CS594

Internetworking Protocol

Intended status: IRC Class Project Specification

November 29th, 2018  
Ching-Wei Lin

Internet Relay Chat Room Project

Status of this Memo

This is the initial version of the 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.

The list of current project can be accessed at <https://github.com/wning0101>

Copyright Notice

Copyright (c) 0000 IETF Trust and the persons identified as the document authors. All rights reserved. The code can be used with author’s permission.

Abstract

Nowadays, online chatting room is super common. The document describes a simple version of chatting platform, which can be accessed by multiple user in multiple rooms. The main goal of this project is gaining more hand-on experience on networking protocol application and being more familiar with internet connection.

Table of Contents

[Introduction 3](#_Toc531285798)

[Basic Information 3](#_Toc531285799)

[Client Message 3](#_Toc531285800)

[First message sent to the server 3](#_Toc531285801)

[Listing Rooms 4](#_Toc531285802)

[Joining and Creating Rooms 4](#_Toc531285803)

[Leaving a Room 4](#_Toc531285804)

[List user which in the room 4](#_Toc531285805)

[Sending Messages 4](#_Toc531285806)

[Sending Message to selected room 5](#_Toc531285807)

[Private Message 5](#_Toc531285808)

[List all the rooms with members 5](#_Toc531285809)

[Server Message 5](#_Toc531285810)

[Listing Response 5](#_Toc531285811)

[First message management 5](#_Toc531285812)

[Create Rooms 6](#_Toc531285813)

[Join Rooms 6](#_Toc531285814)

[Leaving a Room 6](#_Toc531285815)

[Forwarding Messages to Clients 6](#_Toc531285816)

[Error Handling 6](#_Toc531285817)

[Same user name 6](#_Toc531285818)

[Private message 6](#_Toc531285819)

[Leave a room 6](#_Toc531285820)

[Join a room 6](#_Toc531285821)

[Send message to selected room 7](#_Toc531285822)

[Crash Handling 7](#_Toc531285823)

[Extra Feature 7](#_Toc531285824)

[Conclusion and Future Work 7](#_Toc531285825)

[Security Consideration 7](#_Toc531285826)

[Acknowledgment 8](#_Toc531285827)

# 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.

Users can join rooms, which are groups of users that are subscribed to the same message stream. Any message sent to that room is forwarded to all users currently joined to that room. User can list all the members in the room and all the rooms current existence.

Users can also send private messages directly to other users or join multiple rooms at the same time.

# Basic Information

All communication described in this protocol takes place over TCP/IP, with the server listening for connections on port 8088. 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. The client is free to send messages to the server at any time, and the server may send messages back to the client. Both the server and client may terminate the connection at any time for any reason.

The application is written by python with python sockets.

The server may choose to allow only a finite number of users and rooms, depending on the implementation and resources of the host system.

# Client Message

First message sent to the server

Before subsequent messages can be sent, a connecting client must provide a user name and their socket connection.

The server must associate the client's chat name with the socket connection of the user. The user name can only be used once; if the server receives the username more than once, the server would terminate the client's connection.

Listing Rooms

Sent by the client to request a list of all of the rooms currently.

Usage: “roomlist”

Joining and Creating Rooms

Sent by the client to join a chat room. If no room by that name exists, nothing happens. Sent by the client to create a chat room. There is a default main chat room existing at the very beginning and the following chat room will be name room1, room2, room3 … etc. The server will check the command and do the responding action. If a new room is created, the user list for the room will be created at the same time.

Usage:

“join” and wait for the prompt and “the name of room that you want to join”

“create”

Leaving a Room

Sent by the client to leave a chat room. If a new client leaves a room, the user list will be modified at the same time. If the user is not in that room, return a error message.

Usage:

“leave” and wait for the prompt and “the name of room that you want to leave”

List user which in the room

Sent by the client to request a list of all of the users currently joining the room.

Usage:

“memberlist” and wait for the prompt and “the name of room that you want display its list”

Sending Messages

Sent by a client to send a text message to either a room or another user.

If the message is for a room, after validating this message, the server sends a message to all users in the specified room and the parameter set to the name of the user who sent the message.

Usage: just enter the message you want to send except the commands

Sending Message to selected room

Sent by a client to send a text message to the selected room. Any user in that room would receive the message, otherwise, they don’t.

Usage:

“selectroom” and wait for the prompt and “the name of room that you want to send message” and then the message you want to send

Private Message

If the message is for another user, then the target is another user and, after validating this message, the server sends a message to that specific user with the forwarded message and the parameter set to the name of the user who sent the message.

Usage: “private” and wait for the prompt and “the name of user that you want send message” and then enter the message you want to send

List all the rooms with members

Sent by the client to request a list of all of the rooms with their members.

Usage: “printall”

# Server Message

Listing Response

Generic listing response message sent by the server to inform the client of a list. Used for both listing rooms and listing users in a room.

First message management

The server handles the first message by adding the user name and its socket connection to each list. Then, the server will know all the user who connects to the server and also their connection.

Create Rooms

There’s a dictionary of rooms which has its own list recording the member in it. Therefore, we can add a new value to the dictionary to create a new room.

Join Rooms

Likewise, the server allows users joining a room by adding them to the selected list in the dictionary.

Leaving a Room

the server allows users joining a room by removing them from the selected list in the dictionary.

Forwarding Messages to Clients

If the message is for a room, then this is a forwarded message from the server to a room the client is joined to. Server sets the name of the room to which the message belongs. Clients ignores this message if the name does not match one of the rooms the client joined to. If the message is for another user, then this is a forwarded private message from another user intended.

# Error Handling

Same user name

If a client selects a user name which has already been used, the server send an error message to client and disconnects it.

Private message

If a client sends a private message to a user who doesn’t connect to the server, the server sends an error message to the client.

Leave a room

If a client tries to leave a room which he is not in it, the server sends an error message to the client.

Join a room

If a client tries to join a room which he is already in it, the server sends an error message to the client.

Send message to selected room

If a client tries to send message to selected room which doesn’t exist, the server sends an error message to the client.

# Crash Handling

Both server and client 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 removes the client from all rooms to which they are joined. If the client detects that the connection to the server has been lost, it considers itself disconnected.

# Extra Feature

Sending Private message is my extra feature, which only the selected user can see the message.

# Conclusion and Future Work

This is my first python socket program. At the beginning, I knew little of it. Barely know how to start it, but after I devoted myself in it. I can I am capable of writing my own socket program which functions pretty well. However, this program can only be operated in terminal, so it’s not quite convenient to use. For my future work, I would like to build a user friendly interface, such as website or mobile application for user’s convenience.

# Security Consideration

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.

# Acknowledgment

1. CS594SampleRFC.pdf (provided by instructor)
2. https://www.geeksforgeeks.org/simple-chat-room-using-python/