

File Server

Rwithik Manoj, Roll No. 53

Aim

To develop a concurrent file server which will provide the file requested by client if it exists. If not server sends appropriate message to the client. Server should also send its process ID (PID) to clients for display along with file or the message.

Theory

FTP (File Transfer Protocol) is a client-server protocol that may be used to transfer files between computers on the internet. The client asks for the files and the server provides them. It is a standard network protocol used for the transfer of computer files between a client and server on a computer network. FTP is built on a client-server model architecture and uses 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.

Server - Server should accept the incoming connection requests and establish a connection with the client. After this, it should check if the corresponding requested file is present or not. If found, the file is to be returned. Else the appropriate message is to be returned. The process ID is also to be sent to the client.

Client - After proper connection is established, the client should send the name of the file to be searched for. If found, the client should be able to receive the file as well as the PID of the server.

Code

Server

```
import socket
from _thread import start_new_thread
import threading
from os import getpid

print_lock = threading.Lock()
```

```

pid = getpid()

def threaded(c):
    while True:
        try:
            data = c.recv(1024)
        except Exception:
            break

        if not data:
            break

        fileName = data.decode()
        print(f"filename: {fileName}")
        c.send(str(pid).encode())

        try:
            file = open(fileName, "rb")
        except FileNotFoundError:
            c.send("File Not Found")
            print("File Not Found")
            break

        inp = file.read(1024)
        while inp:
            c.send(inp)
            inp = file.read(1024)
        c.send("".encode())

        c.close()

    print(f"{c} disconnected")
    print_lock.release()

def main():
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.bind(("127.0.0.1", 9999))
    s.listen(5)
    print("Listening for connections")

    while True:
        c, addr = s.accept()

```

```

        print_lock.acquire()
        print("Connected to :", addr)

        start_new_thread(threaded, (c,))
    s.close()

if __name__ == "__main__":
    main()

Client

import socket

def main():
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(("127.0.0.1", 9999))

    fileName = input("File name: ")
    # fileName = "test-file.txt"

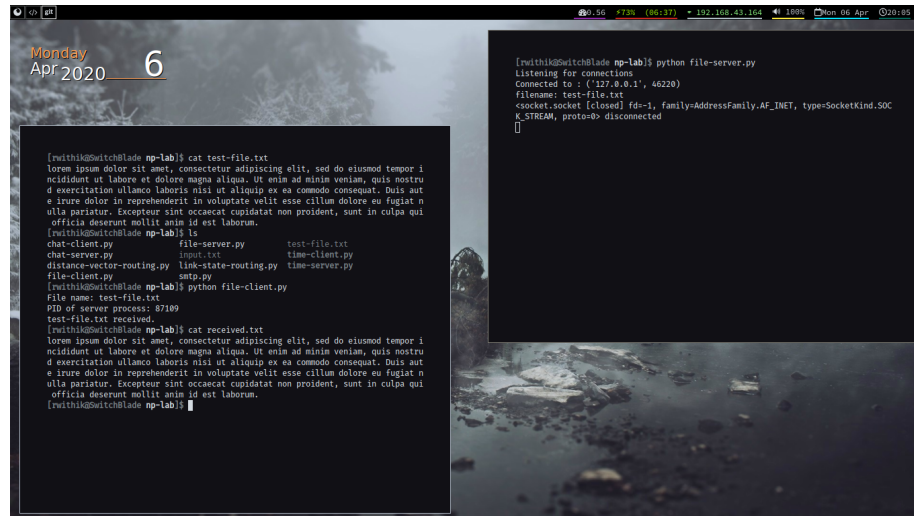
    outputFile = open("received.txt", "wb")
    s.send(fileName.encode())
    pid = s.recv(1024)
    print(f"PID of server process: {pid.decode()}")
    data = s.recv(1024)
    while data:
        if not data.decode():
            s.shutdown(socket.SHUT_WR)
            break
        # print(data)
        outputFile.write(data)
        data = s.recv(1024)

    s.close()
    print(f"{fileName} received.")

if __name__ == "__main__":
    main()

```

Output



The screenshot shows a terminal window with a dark background and a light-colored font. The window title is "Monday APR 2020 6". The terminal output shows a user at a prompt [rwithik@SwitchBlade np-lab] running a series of commands. First, they run 'cat test-file.txt' which displays a block of Lorem Ipsum text. Then, they run 'ls' which lists several files: chat-client.py, chat-server.py, distance-vector-routing.py, file-client.py, file-server.py, input.txt, link-state-routing.py, smtp.py, test-file.txt, time-client.py, and time-server.py. Next, they run 'python file-client.py', which prompts for a file name (test-file.txt) and a server PID (87109). The output then shows 'test-file.txt received.' and 'cat received.txt', which again displays the same Lorem Ipsum text. The terminal window is overlaid on a background image of a rocky, mountainous landscape.

```
[rwithik@SwitchBlade np-lab]$ cat test-file.txt
lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor i
ncidunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostru
d exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aut
e irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat n
ulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui
officia deserunt mollit anim id est laborum.
[rwithik@SwitchBlade np-lab]$ ls
chat-client.py      file-server.py      test-file.txt
chat-server.py      input.txt           time-client.py
distance-vector-routing.py  link-state-routing.py  time-server.py
file-client.py      smtp.py
[rwithik@SwitchBlade np-lab]$ python file-client.py
File name: test-file.txt
PID of server process: 87109
test-file.txt received.
[rwithik@SwitchBlade np-lab]$ cat received.txt
lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor i
ncidunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostru
d exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aut
e irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat n
ulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui
officia deserunt mollit anim id est laborum.
[rwithik@SwitchBlade np-lab]$
```



This screenshot shows a terminal window with a dark background and a light-colored font. The window title is "Monday APR 2020 6". The terminal output shows a user at a prompt [rwithik@SwitchBlade np-lab] running a series of commands. First, they run 'cat test-file.txt' which displays a block of Lorem Ipsum text. Then, they run 'ls' which lists several files: chat-client.py, chat-server.py, distance-vector-routing.py, file-client.py, file-server.py, input.txt, link-state-routing.py, smtp.py, test-file.txt, time-client.py, and time-server.py. Next, they run 'python file-client.py', which prompts for a file name (test-file.txt) and a server PID (87109). The output then shows 'test-file.txt received.' and 'cat received.txt', which again displays the same Lorem Ipsum text. The terminal window is overlaid on a background image of a rocky, mountainous landscape.

```
[rwithik@SwitchBlade np-lab]$ cat test-file.txt
lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor i
ncidunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostru
d exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aut
e irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat n
ulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui
officia deserunt mollit anim id est laborum.
[rwithik@SwitchBlade np-lab]$ ls
chat-client.py      file-server.py      test-file.txt
chat-server.py      input.txt           time-client.py
distance-vector-routing.py  link-state-routing.py  time-server.py
file-client.py      smtp.py
[rwithik@SwitchBlade np-lab]$ python file-client.py
File name: test-file.txt
PID of server process: 87109
test-file.txt received.
[rwithik@SwitchBlade np-lab]$ cat received.txt
lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor i
ncidunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostru
d exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aut
e irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat n
ulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui
officia deserunt mollit anim id est laborum.
[rwithik@SwitchBlade np-lab]$
```

```
[rwithik@SwitchBlade np-lab]$ python file-server.py
Listening for connections
Connected to : ('127.0.0.1', 46220)
filename: test-file.txt
<socket.socket [closed] fd=-1, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0> disconnected
█
```