Table of Contents

[**1**](https://tools.ietf.org/html/rfc1459#section-1)**. INTRODUCTION…………………………………………………………………………………………………….P2**

### [**2**](https://tools.ietf.org/html/rfc1459#section-1.1)**. Servers………………………………………………………………………………………………………………….p2**

### **3. Clients………………………………………………………………………………………………………………….p2**

**4. Teams……………………………………………………………………………………………………………………….p3**

### **5. Commands……………………………………………………………………………………………………………….p3**

### **6. Commands list………………………………………………………………………………………………….p4**

**7. Users interactions……………………………………………………………………………….p4-5**

**[1](https://tools.ietf.org/html/rfc1459" \l "section-1). INTRODUCTION**

The goal of this project is to create a server and a CLI client.

My\_Teams is a communication and community management application. The purpose of this project is to create our own protocol and describe it in a RFC’s style documentation, to create our own data model in compliance with the given library technical properties. We need to implement requested commands in the CLI client and use the given server and client libraries to print every events and data. The network communication will be achieved using TCP sockets.

The File Transfer Protocol (FTP) is a standard [network protocol](https://en.wikipedia.org/wiki/Network_protocol) used for the transfer of [computer files](https://en.wikipedia.org/wiki/Computer_file) between [a client and server](https://en.wikipedia.org/wiki/Client%E2%80%93server_model) on a [computer network](https://en.wikipedia.org/wiki/Computer_network).

FTP is built on a client-server model architecture using separate control and data connections between the client and the server.

### **[2](https://tools.ietf.org/html/rfc1459" \l "section-1.1). Servers**

The server forms the backbone of the my\_teams, providing a point to which clients may connect to to talk to each other forming an FTP network. Clients communicate with the server which communicates with clients

[My\_Teams server]

/ | \

/ | \

/ | \\_\_\_\_

/ | \

[Client 1] [Client 2] [Client 3]

[Format of FTP My\_Teams Server Network]

### 

### **3. Clients**

A client is anything connecting to a server that is not another server. Each client is distinguished from other clients by a unique nickname having a maximum length of 32 characters and a unique uuid. See the protocol grammar rules for what may and may not be used in a nickname

### 

### **4. Teams**

A team is a named group of one or more clients which will all receive messages addressed to that team. A team can be created using the /create command and cannot be destroyed. the command /subscribe makes it possible to join a team and /unsubscribe to leave it. The command /subscribed makes it possible to list the teams to which the client is assigned and to display the other clients who are registered to.

Teams names are strings of length up to MAX\_NAME\_LENGTH constant. The only restriction on a channel name is that it may not contain any doubles cotes ('”').

There is one type of channel allowed by this protocol. Which allows customers who are registered to the team to communicate with each other by message only visible by the team members.

To create a new channel the client need to use the /create [“team\_name”] [“team\_description”] command or become part of an existing channel by using /subscribe [“team\_uuid”] command, there is no user required to JOIN the team. If the team doesn't exist prior to joining, the server return Unknown team "Team\_name". As part of the protocol, a user may be a part of several channels at once.

### **5. Commands**

Server and clients send each other messages which may generate a reply. If the message contains a valid command, as described in later sections, the client should expect a reply as specified but it is not advised to wait forever for the reply; client to server and server to client communication is essentially asynchronous in nature.

Each My\_Teams message may consist of up to two main parts: the command, and the command parameters. The command, and all parameters are separated by one ASCII space character and ech parameter is wrapped into double quotes.

[My\_Teams server data processing]

/ \

/ \

/ \\_\_\_\_

/ \

[Client 1 send messages] [Client 1 receive server reply]

[Format of FTP My\_Teams messages Network]

### **6. Commands list**

• /help : show help

• /login [“username”] : set the username used by client

• /logout : disconnect the client from the server

• /users : get the list of all users that exist on the domain

• /user [“user\_uuid”] : get information about a user

• /send [“user\_uuid”] [“message\_body”] : send a message to a user

• /messages [“user\_uuid”] : list all messages exchange with an user

• /subscribe [“team\_uuid”] : subscribe to the event of a team and its sub directories (enable reception of all events from a team)

• /subscribed ?[“team\_uuid”] : list all subscribed teams or list all users subscribed to a team

• /unsubscribe [“team\_uuid”] : unsubscribe from a team

• /use ?[“team\_uuid”] ?[“channel\_uuid”] ?[“thread\_uuid”] : use specify a context team/channel/thread

• /create : based on what is being used create the sub resource (see below)

• /list : based on what is being used list all the sub resources (see below)

• /info : based on what is being used list the current (see below)

**7. Users interactions**

Clients can use the command /login [“username”] to set the username used by the client which will allow them to access the command not usable by client not logged. When connected the client can use the command /logout to disconnect from the server which will make the client close.

Two connected clients can interact with each other using the /send [“user\_uuid”] [“message\_body”] command. If the receiving user is logged, he will receive a notification indicating that he has received a message sent by another user. all connected users can view the messages they sent or received using the command /messages [“user\_uuid”]: list all messages exchange with a user