2020

9th January

LAB 11

OPERATING SYSTEM

BESE-(9A)

Ali Tariq (241916)

Muhammad Abdullah (241548)

Syed Afraz Hasan Kazmi (241847)

File System Simulation

File System (THREADED)

The objective of this lab is to simulate a filesystem based on an existing directory structure given as an input file and allow multiple clients to manipulate the system residing on a server without inconsistencies.

The program accepts the following parameters:

- -f [input file for directory structure]
- -s [Disk Size]
- -b [Block/Sector Size]

Run **server** using the format,

python3 filename.py -f -s -b

Example: python3 filesystemwolf.py list 65536 512

Note: A prerequisite to this program is installing the `llist` module which is library of Doubly and Singly Linked Lists. To do so, open Linux terminal, cd into the program directory and write the following command 'pip3 install llist'. (Needs to be done only once)

The program will use the input parameters and create a file system structure, against which you will be able to run commands.

Run **client** using the format,

python3 filename.py

Example: python3 fileSysClient.py

Input file (the -f parameter)

The input files are already provided in this repo but a user can make their own input files using the find Unix command. There are two files namely, ‘dir_list.txt’ and ‘file_list.txt’. The directory list file has **absolute** paths of all directories. The file list file has **size** and **absolute** paths of all files

`find ./ -type d > dir_list.txt` - Generates a list of all directories and subdirectories and outputs it into a txt file.

`find ./ -type f -ls > file_list.txt` - Generates a list of all files with their size and outputs into a txt file. (Note: This file may have to be modified to conform to the required format)

Introducing Client-Server Architecture

The threading achieved in previous lab (lab-09) is still kept intact because we are still required to entertain multiple clients(hence multiple threads).

The design catered keeps the server 'heavy' and client 'light', meaning the filesystem is maintained on a single master server. Multiple clients can concurrently connect to the server and send requests in the same manner as it was done earlier.

The server maintains a list of connected users. Every element of this list holds metadata about the connected client such as username, thread number, IP Address of client, its port number and lastly a count for the times client has accessed the system(to be used in lab 12)

The protocol used for the transport layer is **TCP**. The reason for using TCP is, the communication between server and client requires exchanging of text passed messages and not of actual files because, again, this is just a simulation of the file system. Suppose, the client sends a message such as 'appendbytes foo.txt 100'. Upon receiving this message, the server then writes 100 bytes to foo.txt.

TCP is also preferred as it ensures data integrity. The key characteristics which are crucial to us are in-order delivery of uncorrupted messages, without loss or duplications.

The interface remains the same as before, both client and server run on the terminal/command-line interface.

The completed tasks are as follows,

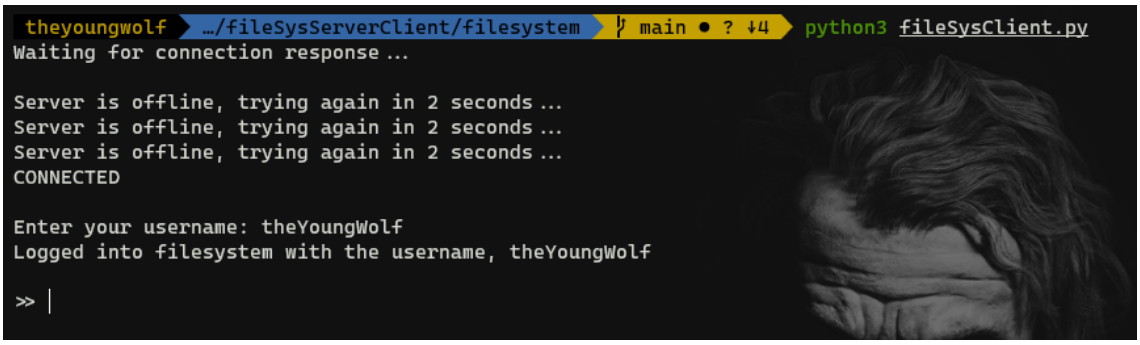
Tasks	
✓	Two programs, server and client
✓	Static IP of server
✓	Client identified by username provided
✓	Interaction: Command line interface
✓	Client waits for set amount of time if server is offline
✓	Response from server is displayed on client side
✓	Server capable of entertaining multiple clients
✓	Server binds to port 21018 (not 95)
✓	Server and client can run on separate machines within local network

- Make directory function
- Remove directory function
- Make file function
- Remove file function
- Append to file function
- Truncate from file function
- Memory map function
- Print disk status
- Print tree hierarchy

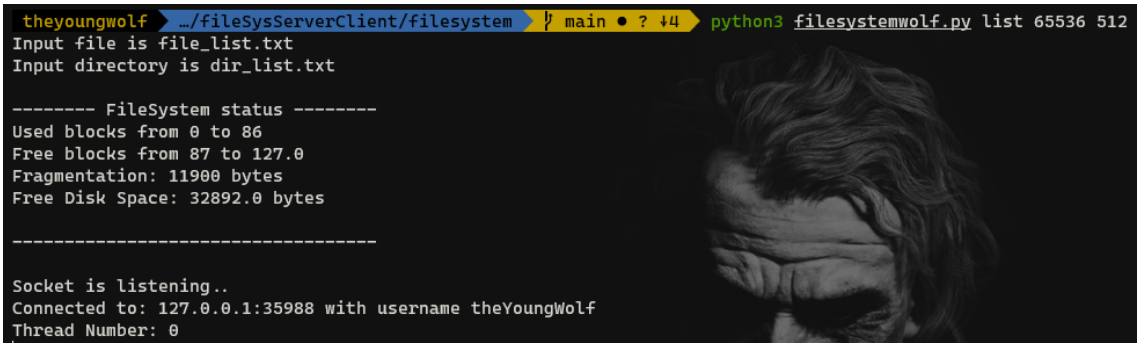
Note: Due to our negligence, screenshots of previous labs were missed in their reports, hence the screenshots of this lab are attached herewith implying or as collective proof of a working system.

SCREENSHOTS

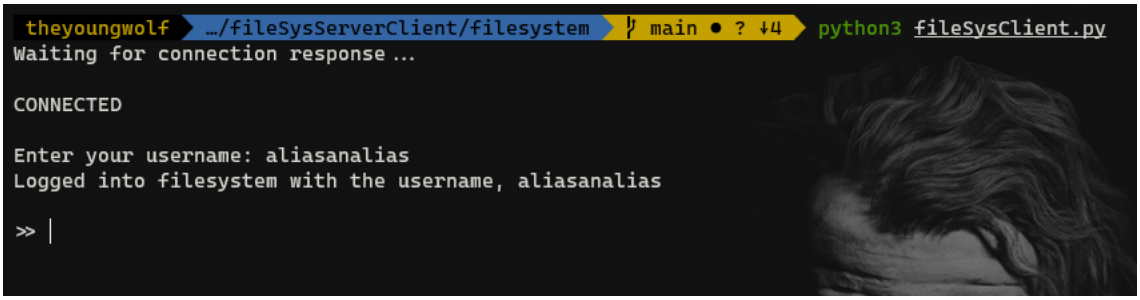
Client 1



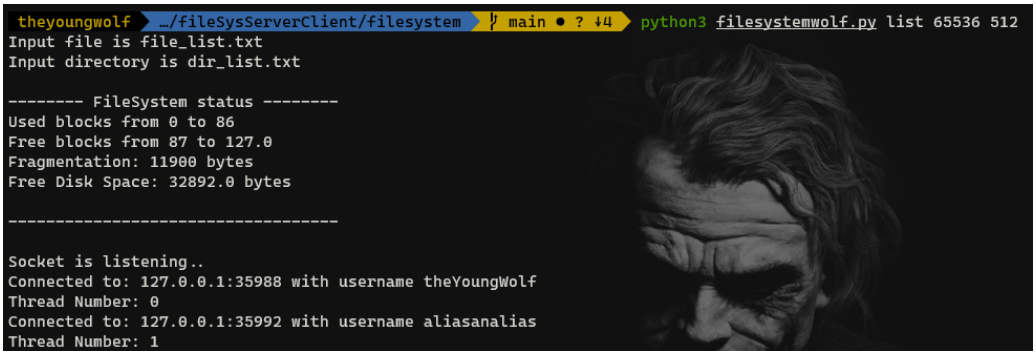
Server



Client 2



Server



Client 1 requests

```
Enter your username: theYoungWolf
Logged into filesystem with the username, theYoungWolf

>> cd 1A


./1A/>> ls

2A
2B
2c
f2b.o
f2d.txt
f2c.c
f2a.c
f13.exe

./1A/>> cd 2A


./1A/2A/>> ls

3C
3B
3A
f3d.txt
f3a.c
f3c.c
f3b.o
```

```
./1A/2A/>> printfiles

------------------------------------------------------------
1. File name: f3d.txt
2. Full name: ./1A/2A/f3d.txt
3. File path: ./1A/2A/
4. File size: 334 bytes

File stored in 1 piece(s):

        1. Memory location: 2560- Used memory: 334/512 bytes
------------------------------------------------------------
------------------------------------------------------------
1. File name: f3a.c
2. Full name: ./1A/2A/f3a.c
3. File path: ./1A/2A/
4. File size: 82 bytes

File stored in 1 piece(s):

        1. Memory location: 3072- Used memory: 82/512 bytes
------------------------------------------------------------
------------------------------------------------------------
1. File name: f3c.c
2. Full name: ./1A/2A/f3c.c
3. File path: ./1A/2A/
4. File size: 1016 bytes

File stored in 2 piece(s):

        1. Memory location: 3584- Used memory: 512/512 bytes
        2. Memory location: 4096- Used memory: 504/512 bytes
------------------------------------------------------------
------------------------------------------------------------
```

```
------------------------------------------------------------
1. File name: f3b.o
2. Full name: ./1A/2A/f3b.o
3. File path: ./1A/2A/
4. File size: 1604 bytes

File stored in 4 piece(s):

        1. Memory location: 8704- Used memory: 512/512 bytes
        2. Memory location: 9216- Used memory: 512/512 bytes
        3. Memory location: 9728- Used memory: 512/512 bytes
        4. Memory location: 10240- Used memory: 68/512 bytes
------------------------------------------------------------

./1A/2A/>>
```

Client 2 requests

```
Enter your username: aliasanalias
Logged into filesystem with the username, aliasanalias

>> cd 2A

2A: No such file or directory

./>> cd 1X


./1X/>> ls

2X

./1X/>> create foo.txt 516


./1X/>> ls

2X
foo.txt

./1X/>> printfiles

------------------------------------------------------------
1. File name: foo.txt
2. Full name: ./1X/foo.txt
3. File path: ./
4. File size: 516 bytes

File stored in 2 piece(s):

        1. Memory location: 44544- Used memory: 512/512 bytes
        2. Memory location: 45056- Used memory: 4/512 bytes
------------------------------------------------------------
```

```
./1X/>> printdisk

Used blocks from 0 to 88
Free blocks from 89 to 127.0
Fragmentation: 12408 bytes
Free Disk Space: 32376.0 bytes

./1X/>> rmdir 2X


./1X/>> ls

foo.txt

./1X/>> appendbytes foo.txt 100


./1X/>> printfiles

------------------------------------------------------------
1. File name: foo.txt
2. Full name: ./1X/foo.txt
3. File path: ./
4. File size: 616 bytes

File stored in 2 piece(s):

       1. Memory location: 45568- Used memory: 512/512 bytes
       2. Memory location: 46080- Used memory: 104/512 bytes
------------------------------------------------------------

./1X/>>
```

Client 2 exits

```
./1X/>> exit
theyoungwolf …/fileSysServerClient/filesystem  main ● ? ↓4
```

Server

```
User aliasanalias with Thread number 1 exiting.
```

Client 1 exits

```
./1A/2A/>> exit
theyoungwolf …/fileSysServerClient/filesystem  main ● ? ↓4
```

Server

```
User theYoungWolf with Thread number 0 exiting.
Shutting down server as no more clients to entertain
theyoungwolf …/fileSysServerClient/filesystem  main ● ? ↓4
```