Homework 4 Report – CSE344

Mehmet Mahir Kayadelen
Student no. 210104004252

May 2023

1 Composition

For this project, I used a seperate c file for each side server.c and client.c and one makefile for compile them for easy testability.

2 Program Logic

The server-client program consists of two main components: the server and the client. The server is responsible for accepting client connections, managing client requests, and providing appropriate responses. The client connects to the server and sends commands, receiving responses from the server.

1)Server side

The server side program is designed to handle client connections and process their commands. It utilizes multiple threads for concurrent execution. The main thread initializes variables, semaphores, and sets up signal handling.I opened 3 threads and pool threads in main . The Connection Reader Thread waits for clients to connect, assigns them unique IDs, and establishes communication channels through named pipes. It manages client connections. The Command Reader Thread continuously reads commands from a named pipe and adds them to the command queue. Worker Threads(Pool threads) retrieve commands from the queue, parse them, and handle the corresponding client requests. The server supports commands such as list, readF, writeT, upload, download, quit, killServer. The server maintains a log file for all commands and events. The program incorporates semaphores to ensure synchronization and prevent race conditions when accessing shared resources. Overall, server program provides a concurrent and synchronized environment for handling client connections, processing commands, and logging events.

Thread details:

Connection Reader Thread: This thread is responsible for handling client connections. It waits for clients to connect and stores their information in an array.

Worker Threads(Pool Threads): Multiple worker threads are created to process commands from the command queue concurrently. Each worker thread retrieves a command from the queue, parses it, and handles the client's request accordingly.

Command Reader Thread: This thread reads commands from the named pipe When a command is received, it adds it to the command queue for processing.

Problems and Solutions)

Thread Communication:

-Server has a command queue using the circular buffer approach to facilitate communication between the connection reader thread and the worker threads. The command queue is protected using the queueAccessSemaphore semaphore to prevent simultaneous access.

Race conditions:

-Server uses flock when reading and writing files(mutual exclusion)

-Server creates a semaphore to limit the number of simultaneous client connections to a specified maximum

Signal handling:

I implemented a signal handler function named handle_signal that captures the SIGUSR1 and SIGINT signals. When these signals are received, the handler terminates the program gracefully by notifying and killing clients, cleaning up resources, removing FIFOs, closing file descriptors, and releasing semaphores.

2)Client side

My client side code is same as midterm client code.The only change I've made is that it sends the requests to the main command receiving pipe, not the client specific pipe. I did not change the response waiting mechanism. It still takes from the customer specific pipe.
The client code establishes a connection with the server by opening a named pipe for writing and optionally for reading. It sends a connection request to the server and waits for the server's response. The client then enters a command handling loop where it prompts the user for commands, processes them accordingly, and communicates with the server by writing to the named pipe. The client supports commands such as listing files, reading file content, writing to files, uploading and downloading files, quitting, and killing the server. It waits for the server's responses for certain commands and displays them to the user. The client also handles termination requests gracefully by implementing signal handling. It captures signals such as SIGUSR2, which indicates the server is shutting down, and SIGINT, which allows the client to exit gracefully by notifying the server. Additionally, the client prevents the use of SIGTSTP, as it is not allowed for client-side operations. When receiving these signals, the client takes appropriate actions such as displaying messages, sending requests to the server, and closing the named pipes before exiting.

3 Testing

list,readF,writeT,upload,download,quit,killServer

Test 1) A single client connects to the server, performs basic operations(operation for a large file (>100 MB)), and disconnects successfully.

```
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4/serverdir$ ls -l
total 129792
-rw-r--r-- 1 mahir mahir 132896656 May 16 21:36 TelegramToDownload.exe
-rw-r--r-- 1 mahir mahir        26 May 21 22:18 toRead.txt
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4/serverdir$
```

1

```
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$ ls -l
total 129888
-rw-r--r-- 1 mahir mahir 132896656 Apr 25 11:58 TelegramToUpload.exe
-rwxr-xr-x 1 mahir mahir     22136 May 21 22:22 client
-rw-r--r-- 1 mahir mahir     12935 May 21 22:15 client.c
-rwxr-xr-x 1 mahir mahir     32192 May 21 22:22 server
-rw-r--r-- 1 mahir mahir     24743 May 21 19:26 server.c
drwxr-xr-x 2 mahir mahir      4096 May 21 22:18 serverdir
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$
```

2

(1&2) Before performing operations Working Directory and Server's Directory

```
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$ ./server serverdir 1 1
>> Server Started PID 15026
>> Waiting for clients...
>>Client PID:'15031' connected as 'client1'
client1 disconnected
>>Client PID:'15038' connected as 'client2'
>>Kill signal from client15038.. Terminating...
>>bye
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$
```

3

```
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$ ./client Connect 15026
Waiting for server queue...Connection established.
>>Enter command: list
Files in server directory:
TelegramToDownload.exe
all_logs.txt
toRead.txt
>>Enter command: readF toRead.txt
File content:
Hello from toRead.txt :)

>>Enter command: writeT toRead.txt helloServer:)
Write successful.
>>Enter command: readF toRead.txt
File content:
Hello from toRead.txt :)
helloServer:)

>>Enter command: download TelegramToDownload.exe
Server response:
File downloaded successfully. Transferred 132896656 bytes.
>>Enter command: upload TelegramToUpload.exe
Server response:
File uploaded successfully. Transferred 132896656 bytes.
>>Enter command: list
Files in server directory:
TelegramToDownload.exe
all_logs.txt
toRead.txt
TelegramToUpload.exe
>>Enter command: quit
quit request sent to server.
Log message from server: client is shutting down.
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$ ./client Connect 15026
Waiting for server queue...Connection established.
>>Enter command: killServer
Kill request sent to server.
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$
```

4

(3&4) Connect and Performing operations Working Directory and Server's Directory

```
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4/serverdir$ ls -l
total 259580
-rw-r--r-- 1 mahir mahir 132896656 May 16 21:36 TelegramToDownload.exe
-rw-r--r-- 1 mahir mahir 132896656 May 21 22:25 TelegramToUpload.exe
-rw------- 1 mahir mahir       486 May 21 22:26 all_logs.txt
-rw-r--r-- 1 mahir mahir        40 May 21 22:24 toRead.txt
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4/serverdir$
```

5

```
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$ ls -l
total 259676
-rw-r--r-- 1 mahir mahir 132896656 May 21 22:25 TelegramToDownload.exe
-rw-r--r-- 1 mahir mahir 132896656 Apr 25 11:58 TelegramToUpload.exe
-rwxr-xr-x 1 mahir mahir     22136 May 21 22:22 client
-rw-r--r-- 1 mahir mahir     12935 May 21 22:15 client.c
-rwxr-xr-x 1 mahir mahir     32192 May 21 22:22 server
-rw-r--r-- 1 mahir mahir     24743 May 21 19:26 server.c
drwxr-xr-x 2 mahir mahir      4096 May 21 22:25 serverdir
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$
```

6

(5&6) After performing operations Working Directory and Server's Directory

We can see that the upload and download operations are successful.

Test2) Multiple clients try to connect to the server, but the server's client capacity is full. When space is free, it connects and sends killServer command, Server shuts down all clients. In order to understand the input/output behavior in different SS, I have colored it, each color represents a certain time.

```
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$ ./server dirname 2 1
>> Server Started PID 21866
>> Waiting for clients...
>>Client PID:'21873' connected as 'client1'
>>Client PID:'21877' connected as 'client2'
>>Connection request PID 21878… Que FULL. Client leaving...
>>Connection request PID 21879… Que FULL.
client2 disconnected
>>Client PID:'21879' connected as 'client3'
>>Kill signal from client21879.. Terminating...
>>bye
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$
```

1

First Connection:

```
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$ ./client Connect 21866
Waiting for server queue...Connection established.
>>Enter command: quit
quit request sent to server.
Log message from server: client is shutting down.
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$
```

2

Second Connection:

```
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$ ./client tryConnect 21866
Connection established.
>>Enter command: Server is shutting down. Terminating client.
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$
```

3

Third Connection attempt with tryConnect(no slot) after with Connect option(waits queue till first connection quits.) After sends killServer command to server:

```
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$ ./client tryConnect 21866
Server is Full!
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$ ./client Connect 21866
Waiting for server queue...Connection established.
>>Enter command: killServer
Kill request sent to server.
>>Enter command: Server is shutting down. Terminating client.
mahir@DESKTOP-9RRECEV:~/CSE344-SYSTEM/hw4$
```

4

Server log file (all_logs.txt):