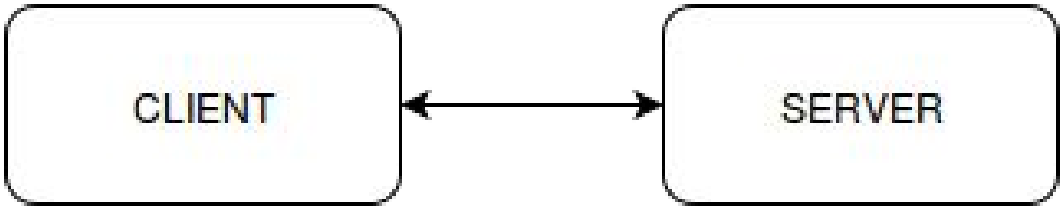
REMOTE FILE OPERATIONS

Damian Królikowski

Table of contents

1. General overview



Remote file operations enable an application that runs on a client computer to access files stored on server. There is two way communication between client and server. Client can send four commands to server :

- pwd
- ls
- cat
- cd

After successful connection, user input command in the client interface. Client verifies input data and send it to the server. Server, after recieve a valid command, verify exist and permission of files which client try to access. Depending on verify result server send back to the client appropriate message containing result or error. Client prints result in console. Application allows to send multiple messages. Communication between client and server ends when client close connection with an exit command.

2. Server

2.1 Compile and run

To compile server user has use ‘*make’* in the server's main directory named /*Server*. It will create server’s executable file which you have to run through *run.sh* script in the main directory. *run.sh* script sets library path, and run server with user defined parameters (-p <port> -d <starting directory>).

2.2 Server running

If any error during server start occurs, server print appropriate message and exit. After server starts, its waiting for client to connect. If client send connection request, server accept it, and waits for events from client. After client send packets to the server, its reading it and execute command from packets. After that server create own packets depending on command result and send them to the client.

2.3 Server close

There is no any command that user can put in server’s command line to close server. Thats because server is always waiting for client events. Server can be closed only by client disconnect, or by kill signal.

3. Client

3.1 Compile and run

To compile client user has use ‘*make’* in the client's main directory named /*Client*. It will create client’s executable file which you have to run through *run.sh* script in the main directory. *run.sh* script sets library path, and run client with user defined parameters (-p <server port> -i <server ip>).

3.2 Client running

If any error during server start occurs, client print appropriate message and exit. Client is automatically connected to server after it starts. After client connects, its waiting for user to put a command. When user puts command, client verify it and send it to the server. If command is invalid client prints appropriate message and waits for next command from user . If command was valid, client receives data from server and print it. After that client ask user to put another command.

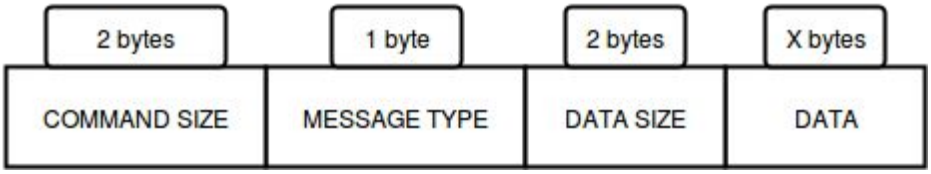
3.3 Client close

To close client user has to put “exit” command to the client.

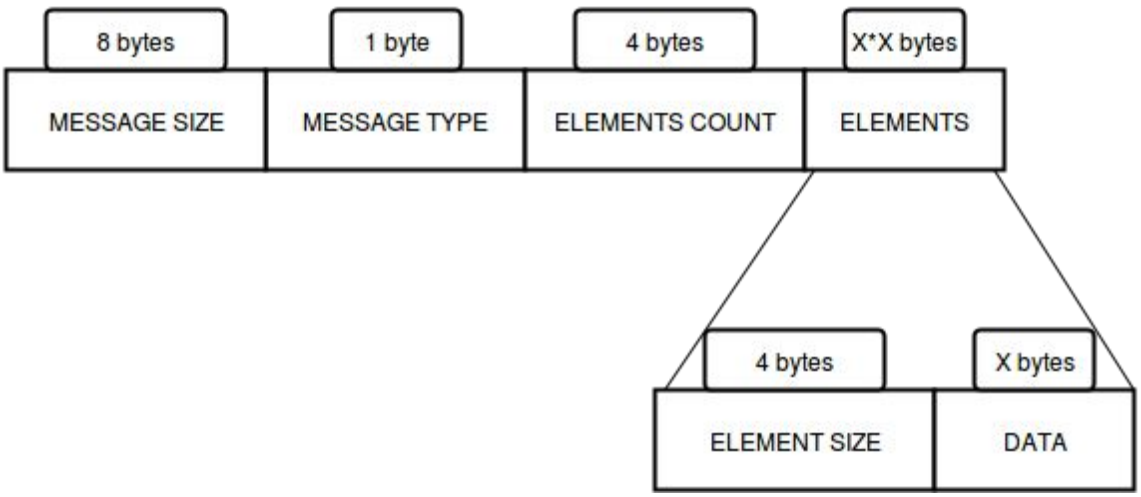
4. Protocol and data

4.1 Packet structure

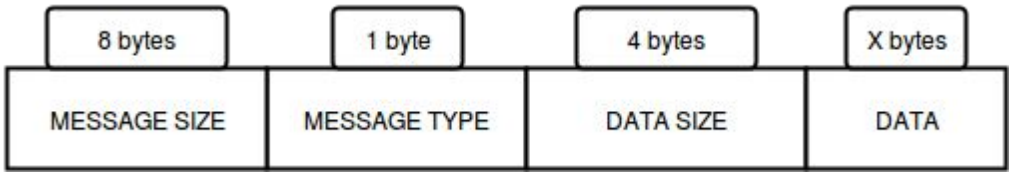
4.1.1 Client - Server command



4.1.2 Server - Client ls response



4.1.3 Server - Client cat, pwd, error response



4.2 Data

4.2.1

Max frame size is set to 1500 bytes.
There is no max file size.

5 . Diagrams

5.1 General UML diagram