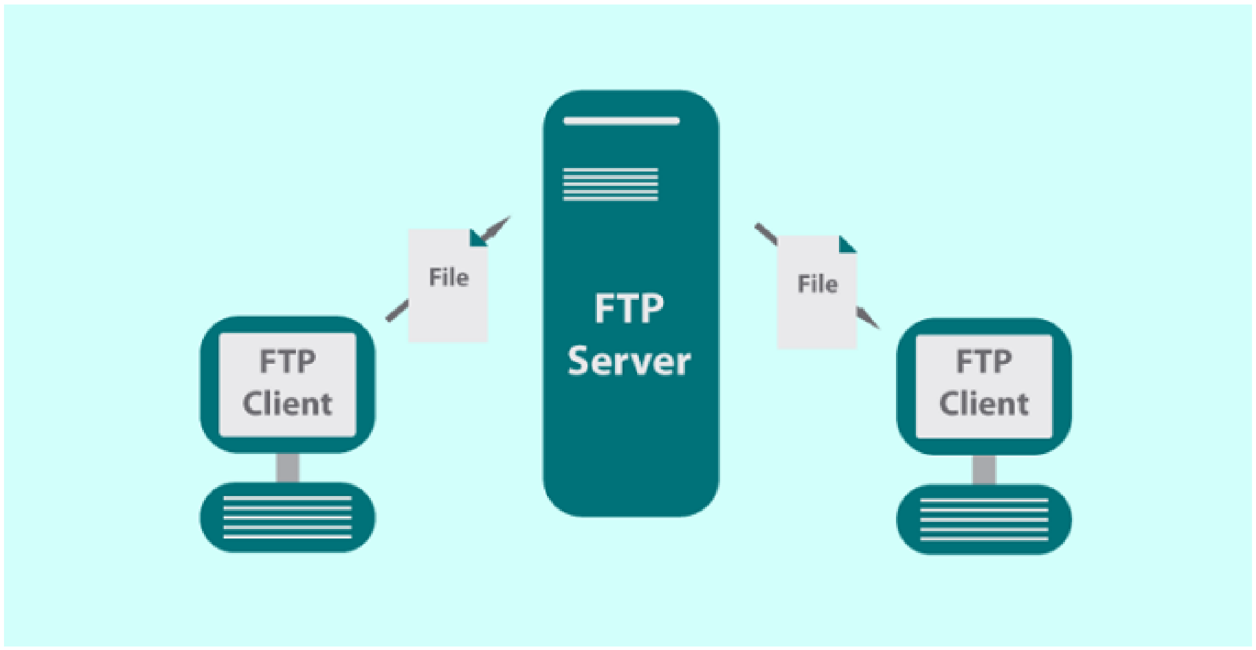
Distributed Networks and Systems Socket

Programming Sockets



Joseph Francisco Gabino Rodríguez

Index

1. A description of the developed application

In this practice we will work on the implementation of a simple FTP server. The basic functions will be the implementation of a server, so that it allows uploading and downloading files from a single folder. We will use different libraries related to sockets to establish communications and create processes.

We will work with two types of mode when executing the commands both in passive mode that we would have the problem of the firewall and in active mode where the client initiates both connections.

2. A description of the developed protocol

The File Transfer Protocol (FTP) is a standard communication protocol used for the transfer of computer files from a server to a client on a computer network. FTP is built on a client – server model architecture using separate control and data connections between the client and the server. FTP users may authenticate themselves with a clear-text sign-in protocol, normally in the form of a username and password, but can connect anonymously if the server is configured to allow it. For secure transmission that protects the username and password, and encrypts the content, FTP is often secured with SSL / TLS (FTPS) or replaced with SSH File Transfer Protocol (SFTP).

3. Compilation guide

The first would be to compile the program for them, we have a make file that compiles everything so that we do not have to indicate the files by command line, we will follow the following steps:

```
make
```

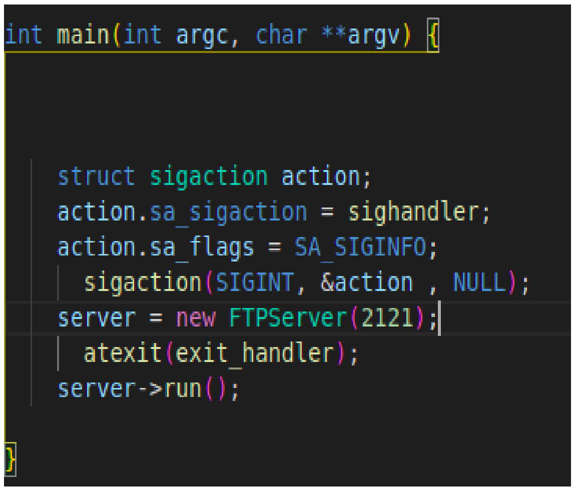
Then we execute a ls, and we will see the executable, to execute it we will put the following line:

```
./ftp_server
```

Once we have carried out these two steps we will have the server listening now, we would only have to open a new terminal and execute the following:

```
ftp -d
```

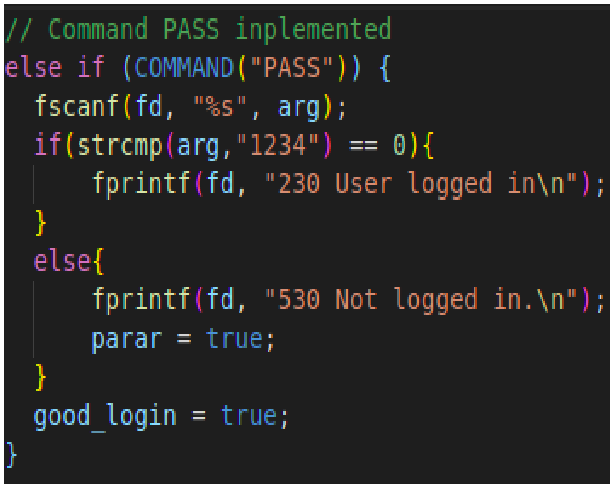
Once this is executed we would already be in FTP mode if we look at it code is waiting for port 2121



```
int main(int argc, char **argv) {


    struct sigaction action;
    action.sa_sigaction = sighandler;
    action.sa_flags = SA_SIGINFO;
      sigaction(SIGINT, &action , NULL);
    server = new FTPServer(2121);
      atexit(exit_handler);
    server->run();

}
```



```
// Command PASS inplemented
else if (COMMAND("PASS")) {
  fscanf(fd, "%s", arg);
  if(strcmp(arg,"1234") == 0){
      fprintf(fd, "230 User logged in\n");
  }
  else{
      fprintf(fd, "530 Not logged in.\n");
      parar = true;
  }
  good_login = true;
}
```

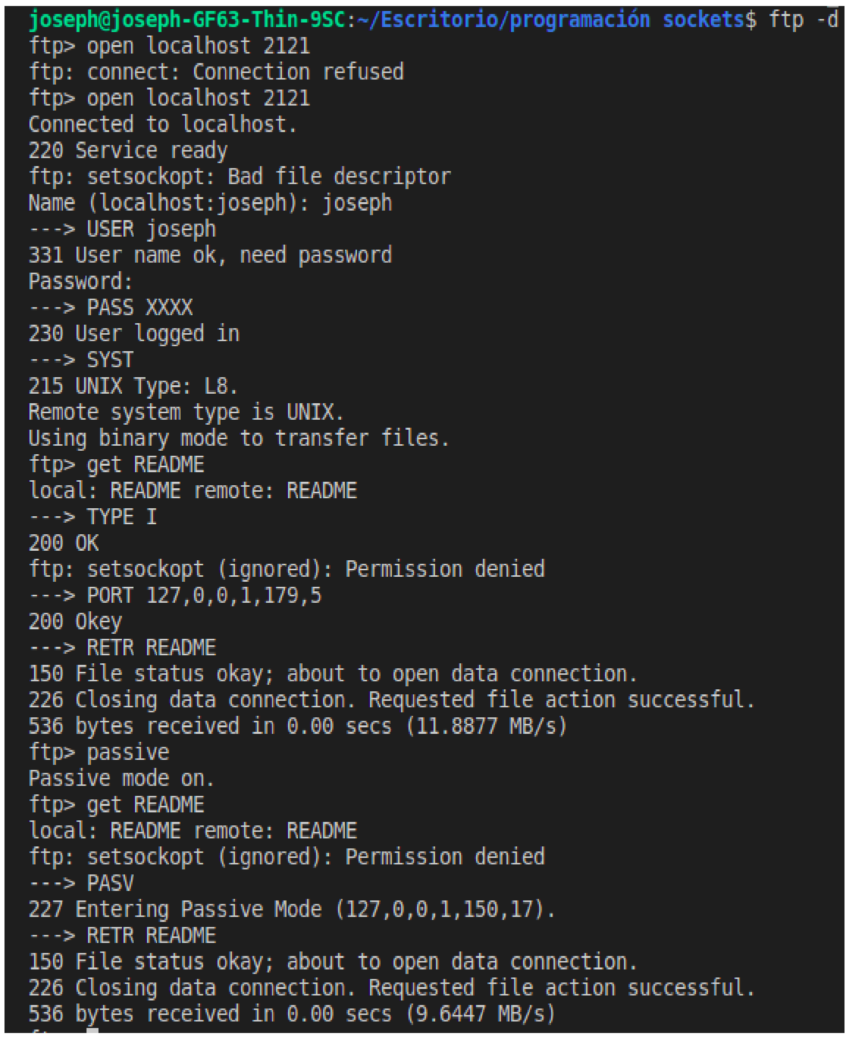
Therefore, we will use:

```
ftp> open localhost 2121
```

And we put our username and password 1234 which is also in the code. Once inside we can make a get file_name, put file_name, ls to see the content of the server in addition to the above both in passive and active.

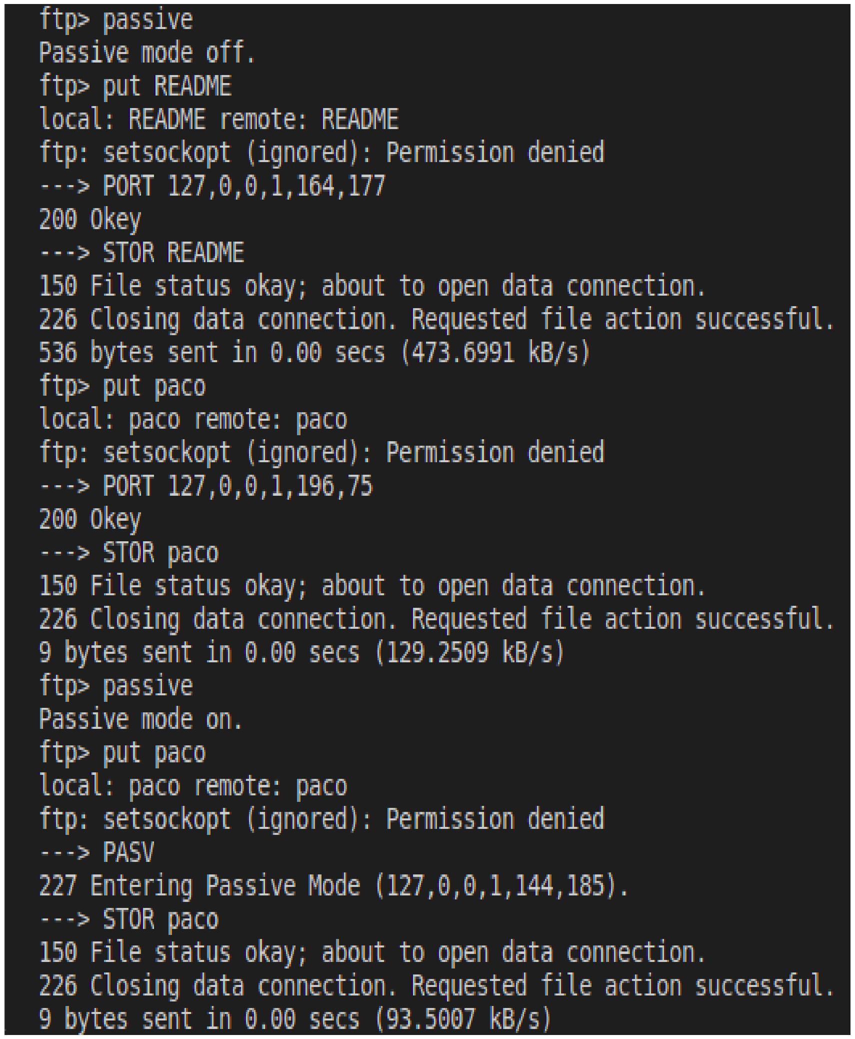
4. Test cases

Get file in active and passive mode



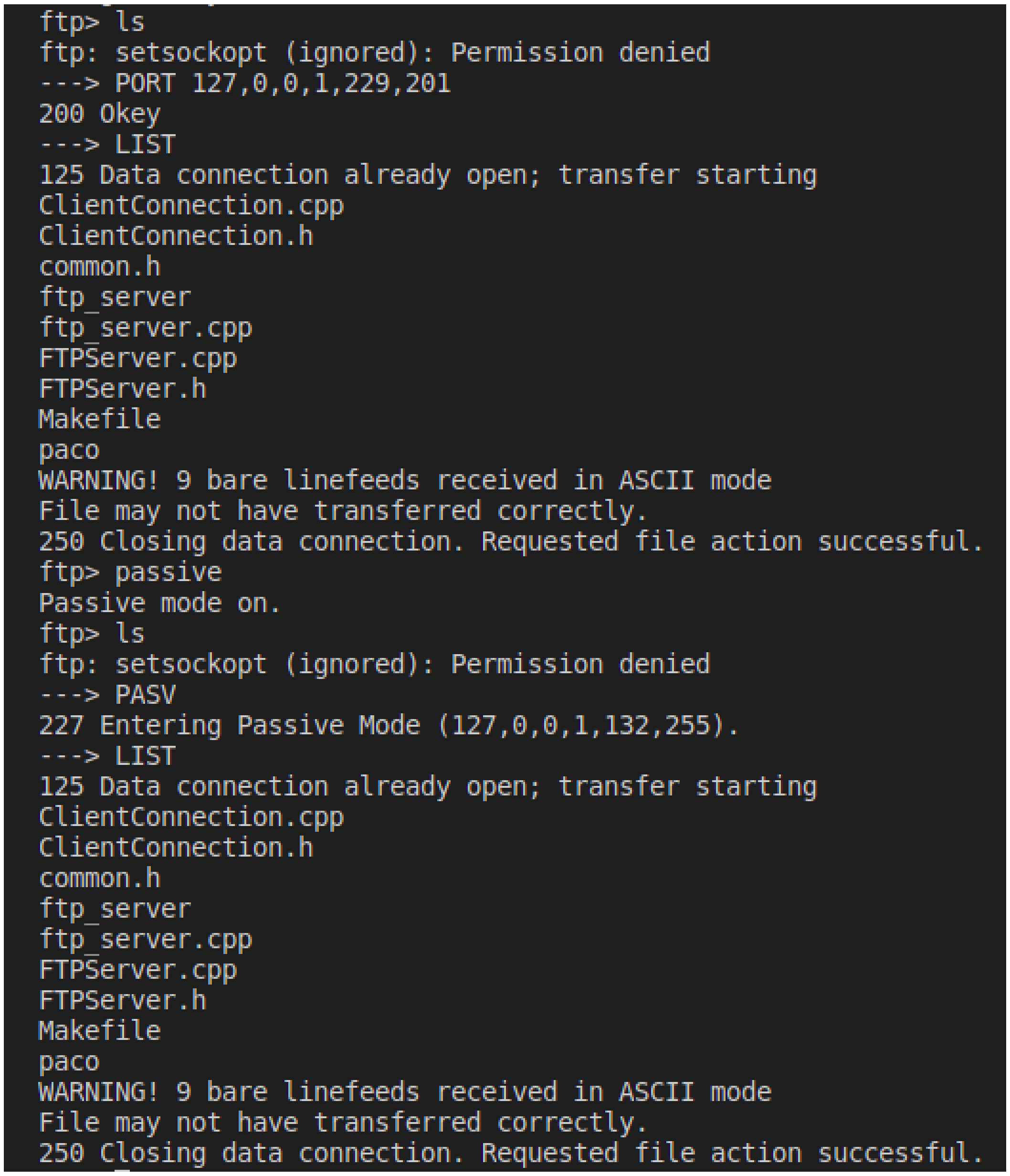
```
joseph@joseph-GF63-Thin-9SC:~/Escritorio/programación sockets$ ftp -d
ftp> open localhost 2121
ftp: connect: Connection refused
ftp> open localhost 2121
Connected to localhost.
220 Service ready
ftp: setsockopt: Bad file descriptor
Name (localhost:joseph): joseph
---> USER joseph
331 User name ok, 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 127,0,0,1,179,5
200 Okey
---> RETR README
150 File status okay; about to open data connection.
226 Closing data connection. Requested file action successful.
536 bytes received in 0.00 secs (11.8877 MB/s)
ftp> passive
Passive mode on.
ftp> get README
local: README remote: README
ftp: setsockopt (ignored): Permission denied
---> PASV
227 Entering Passive Mode (127,0,0,1,150,17).
---> RETR README
150 File status okay; about to open data connection.
226 Closing data connection. Requested file action successful.
536 bytes received in 0.00 secs (9.6447 MB/s)
```

Put file in active and passive mode



```
ftp> passive
Passive mode off.
ftp> put README
local: README remote: README
ftp: setsockopt (ignored): Permission denied
---> PORT 127,0,0,1,164,177
200 Okey
---> STOR README
150 File status okay; about to open data connection.
226 Closing data connection. Requested file action successful.
536 bytes sent in 0.00 secs (473.6991 kB/s)
ftp> put paco
local: paco remote: paco
ftp: setsockopt (ignored): Permission denied
---> PORT 127,0,0,1,196,75
200 Okey
---> STOR paco
150 File status okay; about to open data connection.
226 Closing data connection. Requested file action successful.
9 bytes sent in 0.00 secs (129.2509 kB/s)
ftp> passive
Passive mode on.
ftp> put paco
local: paco remote: paco
ftp: setsockopt (ignored): Permission denied
---> PASV
227 Entering Passive Mode (127,0,0,1,144,185).
---> STOR paco
150 File status okay; about to open data connection.
226 Closing data connection. Requested file action successful.
9 bytes sent in 0.00 secs (93.5007 kB/s)
```

ls in active and passive mode



```
ftp> ls
ftp: setsockopt (ignored): Permission denied
---> PORT 127,0,0,1,229,201
200 Okey
---> LIST
125 Data connection already open; transfer starting
ClientConnection.cpp
ClientConnection.h
common.h
ftp_server
ftp_server.cpp
FTPServer.cpp
FTPServer.h
Makefile
paco
WARNING! 9 bare linefeeds received in ASCII mode
File may not have transferred correctly.
250 Closing data connection. Requested file action successful.
ftp> passive
Passive mode on.
ftp> ls
ftp: setsockopt (ignored): Permission denied
---> PASV
227 Entering Passive Mode (127,0,0,1,132,255).
---> LIST
125 Data connection already open; transfer starting
ClientConnection.cpp
ClientConnection.h
common.h
ftp_server
ftp_server.cpp
FTPServer.cpp
FTPServer.h
Makefile
paco
WARNING! 9 bare linefeeds received in ASCII mode
File may not have transferred correctly.
250 Closing data connection. Requested file action successful.
```