MyTeams Protocol Documentation

Introduction

The **myTeams** project is a collaborative communication application aimed at facilitating team-based discussions and personal messaging. It consists of a server and a Command Line Interface (CLI) client implemented in C programming language. The communication between the server and clients is established using TCP sockets.

Project Overview

myTeams allows users to create, join, and leave teams, create channels within teams, start threads in channels, and engage in personal discussions. The server is responsible for managing user authentication, team subscriptions, message storage, and event handling. The CLI client provides a set of commands to interact with the server and perform various actions such as logging in, sending messages, subscribing to teams, creating threads, and more.

Server Features:

- Handles multiple client connections simultaneously using select for command management.
- Persists internal information upon shutdown and loads it upon startup.
- Implements a collaborative communication application similar to Microsoft Teams®, organizing discussions into threads within channels.

CLI Client Features:

- Provides commands to interact with the server, including logging in, sending messages, subscribing to teams, creating threads, and more.
- Displays responses from the server and handles user input for executing commands.

Data Transfer Protocol

This section describes the data transfer protocol used in the MyTeams application for transmitting messages and other information between the server and the client.

Data Transfer Format

The data transfer format consists of structured messages exchanged between the server and the client. Each message comprises a header and payload, where the header provides metadata about the payload.

Message Header Format

- **Command**: Indicates the type of data being transferred.
- **Length**: Specifies the length of the payload in bytes.

Message Payload Format

- **Payload Content**: Actual content of the data being transferred, such as UUIDs, names, messages, statuses, and timestamps.

Data Transfer Process

- **Server to Client**:
  1. The server constructs a message by concatenating the command type and the content of the data being sent.
  2. The server sends the message to the client over the network.
- **Client**:
  1. The client receives the message from the server.
  2. The client parses the message, extracts the command type, and determines the appropriate action to take based on the command.
  3. The client processes the content of the message according to the command type.

Send data to client

- **UUID:** Send "UUID <SP>" followed by the UUID you want to transmit.
- **NAME:** Send "NAME <SP>" followed by the name you want to transmit.

- **MESSAGE:** Send "MESSAGE <SP>" followed by the message body you want to transmit.
- **STATUS:** Send "STATUS <SP>" followed by the status you want to transmit.
- **TIMESTAMP:** Convert your time_t into a string, then send "TIME <SP>" followed by the string representation of the timestamp you want to transmit.

Send command to execute in the client

Send "COMMAND" in uppercase without spaces to the client after sended all informations needed for the command. The available commands are:

- **"LOGGED":** Requires UUID and NAME before sending.
- **"NOLOGGED":** Requires UUID and NAME before sending.
- **"USERS":** Requires UUID, NAME, and STATUS before sending.
- **"SEND":** Requires UUID and MESSAGE before sending.
- **"ONEUSER":** Requires UUID, NAME, and STATUS before sending.
- **"PRINTMESS":** Requires UUID, MESSAGE, and TIMESTAMP before sending.

Take care to send carefully all information that the command need before actually send the command!

All Commands available:

/help <CRLF>: Show help

/login <SP> “<username>” <CRLF> : set the user_name used by client

/logout <CRLF> : disconnect the client from the server

/users <CRLF> : get the list of all users that exist on the domain

/user <SP> “<user_uuid>” <CRLF> : get details about the requested user

/send <SP> “<user_uuid>” <SP> “<message_body>” <CRLF> send a message to specific user

/messages <SP> “<user_uuid>” <CRLF> : list all messages exchanged with the specified user

/subscribe <SP> “<team_uuid>” <CRLF> : subscribe to the events of a team and its sub directories (enable reception of all events from a team)

/subscribed <SP> “<team_uuid>” <CRLF> : list all subscribed teams or list all users subscribed to a team

/unsubscribe <SP> “<team_uuid>” <CRLF> : unsubscribe from a team

/use <SP> “<team_uuid>” [<SP> ”<channel_uuid>”] [<SP> ”<thread_uuid>”] <CRLF> : Sets the command context to a team/channel/thread

/create <CRLF>: create a team, a channel, a thread or a reply based on the context you are currently in

/list <CRLF> : list all teams, channels, threads or replies based on the context you are currently in

/info <CRLF> : display details of the users, the selected team, the selected channel or the selected thread based on the context you are currently in

<SP> = space, <CRLF> = end-of-line, [”<argument>”] = optional argument

All Reply Codes:

Logs and users reply codes:

1xx: base code for logs and users commands

102 client connected to the server

120 user logged successfully

122 user already logged in

124 bad argument for /login command

140 user successfully disconnected

160 list of users retrieved successfully

162 user details retrieved successfully

170 list of commands retrieved successfully

Messages sending reply codes:

2xx: base code for messages commands

202 message sent successfully

224 bad arguments for /send command

226 user not found for this message

240 list of messages retrieved successfully

244 no message exchange found with this user

Subscribe commands reply codes:

3xx: base code for subscribe commands

302 subscribed successfully

303 can’t subscribe, team not found

320 unsubscribed successfully

322 you are not subscribed to this team

326 can’t unsubscribe, team does not exist

340 list of subscribed teams retrieved successfully

343 can’t list subscribed users, you are not in this team

346 can’t list subscribed users, team does not exist

Context commands reply codes:

4xx: base code for Context commands

402 context set successfully

403 bad arguments for /use command

404 context not found

412 creation successful

413 bad arguments for /create command

420 list retrieved successfully

430 information retrieved successfully

Errors reply codes:

5xx: base code for errors reply

530 user not logged in

550 server closed the connection