Generic Message Format

<intent> <target> <your message here>

5.1.1 Field Definitions

- intent: specifies the intent of the user and what action needs to be taken.
- target: specifies the recipient of the message, either a chat room or an individual host.
- message: the message to be sent, MUST be a string.

5.1.2 Intents

- Join-room
- Exit-room
- Chat-room
- Private-messaging (pvt-msg)
- Secure-messaging (secure-msg)
- Decryption-key
- List
- My-room
- Set-file-transfer-key
- Send-file
- Broadcast
- Quit-IRC
- Help

5.2 Error Messages

<error code> <error message>

5.2.1 Usage

It can be sent from either the client or the server, explaining the cause of the error. Depending on the type of error that occurred, the server may allow the client to rectify the error or terminate the connection.

5.2.2 Field Definitions

- error code : specifies the type of error that occurred
- error message : briefly describes the error

5.2.3 Errors

- RE001 : Username already exists

- NE001 : Username cannot have any spaces
    - Try a different username
- NE002 : Username must have at least one character
- NE003 : Username can have maximum length of 20
- NE004 : Room name can have maximum length of 10
- AE001 : Too many arguments given
- AE002 : Too few arguments given
- AE003 : Username <username> does not exist
- AE004 : Invalid value of argument/s for given command
- ME001 : Unauthorized message
- ME002 : Message size bigger than permitted
- CE001 : Invalid command
- CE002 : This command can only be executed once for given set of arguments
- IOE01 : IRC could not open file
- FTE01 : User has not set a file transfer key

6. Naming Conventions

Naming conventions apply to both the client's usernames and the chat room names. All names MUST be validated as follows:

- The name should be a character string.
- The name should have at least 1 character, at most 20 characters for username and at most 10 characters for room name.
- No spaces are allowed.
- Each name must be unique.

7. Client Intents

7.1 Connecting to the server

socket.connect(host_id, port)

7.1.1 Usage

The first thing that a client does is to request the server to connect using host_id and port number. The client can select a username only after the connection is established. The requested username MUST follow naming conventions.

7.2 Listing

- list users
- list rooms
- list members <room-name>

- my-rooms

7.2.1 Usage

Client can request a list of all active users, list of open rooms (for a room to be open, there must be at least one member subscribed to it.), a list of members in a particular room, or list of subscribed rooms.

7.2.2 Response

The server MUST return the requested list to the user.

7.3 Joining and Creating Rooms

join-room <room-name>

7.3.1 Usage

Client sends this message to server, if a room of the same name exists, the server MUST notify all members of the group about the new addition to the group. The server MUST notify the new member that the room already existed. This happens every time someone joins a room. If no such room exists, one is created and the requesting client is the only member in the member list of that room. The requested room name MUST follow the naming conventions. The client may use this message multiple times to join multiple chat rooms simultaneously.

7.4 Leaving a room

exit-room <room name>

7.4.1 Usage

Client sends this message to server to exit a room. If the client is subscribed to the room in question then the server MUST remove client from that room and MUST send a member list update notification to everyone else. If the client is not a member then it gets an Argument Error message.

7.5 Sending Messages

<message-type> <target> <your message here>

7.5.1 Usage

Sent by a client to the server, to be forwarded to either a specified chat room, an individual user, or all active users. When the server receives a valid message like this, it MUST relay it to all the intended recipients. The message MUST inform all recipients who is the sender of the message. If a client attempts to send message to a room it's not subscribed to, the server must discard that message and ask the client to join the room to be able to send messages.

7.5.2 Field Definitions

- message-type: specifies the type of the message user intends to send.
- target: specifies the recipient of the message, either a chat room or an individual host.
- message: the message to be sent, MUST be a string.

7.5.3 Message Types

- chat-room <room-name> <your message here>
- pvt-msg <recipient-name> <your message here>
- broadcast all <your message here>
- secure-msg <recipient-name> <key> <your message here>

7.6 File Transfer

set-file-transfer-key

7.6.1 Usage

Sent by client to the server to set a file transfer key. A client MUST set a file transfer key in order to receive files from other clients. A client MUST know the recipient's file transfer key in order to send a file. A client can change it's file transfer key at any time and as many times as desired.

send-file <file-name.extension> <recipient-name>

7.6.2 Usage

Sent by client to server in order to send a specific file to another user. The sender is then prompted for entering receiver's file transfer key. The server MUST verify the validity of the entered key, if the key is correct the file transfer proceeds, otherwise the sender get notified of the failure. The server MUST notify the client when it receives a file from another user.

7.7 Getting Help

help

7.7.1 Usage

This command displays all possible intents, their purpose and the usage syntax. The users may refer to it when they do not know or do not remember the correct syntax of a command.

7.8 Disconnecting from Server

socket.close()

7.8.1 Usage

Used by the server to disconnect the client without any notification. The server SHOULD discover that the client has left through Probe messages. When such an event occurs, the server MUST remove the client from all the rooms it was subscribed to and send user list update notification to other users.

quit-irc

7.8.2 Usage

Sent by client to the server requesting to terminate its connection. On receiving this request, the server MUST immediately disconnect the client, remove it from the chat rooms it had subscribed to, and send user list update notification to other users. Any further communication from that client MUST require reconnecting to server.

8. Server Messages

8.1 Listing response

The server MUST send the requested list to the client in response to a valid listing request.

8.2 Forwarding Messages to Clients

The server MUST forward each valid incoming message to the target chat room or individual user.

8.3 Notifications

The server MUST send appropriate notifications to clients whenever it is required.

9. Error Handling

In case of a connection failure, either from server or from client, the other party MUST detect it. If the client connection is lost the server MUST remove it from the chat rooms it had subscribed to, and send user list update to other users. Any further communication from that client MUST require reconnecting to the server. If the client detects that the connection to server is lost, it MUST consider itself disconnected and MAY choose to reconnect.

10. Extra Features

10.1 Private Messaging

The users can send private messages to each other. Only the intended recipient receives this message, no one else.

10.2 Secure Messaging

The users can send secured messages to each other. The receiver must have the key to be able to decrypt the received encrypted message. Secure Messaging feature is only available on linux servers.

10.3 File Transfer

The users can send files to each other. The sender must have the receiver's file transfer key to be able to send files. If a client does not wish to receive files from anyone, it may simply choose not to set it's file transfer key. The key may be changed as many times as desired.

11. Security Considerations

This system does not employ any security mechanisms. The communication is vulnerable to any sort of malicious intents and actions. The central server is responsible for forwarding all messages and able to read all of them, hence somewhat undermining the concept of "private" messaging in this system. Even the secure messaging is just secure enough to conceal the message from an ephemeral viewer but not secure enough to withstand a deliberate and malicious attack.

12. Conclusion

This specification describes a simple message passing framework for multiple clients to communicate with each other through message forwarding performed by a central server.

12.1.1 Normative References

[1] Bradner, S., “Key words for use in RFCs to Indicate Requirements Levels”, BCP 14, RFC 2119, March 1997.

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