Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Redes y Sistemas Distribuidos:

FTP protocol by sockets in C++

Second Year of Computer Engineering

Escuela Superior de Ingeniería y Tecnología

Universidad de La Laguna

Pablo Pérez González
(alu0101318318@ull.edu.es)

Índex:

1. Introduction.

The FTP (File Transfer Protocol) [1], is a network communication protocol. It is used to transfer computer files between two computers, the client and the server. The protocol is defined in RFC 959[2]. This document explains everything about the protocol, the error and successful codes, the commands, etc.

There are two ways to communicate between the server and the client: active and passive. The default mode is active, that means that the client starts listening for incoming data connections from the server on port X. It sends the FTP command PORT X to inform the server on which port it is listening. The server then initiates a data channel to the client from its port 21 (in the program we use 2121), the FTP server data port.

Using passive mode, the client uses the control connection to send a PASV command to the server and then receives a server IP address and server port number from the server, which the client then uses to open a data connection from a random client port to the server IP address and server port number received.

The project consists in creating a FTP server by sockets at C++ language. That's very useful because C++ is created on C, and it's the language used to create the POSIX sockets[3], which will be used in the project, specifically the Berkeley sockets[4] methods.

2. Compilation and execution

Inside the project folder, there is another called "src". Inside there is a makefile[5] that should be executed. There are differents options:

- "make" compiles the program.
- "make clear" deletes the executable files. This command can also be invoked by "make clean"
- "make run" deletes the executable files if it exists and compiles the program, after that, it executes the program. Is the fastests way to use the software.

When the software starts, it stays for a client (don't write on the server terminal), you must open another terminal and connect with the server. There are two commands to do this: "**`ftp`**" and "**`ftp -d`**". The last one enables the debug mode, and makes visible all the program traces. After that, you can connect to the

server via "**`open localhost 2121`**". Then, the software asks for a username. I recommend using the name that is inside the brackets. Later, it will ask for a pass, it's "**`1234`**". Now, the client is connected to the software.

There are two ways to use the server. The **active mode** and the **passive mode**. Default is on active mode, but it can be changed by writing "**`passive`**". On the other hand the available commands are:

- "**`ls`**" shows a list of all the files in the server's current directory.
- "**`get <FILE>`**" receives a copy of the file <FILE> to the current client directory from the server (server -> client).
- "**`put <FILE>`**" sends a copy of the file <FILE> to the current server directory from the client (client -> server).

3. Test cases.

3.1. Active mode.

Ls:

```
ftp>
ftp> open localhost 2121
Connected to localhost.
220 Service ready
ftp: setsockopt: Bad file descriptor
Name (localhost:pablo): pablo
---> USER pablo
331 User name ok, need password
Password:
---> PASS XXXX
230 User logged in
---> SYST
215 UNIX Type: L8.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
ftp: setsockopt (ignored): Permission denied
---> PORT 127,0,0,1,221,225
200 PORT OK.
---> LIST
125 List started OK.
ftp_server.cpp
ClientConnection.h
FTPServer.h
README
common.h
ftp_server
ClientConnection.cpp
Makefile
FTPServer.cpp
250 List completed successfully.
ftp>
```

Getting a file:

```
ftp>
ftp> open localhost 2121
Connected to localhost.
220 Service ready
ftp: setsockopt: Bad file descriptor
Name (localhost:pablo): pablo
---> USER pablo
331 User name ok, need password
Password:
---> PASS XXXX
230 User logged in
---> SYST
215 UNIX Type: L8.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> get README
local: README remote: README
---> TYPE I
200 OK
ftp: setsockopt (ignored): Permission denied
---> PORT 127,0,0,1,138,195
200 PORT OK.
---> RETR README
150 File status okay; about to open data connection
226 Closing data connection.
536 bytes received in 0.00 secs (3.6775 MB/s)
ftp>
```

Putting a file:

```
ftp>
ftp> open localhost 2121
Connected to localhost.
220 Service ready
ftp: setsockopt: Bad file descriptor
Name (localhost:pablo): pablo
---> USER pablo
331 User name ok, need password
Password:
---> PASS XXXX
230 User logged in
---> SYST
215 UNIX Type: L8.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> put README
local: README remote: README
---> TYPE I
200 OK
ftp: setsockopt (ignored): Permission denied
---> PORT 127,0,0,1,198,153
200 PORT OK.
---> STOR README
150 File creation okay; about to open data connection
226 Closing data conection
536 bytes sent in 0.00 secs (4.8683 MB/s)
ftp>
```

3.2. Passive mode.

Ls:

```
ftp>
ftp> open localhost 2121
Connected to localhost.
220 Service ready
ftp: setsockopt: Bad file descriptor
Name (localhost:pablo): pablo
---> USER pablo
331 User name ok, need password
Password:
---> PASS XXXX
230 User logged in
---> SYST
215 UNIX Type: L8.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> passive
Passive mode on.
ftp> ls
ftp: setsockopt (ignored): Permission denied
---> PASV
227 Entering Passive Mode (127,0,0,1,206,31)
---> LIST
125 List started OK.
ftp_server.cpp
ClientConnection.h
FTPServer.h
README
common.h
ftp_server
ClientConnection.cpp
Makefile
FTPServer.cpp
250 List completed successfully.
ftp>
```

Getting a file:

```
ftp>
ftp> open localhost 2121
Connected to localhost.
220 Service ready
ftp: setsockopt: Bad file descriptor
Name (localhost:pablo): pablo
---> USER pablo
331 User name ok, need password
Password:
---> PASS XXXX
230 User logged in
---> SYST
215 UNIX Type: L8.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> get README
local: README remote: README
---> TYPE I
200 OK
ftp: setsockopt (ignored): Permission denied
---> PASV
227 Entering Passive Mode (127,0,0,1,217,133)
---> RETR README
150 File status okay; about to open data connection
226 Closing data connection.
536 bytes received in 0.00 secs (2.2518 MB/s)
ftp> close
---> QUIT
221 Service closing control connection. Logged out if appropriate.
ftp>
```

Putting a file:

```
ftp>
ftp> open localhost 2121
Connected to localhost.
220 Service ready
ftp: setsockopt: Bad file descriptor
Name (localhost:pablo): pablo
---> USER pablo
331 User name ok, need password
Password:
---> PASS XXXX
230 User logged in
---> SYST
215 UNIX Type: L8.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> passive
Passive mode on.
ftp> put README
local: README remote: README
---> TYPE I
200 OK
ftp: setsockopt (ignored): Permission denied
---> PASV
227 Entering Passive Mode (127,0,0,1,238,63)
---> STOR README
150 File creation okay; about to open data connection
226 Closing data conection
536 bytes sent in 0.00 secs (5.4380 MB/s)
ftp>
```

4. References.

[1] FTP protocol (https://en.wikipedia.org/wiki/File_Transfer_Protocol).
[2] rfc959 (https://datatracker.ietf.org/doc/html/rfc959).
[3] Network socket (https://en.wikipedia.org/wiki/Network_socket).
[4] Berkeley sockets (https://en.wikipedia.org/wiki/Berkeley_sockets).
[5] make (https://en.wikipedia.org/wiki/Make_(software)).