Network Assignment 7

Name: Ritabroto Ganguly
Roll: 001910501090
BCSE-III, A3

Objective

Implement FTP protocol using TCP/UDP Socket as suitable.

Implementation

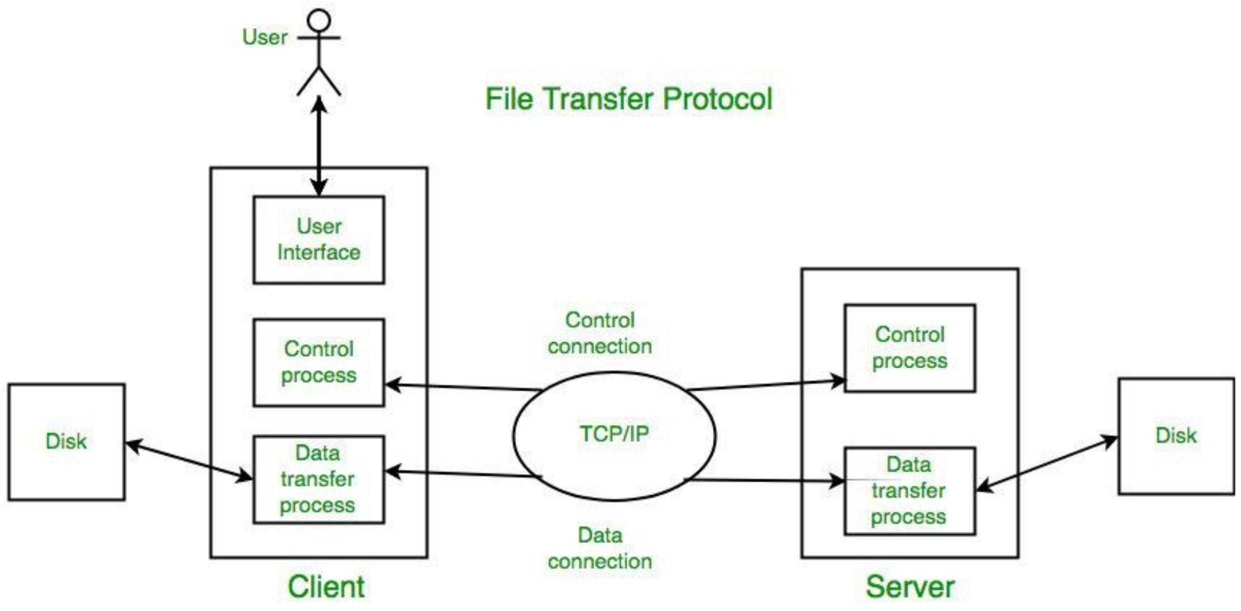


Fig : Basic model of FTP

FTP establishes two connections between the hosts. One connection is used for data transfer, the other for control information (commands and responses).

In my implementation, the client(host.py) and server(ftp_server.py) have established a single TCP socket connection, through which both commands(requests) and responses(data) are delivered bidirectionally. And I have only implemented FTP for ASCII files here.

• A file is to be copied from the server to the client. This is called retrieving aft/e. It is done under the supervision of the RETR command,

• A file is to be copied from the client to the server. This is called storing aft/e. It is done under the supervision of the STOR command.

• A list of directory or file names is to be sent from the server to the client. This is done under the supervision of the LIST command. Note that FTP treats a list of directory or file names as a file. It is sent over the data connection.

Code

```
#!/usr/bin/env python3.9


"""FTP Server implementation in python"""


import socket

import os


HOST = socket.gethostname()      # Standard loopback interface address (localhost)

FTP_PORT = 12345    # Port to listen on (non-privileged ports are > 1023)

PATH = "files/"


class FTPServer:

    """FTP Server class"""


    def __init__(self):

        """Initialize FTP Server"""


        self.file_name = ""

        self.name = ""
```

```
def start_ftp(self):
    """Start The FTP Server"""


    print("FTP Server started!!")
    while True:
        self.data = ""
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        self.sock.bind((HOST, FTP_PORT))
        print("Listening for a connection on its own port....")
        self.sock.listen(5)
        conn, addr = self.sock.accept()
        self.name = conn.recv(1024).decode("utf-8")
        print("Connected to {}".format(self.name))
        self.file_name = conn.recv(1024).decode("utf-8")
        self.file_name = PATH + self.file_name
        print("filename = {}".format(self.file_name))
        opt = conn.recv(1024).decode("utf-8")
        print("COMMAND =",opt)


        if(opt=="RETR"):
            print("\nRequest to retrieve the file {}\n".format(self.file_name))
            with open(self.file_name, "r", encoding="utf-8") as fptr:
                self.data = fptr.read()
            conn.sendall(bytes(self.data, "utf-8"))
            print("File sent successfully")


        elif(opt=="STOR"):
            print("\nRequest to store the file {}\n".format(self.file_name))
            while True:
                x = conn.recv(1024)
                if(len(x)<1):
                    break
                self.data += x.decode("utf-8")


            #print("data to be stored: {}".format(self.data))
            with open(self.file_name, "w", encoding="utf-8") as fptr:
                fptr.write(self.data)
            print("File stored successfully")


        if(opt=="LIST"):
            print("\nRequest to list the files stored in server\n")
            for x in os.listdir(PATH):
                if(x[0] != '.'):
                    self.data += (x+'  ')


            conn.sendall(bytes(self.data, "utf-8"))
            print("File sent successfully")


        conn.close()
        self.sock.close()
```

```
        print("FTP Server still running!")


if __name__ == "__main__":
    ftp_server = FTPServer()
    ftp_server.start_ftp()
#!/usr/bin/env python3.9
```

"""Host for connecting to the server"""

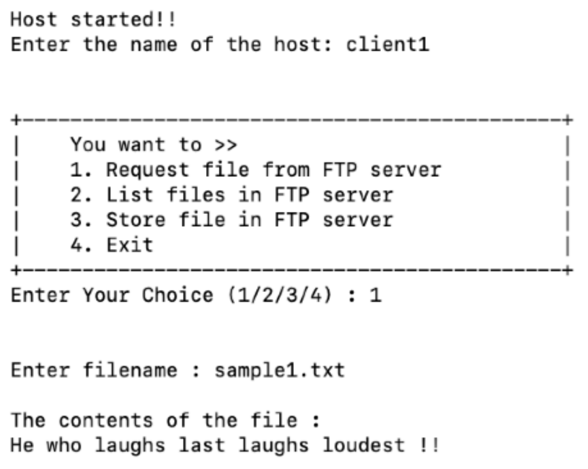
```
import socket
from time import sleep


HOST = socket.gethostname()     # Standard loopback interface address (localhost)
FTP_PORT = 12345




class Host:
    """Host Class for implementing connections"""


    def __init__(self):
        "Initialize The Hosts"


        self.file_name = ""
        self.name = ""


    def connect_hosts(self):
```

```
"""Connects The Hosts To Specific Servers"""

print("Host started!!")
self.name = input("Enter the name of the host: ")

while True:
    self.data = ""

    print("\n")
    print("+--------------------------------------------+")
    print("|    You want to >>                          |")
    print("|    1. Request file from FTP server         |")
    print("|    2. List files in FTP server             |")
    print("|    3. Store file in FTP server             |")
    print("|    4. Exit                                 |")
    print("+--------------------------------------------+")
    choice = input("Enter Your Choice (1/2/3/4) : ")
    print("\n")
    if choice == "4":
        print("Host has been terminated!")
        break
    if choice not in "1234":
        print("Invalid choice! Reselect 1/2/3/4")
        continue

    self.sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    self.sock.connect((HOST, FTP_PORT))
    self.sock.send(bytes(self.name, "utf-8"))
    if(choice in "13"):
        self.file_name = input("Enter filename : ")
    else:
        sleep(0.3)
        self.file_name = "*"
    self.sock.send(bytes(self.file_name, "utf-8"))
    sleep(0.5)

    if choice == "1":
        self.sock.send(bytes("RETR","utf-8"))
        while True:
            x = self.sock.recv(1024)
            if(len(x)<1):
                break
            self.data += x.decode("utf-8")

        print("\nThe contents of the file :")
        print("{}\n".format(self.data))
        self.sock.close()

    if choice == "2":
        self.sock.send(bytes("LIST","utf-8"))
        while True:
```

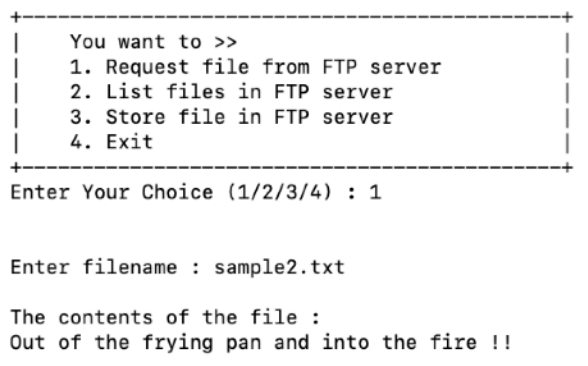
```
                x = self.sock.recv(1024)

                if(len(x)<1):

                    break

                self.data += x.decode("utf-8")


            print("\nThe file in the server are: ")

            print("{}\n".format(self.data))

            self.sock.close()


        elif choice == "3":

            self.sock.send(bytes("STOR","utf-8"))

            self.data = input("Data to be stored: ")

            self.sock.sendall(bytes(self.data,"utf-8"))

            self.sock.close()


if __name__ == "__main__":

    host = Host()

    host.connect_hosts()
```

Outputs

Client: **Server:**

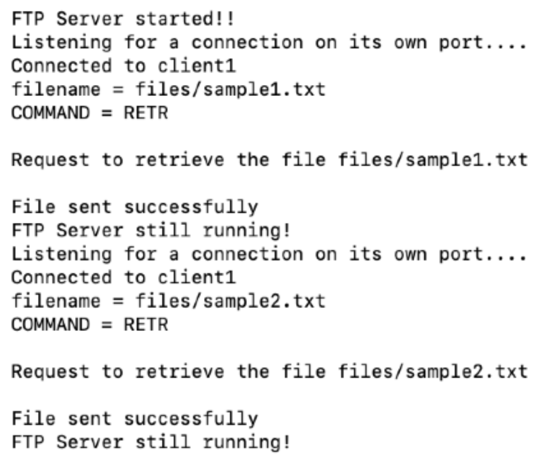


```
Host started!!
Enter the name of the host: client1


+--------------------------------------------+
|    You want to >>                          |
|    1. Request file from FTP server         |
|    2. List files in FTP server             |
|    3. Store file in FTP server             |
|    4. Exit                                 |
+--------------------------------------------+
Enter Your Choice (1/2/3/4) : 1


Enter filename : sample1.txt

The contents of the file :
He who laughs last laughs loudest !!
```

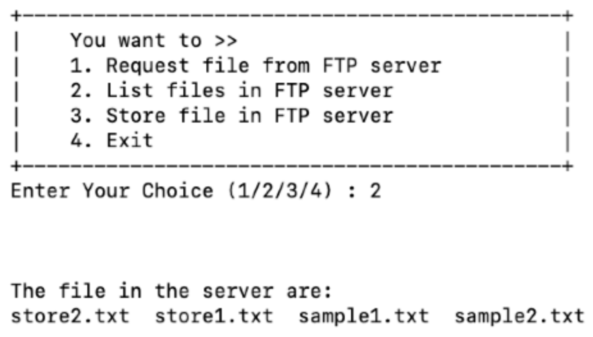


```
+--------------------------------------------+
|    You want to >>                          |
|    1. Request file from FTP server         |
|    2. List files in FTP server             |
|    3. Store file in FTP server             |
|    4. Exit                                 |
+--------------------------------------------+
Enter Your Choice (1/2/3/4) : 1


Enter filename : sample2.txt

The contents of the file :
Out of the frying pan and into the fire !!
```

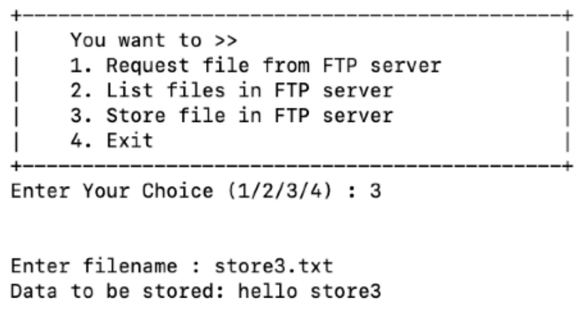
~/Desktop/Programs/7_Othe



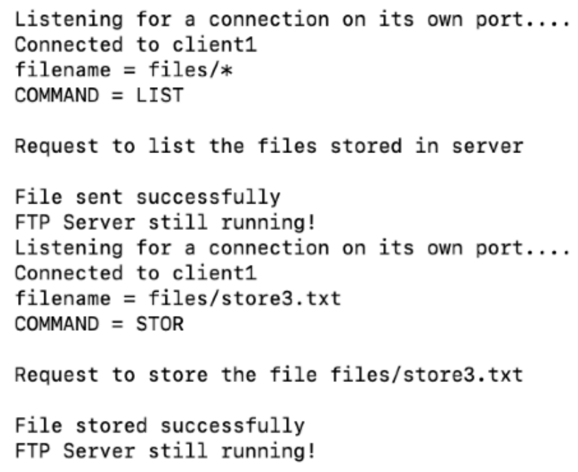
```
FTP Server started!!
Listening for a connection on its own port....
Connected to client1
filename = files/sample1.txt
COMMAND = RETR

Request to retrieve the file files/sample1.txt

File sent successfully
FTP Server still running!
Listening for a connection on its own port....
Connected to client1
filename = files/sample2.txt
COMMAND = RETR

Request to retrieve the file files/sample2.txt

File sent successfully
FTP Server still running!
```



```
+--------------------------------------------+
|    You want to >>                          |
|    1. Request file from FTP server         |
|    2. List files in FTP server             |
|    3. Store file in FTP server             |
|    4. Exit                                 |
+--------------------------------------------+
Enter Your Choice (1/2/3/4) : 2



The file in the server are:
store2.txt  store1.txt  sample1.txt  sample2.txt
```

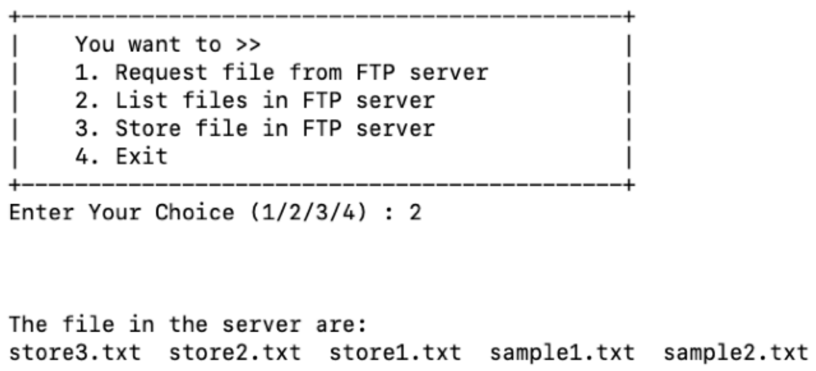


```
+--------------------------------------------+
|    You want to >>                          |
|    1. Request file from FTP server         |
|    2. List files in FTP server             |
|    3. Store file in FTP server             |
|    4. Exit                                 |
+--------------------------------------------+
Enter Your Choice (1/2/3/4) : 3


Enter filename : store3.txt
Data to be stored: hello store3
```

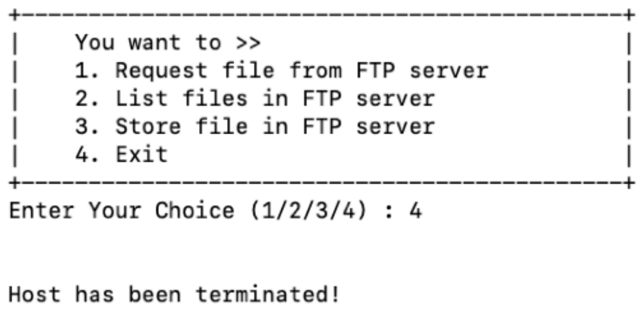


```
Listening for a connection on its own port....
Connected to client1
filename = files/*
COMMAND = LIST

Request to list the files stored in server

File sent successfully
FTP Server still running!
Listening for a connection on its own port....
Connected to client1
filename = files/store3.txt
COMMAND = STOR

Request to store the file files/store3.txt

File stored successfully
FTP Server still running!
```

Client:

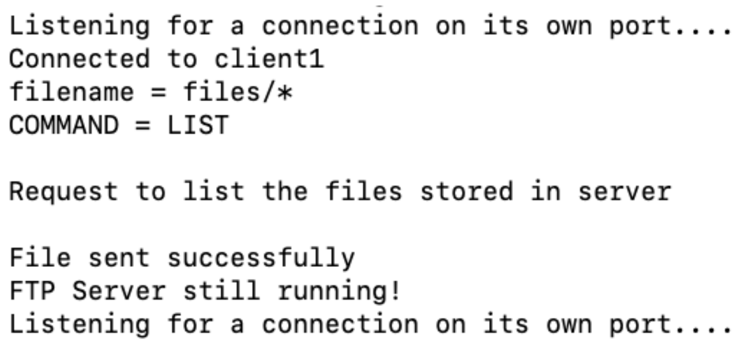


```
+--------------------------------------------+
|    You want to >>                          |
|    1. Request file from FTP server         |
|    2. List files in FTP server             |
|    3. Store file in FTP server             |
|    4. Exit                                 |
+--------------------------------------------+
Enter Your Choice (1/2/3/4) : 2


The file in the server are:
store3.txt  store2.txt  store1.txt  sample1.txt  sample2.txt
```



```
+--------------------------------------------+
|    You want to >>                          |
|    1. Request file from FTP server         |
|    2. List files in FTP server             |
|    3. Store file in FTP server             |
|    4. Exit                                 |
+--------------------------------------------+
Enter Your Choice (1/2/3/4) : 4


Host has been terminated!
```

Server:



```
Listening for a connection on its own port....
Connected to client1
filename = files/*
COMMAND = LIST

Request to list the files stored in server

File sent successfully
FTP Server still running!
Listening for a connection on its own port....
```

<u>Comments</u>

This assignment has helped me in understanding different networking protocols of application layer by researching and implementing them. It has also helped in understanding the demerits of these protocols, and how such demerits are overcome.