SPRING 2023 – CSE 344 SYSTEM PROGRAMMING

FINAL PROJECT

MERT GÜRŞİMŞİR

1901042646

USAGE

First use the “make” command.

Then you will have your BibakBOXClient and BibakBOXServer executable files.

Use these to execute them:

```
BibakBOXServer <directory> <threadPoolSize> <portNumber>

BibakBOXClient <directory > <portNumber> [IP]
```

WARNING!!! → Your executables must be in the same path given by the directory. So you should execute them at the directory that you give as input.

System Architecture, Design Decisions & Implementation Details

To understand the needs of this project we have to know several concepts about system programming that we learnt through the semester which can be simple summarized as follows:

- File input/output
- Signals and Processes
- Communication
- Synchronization
- Threads
- Sockets

I have started to homework by doing simple search in the directory by going through each file/subdirectories and take snapshots. Then I compare snapshot with one former snapshot to detect the changes in the directory. This was my first implementation.

Then I started to make a system design. Threads enables us to do concurrent programming with multiple tasks. So firstly, I have to determine how many threads I am going to use. So I have determined on this design:

- SERVER THREADS
  - Listener thread: Listens for any connection. Puts any incoming connection to array.
  - Checker thread: Always checks changed files. Puts them in array.
  - Sender thread: Sends changed files to client(s).
  - Thread pool: Takes one client from the array and does directory operations.
- CLIENT THREADS
  - Checker thread: Always checks changed files. Puts them in array.
  - Sender thread: Sends changed files to server.
  - Receiver thread: Receive changed files and do directory operations, write, etc.

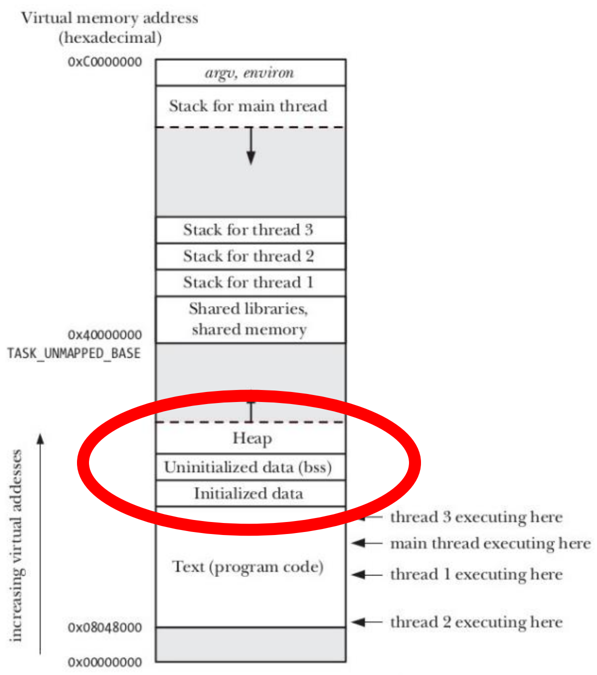
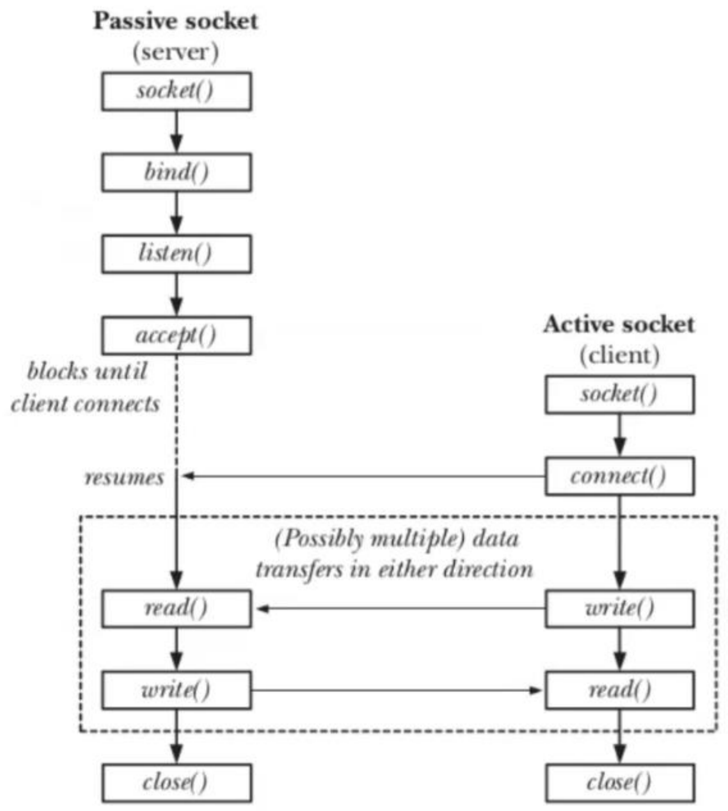


Figure 29-1: Four threads executing in a process (Linux/x86-32)

I have managed communication between threads with global variables and pthread mutexes. That's because threads can access to same memory place (shown with red circle at the left) but there have to be some kind of synchronization so that no two threads try to change the same variable.

So I have created critical sections while I am accessing to global variables so that no inconvenience occurs.

Then socket part comes in. I have checked this figure for lots of time while I am implementing the sockets:



With this valid steps, it was clearer to look what I am doing and follow the each operation.

I used blocks to send data from server to client and from client to server. I have sent filename, type of the operation (modify, delete, etc.), and data for the related file. Each block size is 2048 bytes so I have sent the data block by block which are 2048 bytes each.

On the server side:

- Sender thread → Checks for if there is a change in the directory and send this to client by sendFile function. This function iterates through the array that holds client and their socket number. Then send this data to the related client block by block.
- Changer thread → Take snapshots of the directory and compare it with one former snapshot that is taken by it again. Then adds changed files to an array with creating a critical section with mutex.
- Listener thread → Listens and accepts connections. When a new client connected to server, this thread adds this client to the array (which can be thought as queue) by creating a critical section.
- Thread pool → Each thread in this pool takes care of a client. When client disconnects, another thread from the pool takes the next thread from the queue. Then execute task for the client (receiving information by blocks).

On the client side:

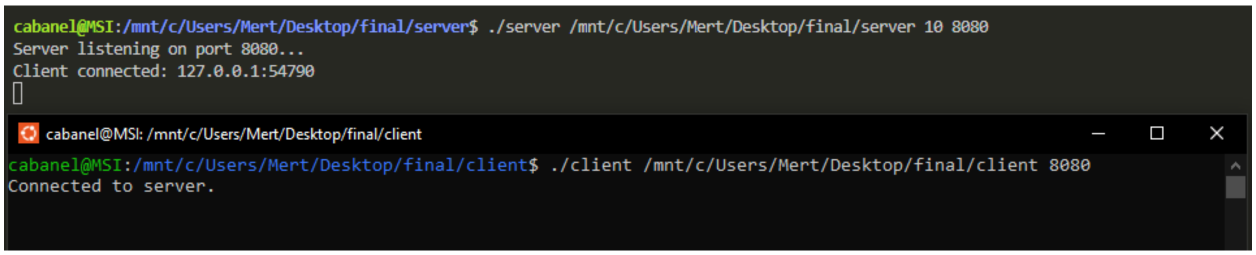
- Receiver thread → Works in an infinite loop. It simply receives data that is coming from the server and do what is needed like creating the file or folder, removing them, etc.
- Sender thread → Works as the same logic with server side sender thread. Instead of sending to each connected client, it just sends to server.
- Checker thread → This one also works as the same logic with server side checker thread.

I have faced with so many difficulties. Synchronizing, signal handling and bug fixing were the hardest parts in this project. I have tried to get through them by trying so many different methods which are things that we have seen in lectures. It was so hard to follow to code at some point but I have followed the figures in the book. At the end, I have successfully built a system that works with many clients in same or different machines.

TEST CASES

1. Can one client connect to the server successfully?
2. Can 2 clients connect to server and run concurrently when pool size is at least 2?
3. Can 2+ clients connect to server and run concurrently?
4. Can server and client run on separate machines?
5. Are files deleted from clients when they are deleted by the server?
6. When a file is modified at the client, can it be seen from the server and other clients?
7. When a new file is added to a client, can it be seen from the server and other clients?
8. When client first connected to server, can it take the files/folders from the server?
9. Is log file creating correctly under the related client's folder?
10. When client exits, can server detects it?
11. When server exits, can clients exits too?

Can one client connect to the server successfully?

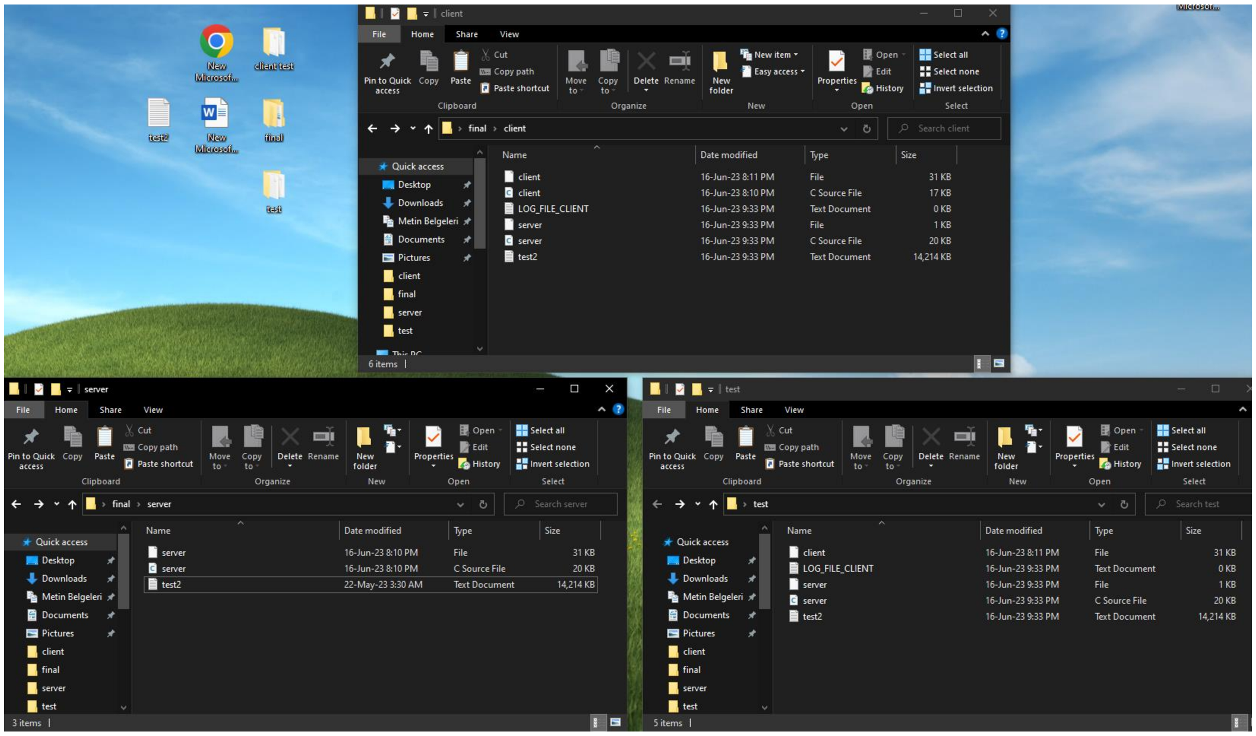


```
cabanel@MSI:/mnt/c/Users/Mert/Desktop/final/server$ ./server /mnt/c/Users/Mert/Desktop/final/server 10 8080
Server listening on port 8080...
Client connected: 127.0.0.1:54790
```

cabanel@MSI: /mnt/c/Users/Mert/Desktop/final/client

```
cabanel@MSI:/mnt/c/Users/Mert/Desktop/final/client$ ./client /mnt/c/Users/Mert/Desktop/final/client 8080
Connected to server.
```

<u>Can 2 clients connect to server and run concurrently when pool size is at least 2?</u>

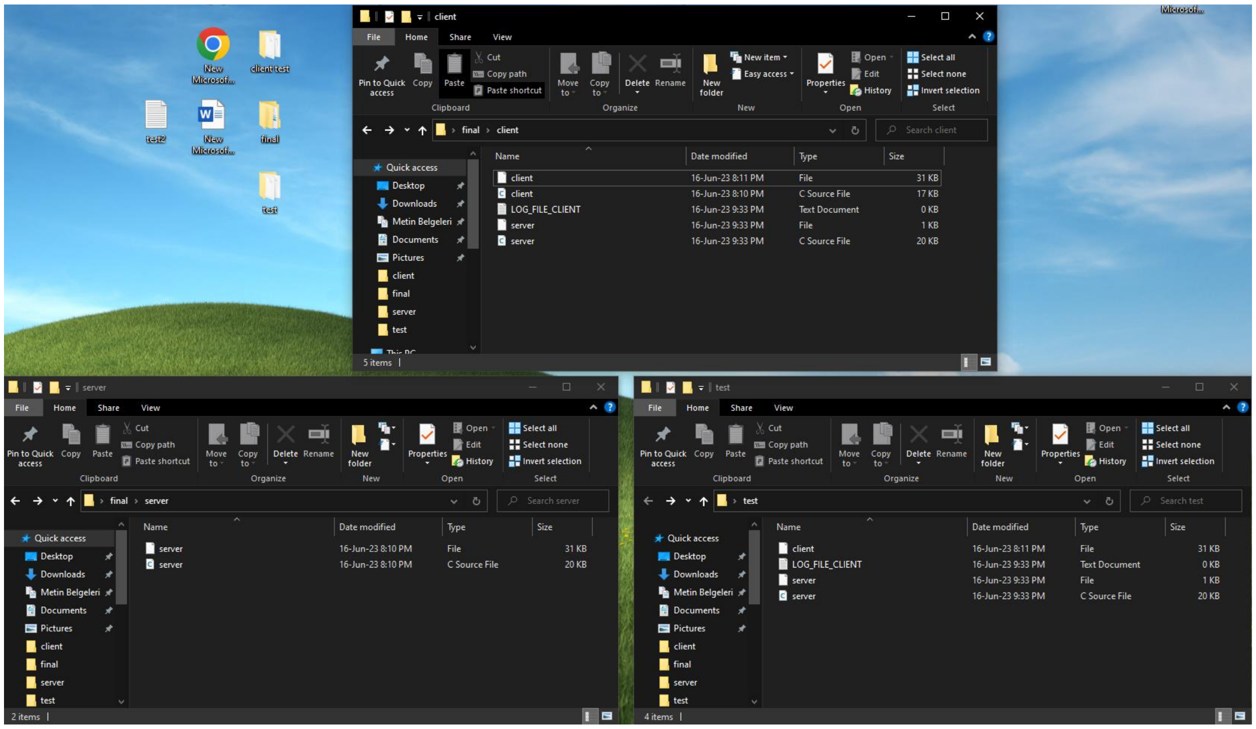


Name	Date modified	Type	Size
client	16-Jun-23 8:11 PM	File	31 KB
client	16-Jun-23 8:10 PM	C Source File	17 KB
LOG_FILE_CLIENT	16-Jun-23 9:33 PM	Text Document	0 KB
server	16-Jun-23 9:33 PM	File	1 KB
server	16-Jun-23 9:33 PM	C Source File	20 KB
test2	16-Jun-23 9:33 PM	Text Document	14,214 KB

Name	Date modified	Type	Size
server	16-Jun-23 8:10 PM	File	31 KB
server	16-Jun-23 8:10 PM	C Source File	20 KB
test2	22-May-23 3:30 AM	Text Document	14,214 KB

Name	Date modified	Type	Size
client	16-Jun-23 8:11 PM	File	31 KB
LOG_FILE_CLIENT	16-Jun-23 9:33 PM	Text Document	0 KB
server	16-Jun-23 9:33 PM	File	1 KB
server	16-Jun-23 9:33 PM	C Source File	20 KB
test2	16-Jun-23 9:33 PM	Text Document	14,214 KB

This is the situation of files, left one is server, middle one is first client, right one is second client. They take all files of server. If we delete test2.txt from second client, it will be deleted from server and first client too:



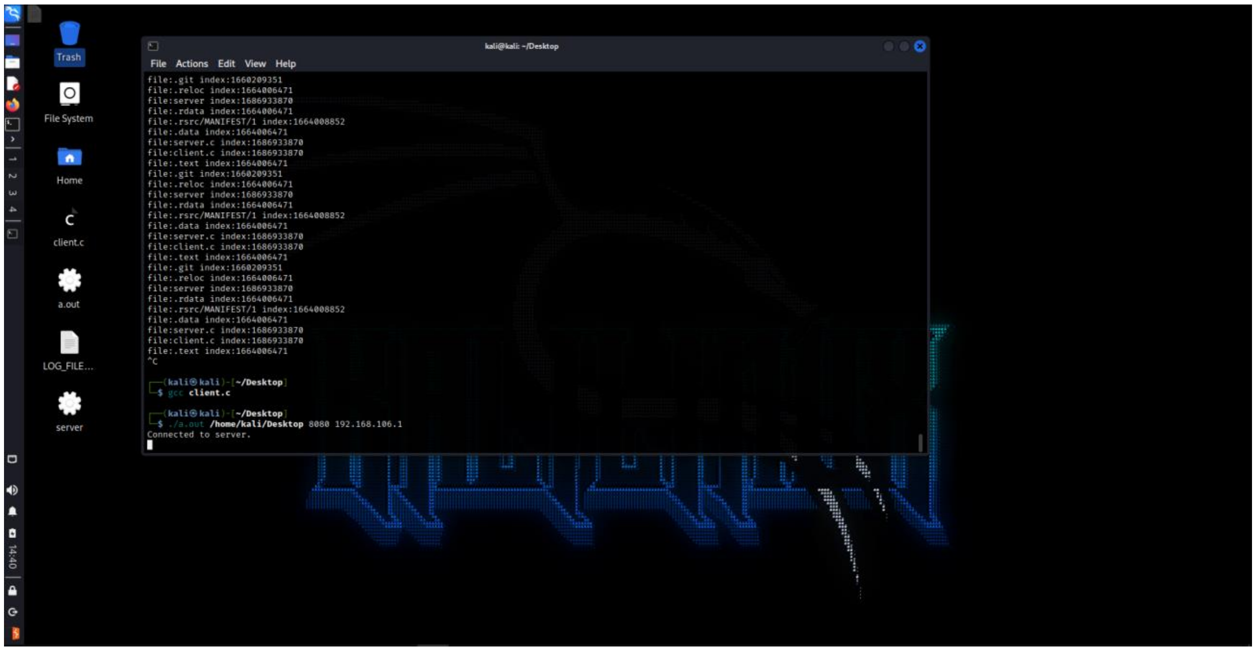
Name	Date modified	Type	Size
client	16-Jun-23 8:11 PM	File	31 KB
client	16-Jun-23 8:10 PM	C Source File	17 KB
LOG_FILE_CLIENT	16-Jun-23 9:33 PM	Text Document	0 KB
server	16-Jun-23 9:33 PM	File	1 KB
server	16-Jun-23 9:33 PM	C Source File	20 KB

Name	Date modified	Type	Size
server	16-Jun-23 8:10 PM	File	31 KB
server	16-Jun-23 8:10 PM	C Source File	20 KB

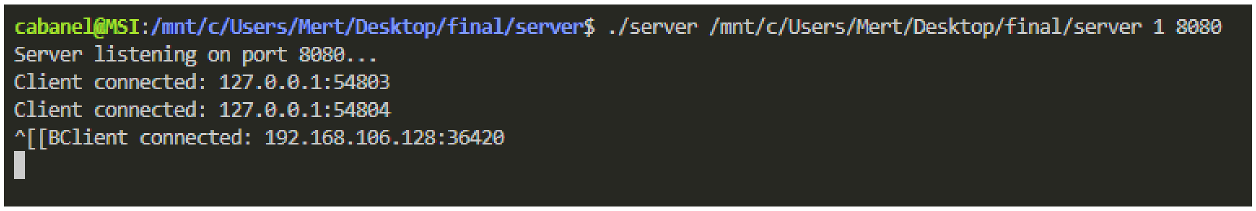
Name	Date modified	Type	Size
client	16-Jun-23 8:11 PM	File	31 KB
LOG_FILE_CLIENT	16-Jun-23 9:33 PM	Text Document	0 KB
server	16-Jun-23 9:33 PM	File	1 KB
server	16-Jun-23 9:33 PM	C Source File	20 KB

<u>Can 2+ clients connect to server and run concurrently? & Can server and client run on separate machines?</u>

Let’s 2 clients stay like above and we add another client from a separate machine:



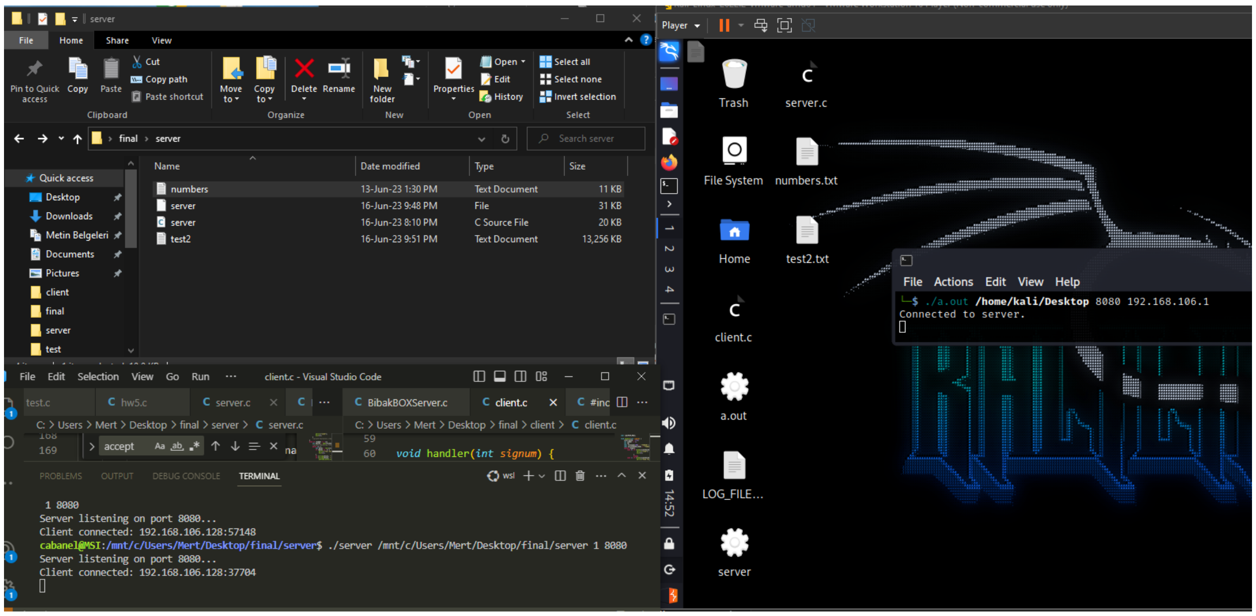
```
file:.git index:1660209351
file:.reloc index:1664006471
file:server index:1686933870
file:.rdata index:1664006471
file:.rsrc/MANIFEST/1 index:1664008852
file:.data index:1664006471
file:server.c index:1686933870
file:client.c index:1686933870
file:.text index:1664006471
file:.git index:1660209351
file:.reloc index:1664006471
file:server index:1686933870
file:.rdata index:1664006471
file:.rsrc/MANIFEST/1 index:1664008852
file:.data index:1664006471
file:server.c index:1686933870
file:client.c index:1686933870
file:.text index:1664006471
file:.git index:1660209351
file:.reloc index:1664006471
file:server index:1686933870
file:.rdata index:1664006471
file:.rsrc/MANIFEST/1 index:1664008852
file:.data index:1664006471
file:server.c index:1686933870
file:client.c index:1686933870
file:.text index:1664006471
^C

┌──(kali㉿kali)-[~/Desktop]
└─$ gcc client.c

┌──(kali㉿kali)-[~/Desktop]
└─$ ./a.out /home/kali/Desktop 8080 192.168.106.1
Connected to server.
```



```
cabanel@MSI:/mnt/c/Users/Mert/Desktop/final/server$ ./server /mnt/c/Users/Mert/Desktop/final/server 1 8080
Server listening on port 8080...
Client connected: 127.0.0.1:54803
Client connected: 127.0.0.1:54804
^[[BClient connected: 192.168.106.128:36420
```

So this one can also connect to the server.

Let’s add numbers.txt and test2.txt to this separate machine:



Name	Date modified	Type	Size
numbers	13-Jun-23 1:30 PM	Text Document	11 KB
server	16-Jun-23 9:48 PM	File	31 KB
server	16-Jun-23 8:10 PM	C Source File	20 KB
test2	16-Jun-23 9:51 PM	Text Document	13,256 KB

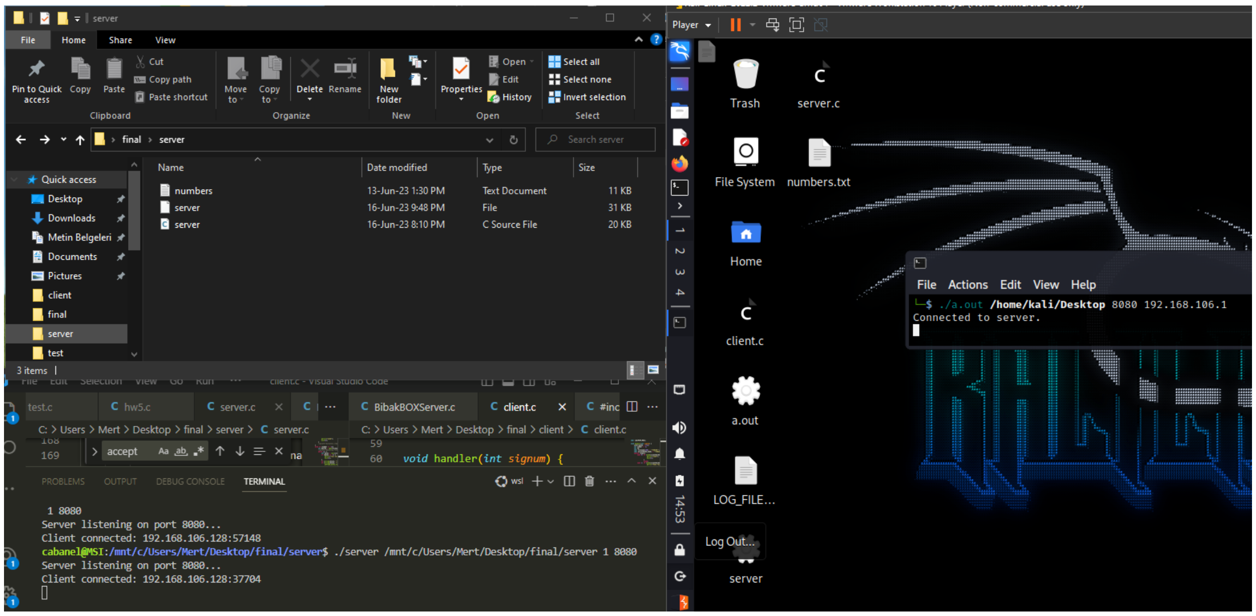
```
 1 8080
Server listening on port 8080...
Client connected: 192.168.106.128:57148
cabanel@MSI:/mnt/c/Users/Mert/Desktop/final/server$ ./server /mnt/c/Users/Mert/Desktop/final/server 1 8080
Server listening on port 8080...
Client connected: 192.168.106.128:37704
```

```
└─$ ./a.out /home/kali/Desktop 8080 192.168.106.1
Connected to server.
```

As it can be seen, we can see the changes on both client and server side.

Are files deleted from clients when they are deleted by the server?

We can fastly delete test2.txt and see the effect:

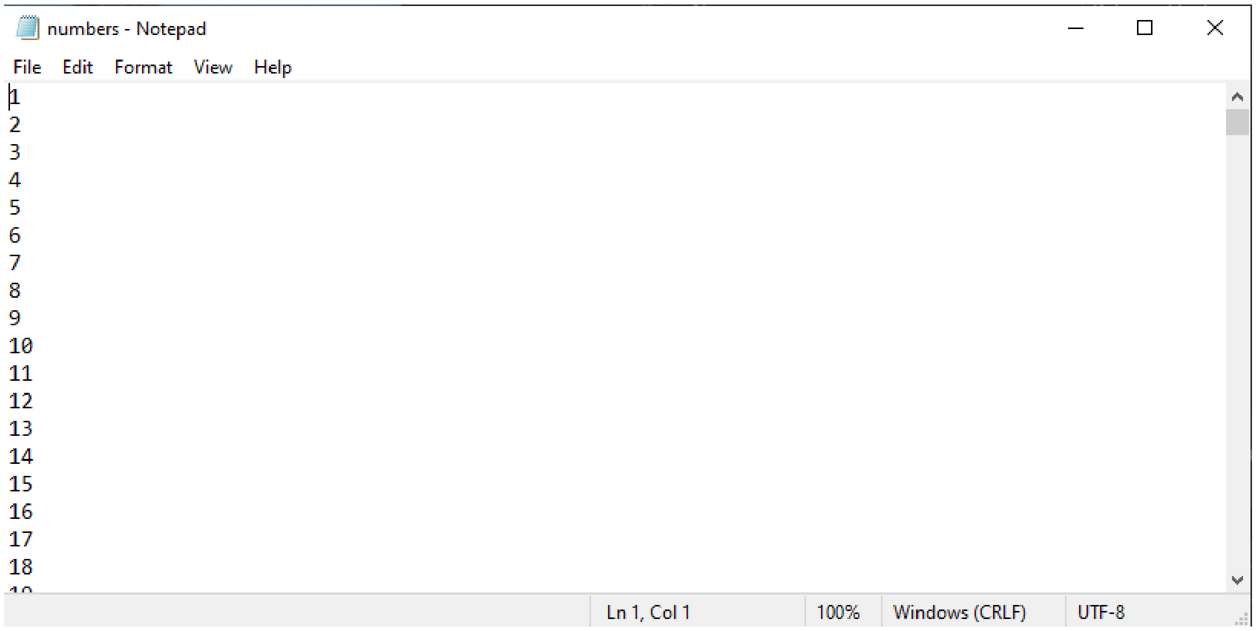


Name	Date modified	Type	Size
numbers	13-Jun-23 1:30 PM	Text Document	11 KB
server	16-Jun-23 9:48 PM	File	31 KB
server	16-Jun-23 8:10 PM	C Source File	20 KB

```
 1 8080
Server listening on port 8080...
Client connected: 192.168.106.128:57148
cabanel@MSI:/mnt/c/Users/Mert/Desktop/final/server$ ./server /mnt/c/Users/Mert/Desktop/final/server 1 8080
Server listening on port 8080...
Client connected: 192.168.106.128:37704
```

```
└─$ ./a.out /home/kali/Desktop 8080 192.168.106.1
Connected to server.
```

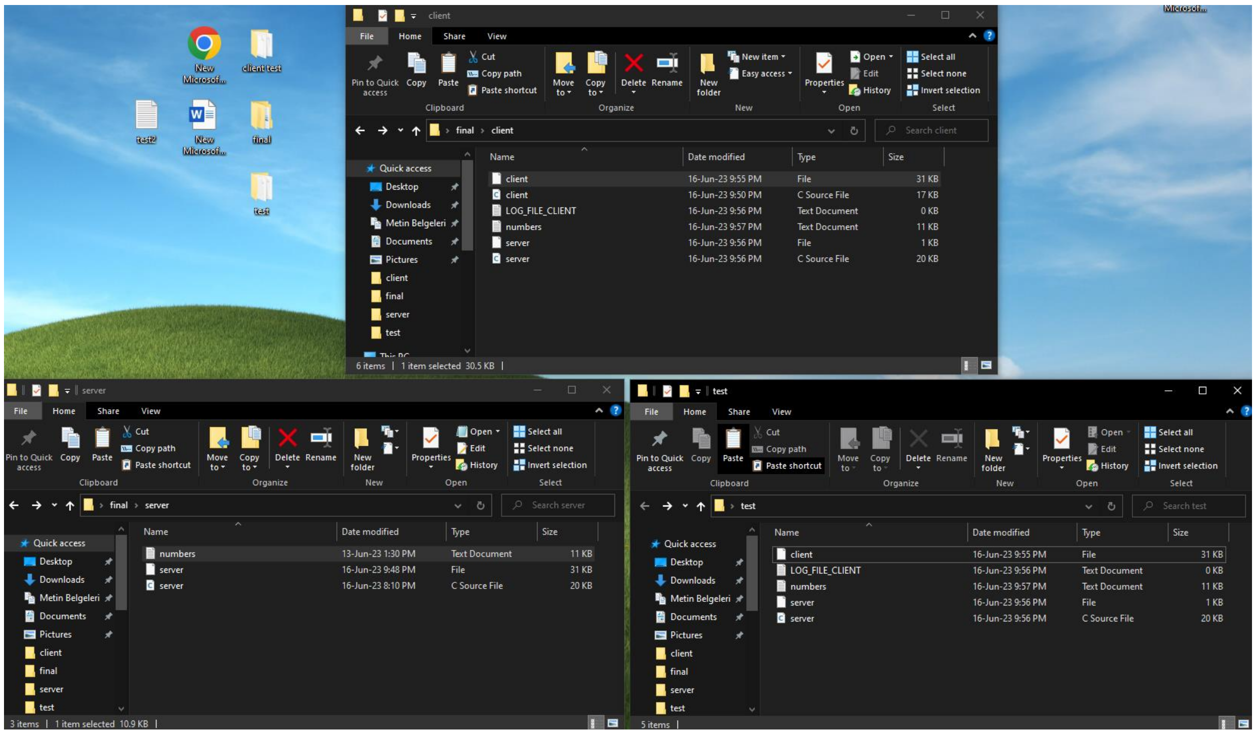
<u>When a file is modified at the client, can it be seen from the server and other clients?</u>


numbers - Notepad

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
```

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

numbers.txt at first. See all clients and server have this file:


client — final › client

Name	Date modified	Type	Size
client	16-Jun-23 9:55 PM	File	31 KB
client	16-Jun-23 9:50 PM	C Source File	17 KB
LOG_FILE_CLIENT	16-Jun-23 9:56 PM	Text Document	0 KB
numbers	16-Jun-23 9:57 PM	Text Document	11 KB
server	16-Jun-23 9:56 PM	File	1 KB
server	16-Jun-23 9:56 PM	C Source File	20 KB

6 items | 1 item selected 30.5 KB

server — final › server

Name	Date modified	Type	Size
numbers	13-Jun-23 1:30 PM	Text Document	11 KB
server	16-Jun-23 9:48 PM	File	31 KB
server	16-Jun-23 8:10 PM	C Source File	20 KB

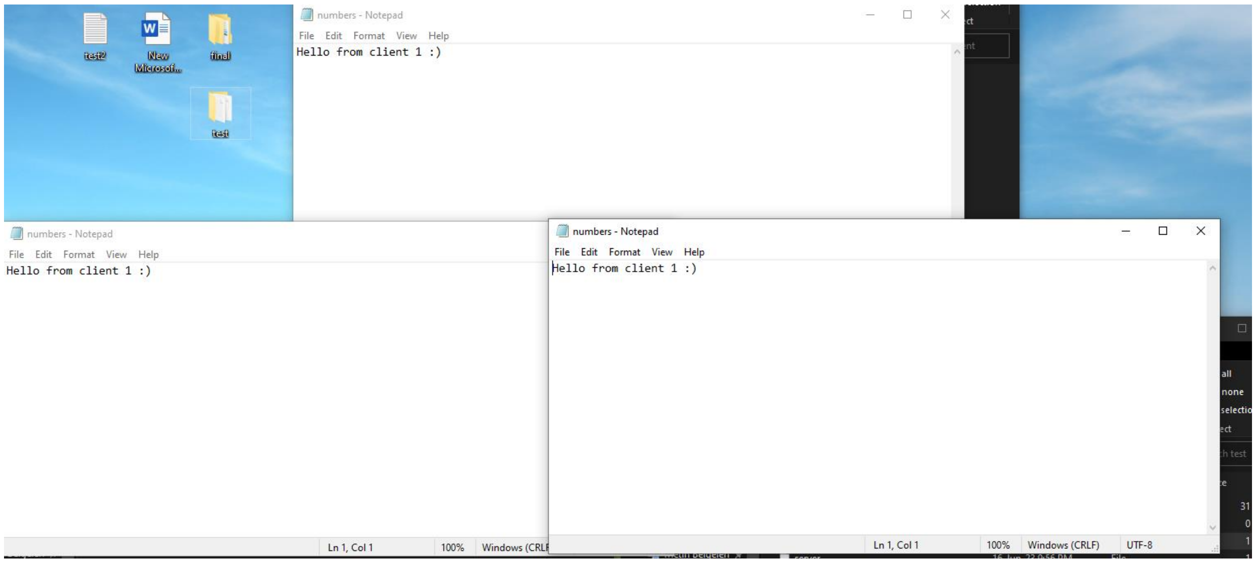
3 items | 1 item selected 10.9 KB

test — test

Name	Date modified	Type	Size
client	16-Jun-23 9:55 PM	File	31 KB
LOG_FILE_CLIENT	16-Jun-23 9:56 PM	Text Document	0 KB
numbers	16-Jun-23 9:57 PM	Text Document	11 KB
server	16-Jun-23 9:56 PM	File	1 KB
server	16-Jun-23 9:56 PM	C Source File	20 KB

5 items

Now change numbers.txt to → “Hello from client 1 :)”


numbers - Notepad

```
Hello from client 1 :)
```

numbers - Notepad

```
Hello from client 1 :)
```

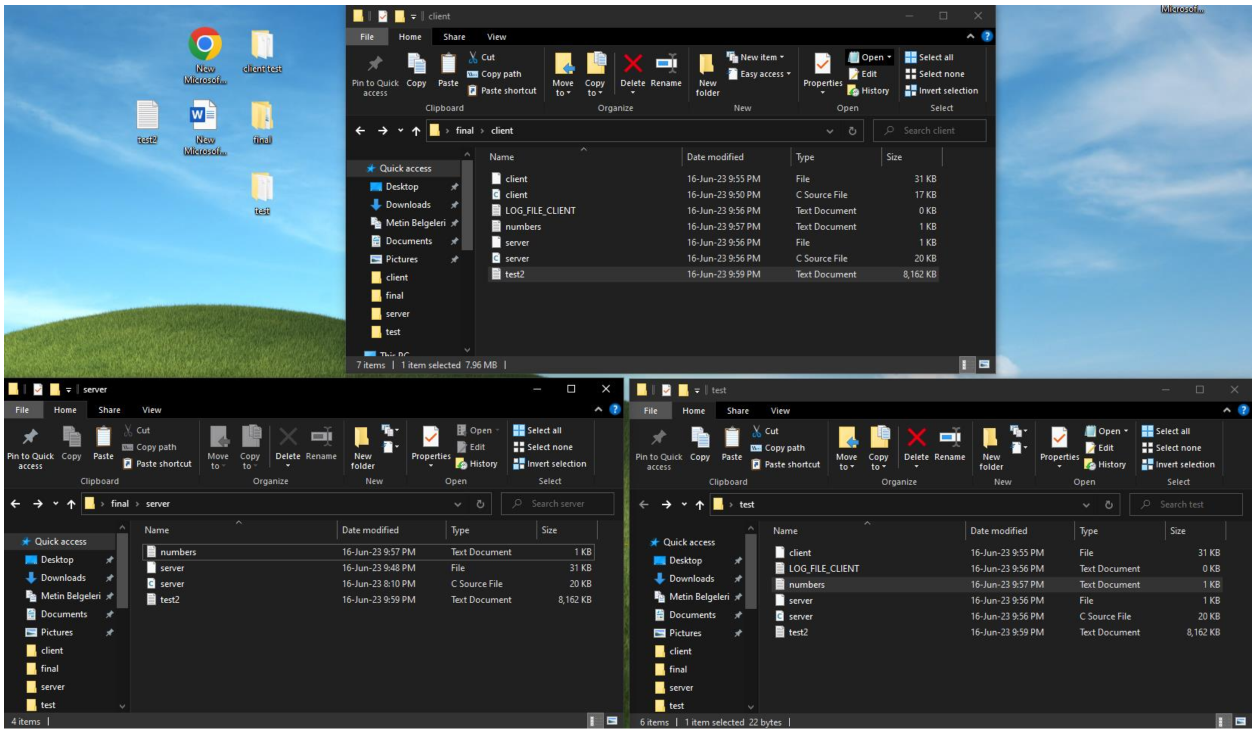
numbers - Notepad

```
Hello from client 1 :)
```

All files have changed successfully.

<u>When a new file is added to a client, can it be seen from the server and other clients?</u>

We can add test2.txt to client1 and see if server and other clients can have this file also:


final › client

Name	Date modified	Type	Size
client	16-Jun-23 9:55 PM	File	31 KB
client	16-Jun-23 9:50 PM	C Source File	17 KB
LOG_FILE_CLIENT	16-Jun-23 9:56 PM	Text Document	0 KB
numbers	16-Jun-23 9:57 PM	Text Document	1 KB
server	16-Jun-23 9:56 PM	File	1 KB
server	16-Jun-23 9:56 PM	C Source File	20 KB
test2	16-Jun-23 9:59 PM	Text Document	8,162 KB

7 items | 1 item selected 7.96 MB

final › server

Name	Date modified	Type	Size
numbers	16-Jun-23 9:57 PM	Text Document	1 KB
server	16-Jun-23 9:48 PM	File	31 KB
server	16-Jun-23 8:10 PM	C Source File	20 KB
test2	16-Jun-23 9:59 PM	Text Document	8,162 KB

4 items

test

Name	Date modified	Type	Size
client	16-Jun-23 9:55 PM	File	31 KB
LOG_FILE_CLIENT	16-Jun-23 9:56 PM	Text Document	0 KB
numbers	16-Jun-23 9:57 PM	Text Document	1 KB
server	16-Jun-23 9:56 PM	File	1 KB
server	16-Jun-23 9:56 PM	C Source File	20 KB
test2	16-Jun-23 9:59 PM	Text Document	8,162 KB

6 items | 1 item selected 22 bytes

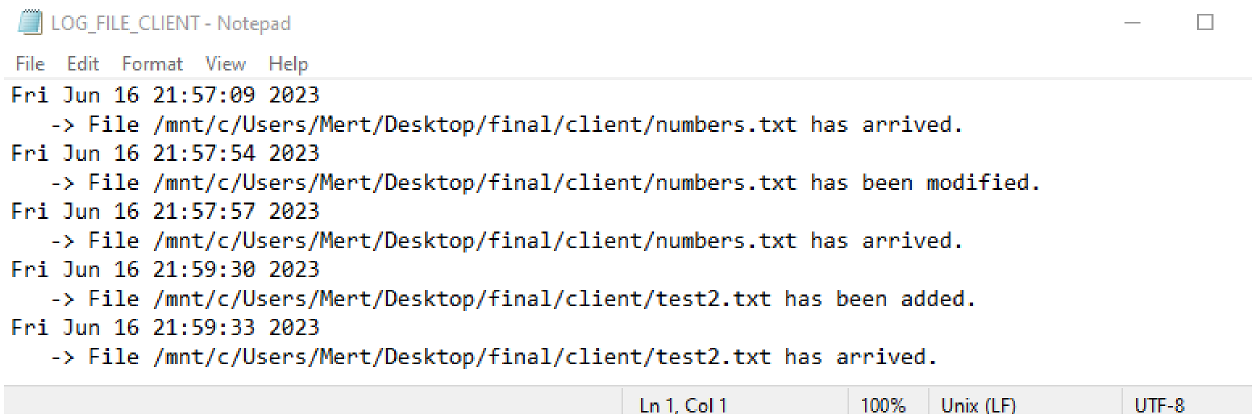
As you can see, all clients and server has the same test2.txt file. You can check the size.

<u>When client first connected to server, can it take the files/folders from the server?</u>

You can see previous outputs. They all have server.c and server executable file under their directory.

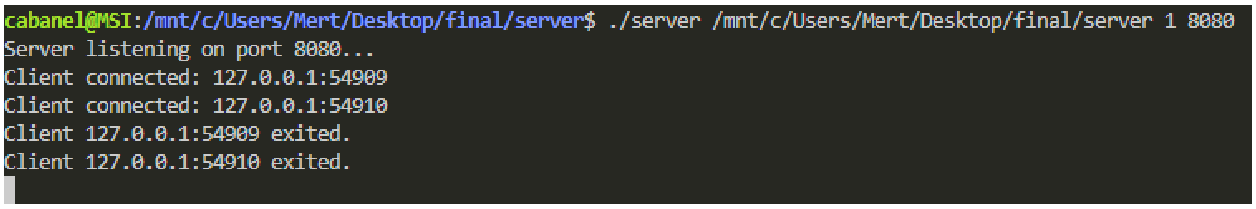
<u>Is log file creating correctly under the related client's folder?</u>

Log file is created for the client as follows:


LOG_FILE_CLIENT - Notepad
File Edit Format View Help
Ln 1, Col 1 100% Unix (LF) UTF-8

```
Fri Jun 16 21:57:09 2023
   -> File /mnt/c/Users/Mert/Desktop/final/client/numbers.txt has arrived.
Fri Jun 16 21:57:54 2023
   -> File /mnt/c/Users/Mert/Desktop/final/client/numbers.txt has been modified.
Fri Jun 16 21:57:57 2023
   -> File /mnt/c/Users/Mert/Desktop/final/client/numbers.txt has arrived.
Fri Jun 16 21:59:30 2023
   -> File /mnt/c/Users/Mert/Desktop/final/client/test2.txt has been added.
Fri Jun 16 21:59:33 2023
   -> File /mnt/c/Users/Mert/Desktop/final/client/test2.txt has arrived.
```

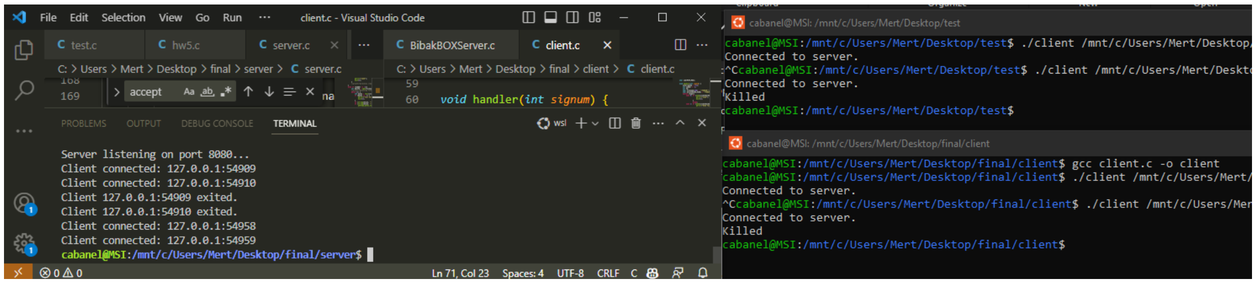
<u>When client exits, can server detects it?</u>



```
cabanel@MSI:/mnt/c/Users/Mert/Desktop/final/server$ ./server /mnt/c/Users/Mert/Desktop/final/server 1 8080
Server listening on port 8080...
Client connected: 127.0.0.1:54909
Client connected: 127.0.0.1:54910
Client 127.0.0.1:54909 exited.
Client 127.0.0.1:54910 exited.
```

It prints out the proper message.

<u>When server exits, can clients exits too?</u>


File Edit Selection View Go Run ··· client.c - Visual Studio Code
test.c hw5.c server.c BibakBOXServer.c client.c
C: > Users > Mert > Desktop > final > server > server.c
C: > Users > Mert > Desktop > final > client > client.c
accept
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
wsl
Ln 71, Col 23 Spaces: 4 UTF-8 CRLF C
cabanel@MSI: /mnt/c/Users/Mert/Desktop/test
cabanel@MSI: /mnt/c/Users/Mert/Desktop/final/client

```
void handler(int signum) {
```

```
Server listening on port 8080...
Client connected: 127.0.0.1:54909
Client connected: 127.0.0.1:54910
Client 127.0.0.1:54909 exited.
Client 127.0.0.1:54910 exited.
Client connected: 127.0.0.1:54958
Client connected: 127.0.0.1:54959
cabanel@MSI:/mnt/c/Users/Mert/Desktop/final/server$
```

```
cabanel@MSI:/mnt/c/Users/Mert/Desktop/test$ ./client /mnt/c/Users/Mert/Desktop/
Connected to server.
^Ccabanel@MSI:/mnt/c/Users/Mert/Desktop/test$ ./client /mnt/c/Users/Mert/Deskto
Connected to server.
Killed
cabanel@MSI:/mnt/c/Users/Mert/Desktop/test$
```

```
cabanel@MSI:/mnt/c/Users/Mert/Desktop/final/client$ gcc client.c -o client
cabanel@MSI:/mnt/c/Users/Mert/Desktop/final/client$ ./client /mnt/c/Users/Mert/
Connected to server.
^Ccabanel@MSI:/mnt/c/Users/Mert/Desktop/final/client$ ./client /mnt/c/Users/Mer
Connected to server.
Killed
cabanel@MSI:/mnt/c/Users/Mert/Desktop/final/client$
```

When Ctrl+C signal is taken from the server, all other clients will be killed.