

# NETWORKS LABORATORY ASSIGNMENT 1

JEREMIAH THOMAS

106119055 | CSE -A

## Do Note:

- Kindly find all the codes in the zip folder as requested
- Do cd into the right directory while running the programs.
- It is preferred to run the programs on a Linux environment.
- Before running the codes for ethernet related interfaces, do ensure that the particular IP address on the local machine matches that of the code.
- To change the IP address to match the code, one can look up :  
<https://unix.stackexchange.com/questions/152331/how-can-i-create-a-virtual-ethernet-interface-on-a-machine-without-a-physical-ad/152334#152334?newreg=fae89b45bb1c4960ae05e6eea9554922>

---

## Question 1:-

- 1) Write a server program for TCP using Python to do the following:
  - a. Server returns the binary value of the text sent by the client. Example: for a text string “commnetsii”, the client should receive “01100011 01101111 01101101 01101110 01100101 01110100 01110011 01101001 01101001”
  - b. The server should be running at the localhost interface
  - c. You are free to choose any port

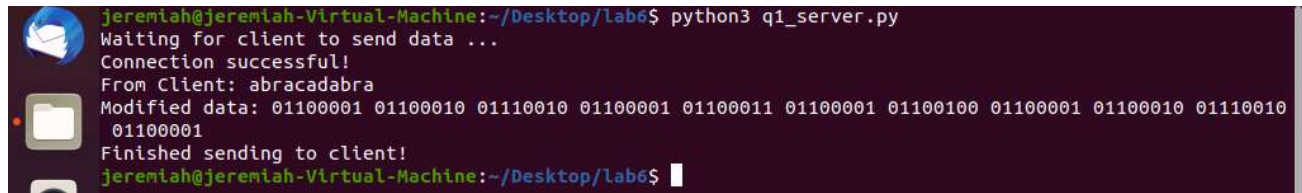
## Output:-

## Client Side:-



```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q1_client.py
Enter the text:abracadabra
Host IP address: 127.0.0.1
Success connecting to the server!
Sending text to the server ...
Waiting for result from the server ...
Result from the server is: 01100001 01100010 01110010 01100001 01100011 01100001 01100100 01100001 01100
010 01110010 01100001
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$
```

## Server Side:-

A terminal window with a dark purple background. The prompt is 'jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6\$'. The command 'python3 q1\_server.py' has been executed. The output shows: 'Waiting for client to send data ...', 'Connection successful!', 'From Client: abracadabra', 'Modified data: 01100001 01100010 01110010 01100001 01100011 01100001 01100100 01100001 01100010 01110010 01100001', 'Finished sending to client!', and the prompt 'jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6\$' again.

```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q1_server.py
Waiting for client to send data ...
Connection successful!
From Client: abracadabra
Modified data: 01100001 01100010 01110010 01100001 01100011 01100001 01100100 01100001 01100010 01110010 01100001
Finished sending to client!
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$
```

## Observations and Inferences:-

- Since it is TCP, a connection must be established before sending/receiving data and the outputs above stay true to that as you can see that first, a connection is established between the server and client, following which a text message is sent for processing.
- Also, the client sends his text across to the server which is processed by a `strToBinary()` fn. This is a simple illustration of the client-server paradigm with a backend where all the backend code is private and secured.
- The client has no access to this function `strToBinary()` but can only make use of the platform that the server provides and in the way the server wants.
- Thus, the server logic (could be metaphorically called “Intellectual Property”) is safe, secured and abstracted from the client.

---

## Question 2:-

- 2) The Binary Decoding Server: Write another server running on `eth1` interface to do the following:

  - a. Server returns the string value of a binary input. Example: the binary string “01100011 01101111 01101101 01101110 01100101 01110100 01110011 01101001 01101001” sent from the client should return “connetsii”

## Output:-

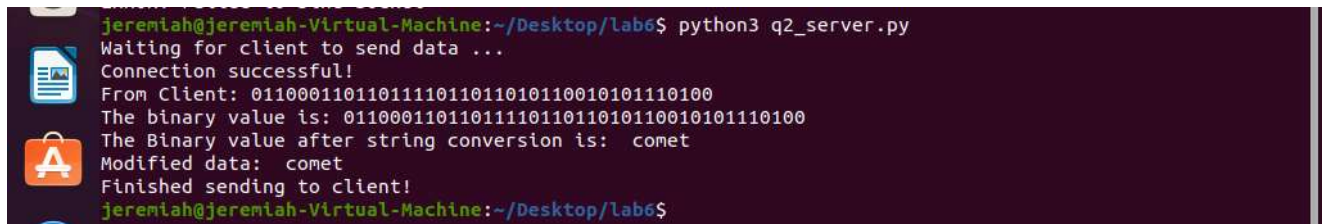
(Do Scroll Down)

## Client Side:-

A terminal window with a dark purple background. On the left, there are three icons: a yellow circle with a black dot, a blue document icon, and an orange shopping bag icon. The terminal text shows the execution of a Python script that sends a binary string to a server and receives the response 'comet'.

```
Result from the server is: comet
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q2_client.py
Enter the binary string:0110001101101111011011010110010101110100
Ethernet IP address: 13.3.3.3
Success connecting to the server!
Sending text to the server ...
Waiting for result from the server ...
Result from the server is: comet
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$
```

## Server Side:-

A terminal window with a dark purple background. On the left, there are three icons: a blue document icon, an orange shopping bag icon, and a blue circle icon. The terminal text shows the execution of a Python script that receives a binary string from a client, converts it to a string, and sends back the response 'comet'.

```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q2_server.py
Waiting for client to send data ...
Connection successful!
From Client: 0110001101101111011011010110010101110100
The binary value is: 0110001101101111011011010110010101110100
The Binary value after string conversion is: comet
Modified data: comet
Finished sending to client!
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$
```

## Observations and Inferences:-

- To be honest, this is very closely linked to Q1. So, the findings for q1 are nearly the same here as well.
- However, in this program, the connecting interface is the ethernet interface which I've set using the [sudo ip address change dev] command to "13.3.3.3."
- With the knowledge I possess as of now and also from a high-level view, it is hard to say where the difference lies.
- One low-level difference might be the speed that they possess. Ethernet must obviously be much faster than the former as it's a physical connection.
- However, we would need some tools to measure the speed between sockets.

-----

(Do Scroll Down)

### Question 3:-

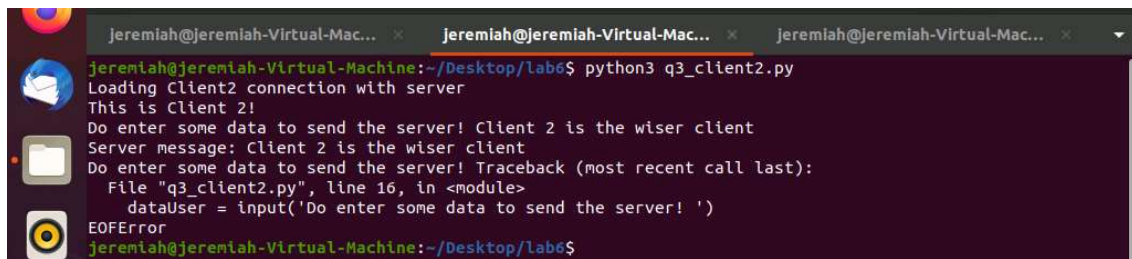
- 3) Write two client programs one for each of the server.
- What happens if the client aborts (e.g., when the input is CTRL/D)?
  - Can you run two clients against the same server? Why or why not? Hint: Multithreaded socket server in Python
  - What happens when you try to connect client1 to server2 by passing a wrong network address (e.g., connecting to localhost instead of eth1 IP)?

### Output & Answer:-

#### a)

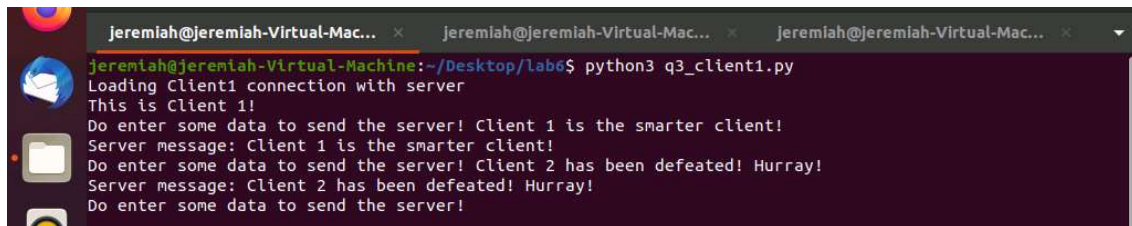
To test this, I have taken Client 2 as the client to abort.  
This is what I observed: -

#### Client 2:-

A terminal window showing the execution of a Python script named q3\_client2.py. The user is at the prompt 'jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6\$'. The script outputs: 'Loading Client2 connection with server', 'This is Client 2!', 'Do enter some data to send the server! Client 2 is the wiser client', and 'Server message: Client 2 is the wiser client'. Then it prompts 'Do enter some data to send the server! Traceback (most recent call last):' followed by a file path and line number. The user enters 'dataUser = input('Do enter some data to send the server! ')', which results in an 'EOFError' and the prompt returns to the shell.

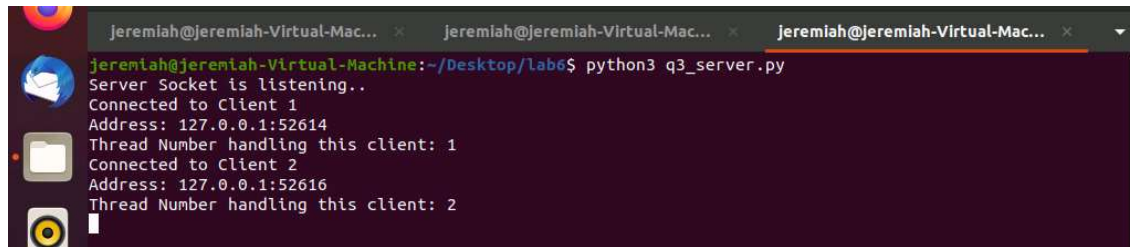
```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q3_client2.py
Loading Client2 connection with server
This is Client 2!
Do enter some data to send the server! Client 2 is the wiser client
Server message: Client 2 is the wiser client
Do enter some data to send the server! Traceback (most recent call last):
  File "q3_client2.py", line 16, in <module>
    dataUser = input('Do enter some data to send the server! ')
EOFError
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$
```

#### Client 1:-

A terminal window showing the execution of a Python script named q3\_client1.py. The user is at the prompt 'jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6\$'. The script outputs: 'Loading Client1 connection with server', 'This is Client 1!', 'Do enter some data to send the server! Client 1 is the smarter client!', 'Server message: Client 1 is the smarter client!', 'Do enter some data to send the server! Client 2 has been defeated! Hurray!', 'Server message: Client 2 has been defeated! Hurray!', and 'Do enter some data to send the server!'.

```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q3_client1.py
Loading Client1 connection with server
This is Client 1!
Do enter some data to send the server! Client 1 is the smarter client!
Server message: Client 1 is the smarter client!
Do enter some data to send the server! Client 2 has been defeated! Hurray!
Server message: Client 2 has been defeated! Hurray!
Do enter some data to send the server!
```

## Server:-

A terminal window with three tabs, all titled 'jeremiah@jeremiah-Virtual-Mac...'. The active tab shows the output of running 'python3 q3\_server.py'. The output is: 'Server Socket is listening..', 'Connected to Client 1', 'Address: 127.0.0.1:52614', 'Thread Number handling this client: 1', 'Connected to Client 2', 'Address: 127.0.0.1:52616', 'Thread Number handling this client: 2'.

```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q3_server.py
Server Socket is listening..
Connected to Client 1
Address: 127.0.0.1:52614
Thread Number handling this client: 1
Connected to Client 2
Address: 127.0.0.1:52616
Thread Number handling this client: 2
```

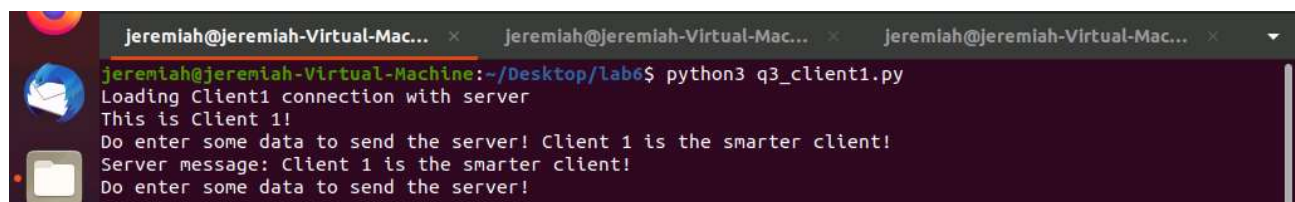
- Client 2 immediately breaks and loses its connection with the server.
- However, client1's connection is unaffected and functions smoothly as you can see. For instance, the data ->(Client 2 has been defeated) was sent after client 1 terminated and the server still echoes it back.
- Thus, because they're handled on 2 separate threads, when one fails, the other still works without a hassle.

---

## b)

- Yes, you can run two clients against the same server. This is possible through the means of a concept called multithreading and in python, one can make use of Multithreaded Socket Servers to achieve the same.
- For instance, in my code, start\_new\_thread is used to spawn a new thread that deals with a client on a unique port with the function passed in its argument, which in this is : multiThreaded(), a fn that essentially servers to echo data from client.

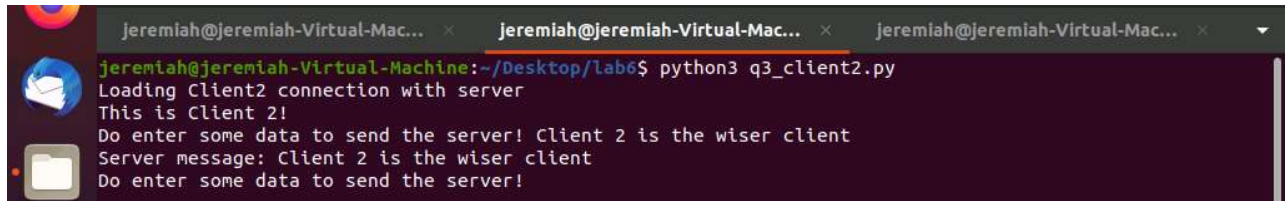
## Client1 Side:-

A terminal window with three tabs, all titled 'jeremiah@jeremiah-Virtual-Mac...'. The active tab shows the output of running 'python3 q3\_client1.py'. The output is: 'Loading Client1 connection with server', 'This is Client 1!', 'Do enter some data to send the server! Client 1 is the smarter client!', 'Server message: Client 1 is the smarter client!', 'Do enter some data to send the server!'.

```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q3_client1.py
Loading Client1 connection with server
This is Client 1!
Do enter some data to send the server! Client 1 is the smarter client!
Server message: Client 1 is the smarter client!
Do enter some data to send the server!
```

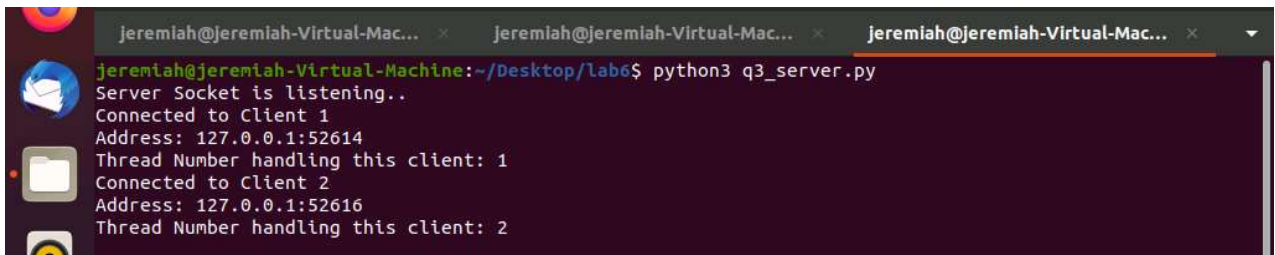
(Do Scroll Down)

## Client2 Side:-

A terminal window with a dark background and light text. The prompt is 'jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6\$'. The command 'python3 q3\_client2.py' has been executed. The output shows the client loading, identifying itself as Client 2, and receiving a message from the server. The user is prompted to enter data to send to the server.

```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q3_client2.py
Loading Client2 connection with server
This is Client 2!
Do enter some data to send the server! Client 2 is the wiser client
Server message: Client 2 is the wiser client
Do enter some data to send the server!
```

## Server Side:-

A terminal window with a dark background and light text. The prompt is 'jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6\$'. The command 'python3 q3\_server.py' has been executed. The output shows the server socket listening, connecting to Client 1, assigning a unique address (127.0.0.1:52614), and spawning a new thread (Thread 1) to handle Client 1. Then, Client 2 connects, and a new thread (Thread 2) is spawned to handle Client 2 simultaneously with Thread 1.

```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q3_server.py
Server Socket is listening..
Connected to Client 1
Address: 127.0.0.1:52614
Thread Number handling this client: 1
Connected to Client 2
Address: 127.0.0.1:52616
Thread Number handling this client: 2
```

- As you can see vividly, Client 1 is the first to make a request of connection to the server. The server immediately gives Client 1 a unique address with a specific port : 52614.
- Furthermore, Client 1 is handled on a new thread called, Thread 1.
- Then, sometime later Client 2 makes a request and the server spawns a new thread(Thread 2) to deal with Client 2 and this thread operates simultaneously at par with Thread 1.
- Also, Server 1 assigns a specific port to client 2 as well :52616.

---

## C)

### Change in Code To Test 3(c):-

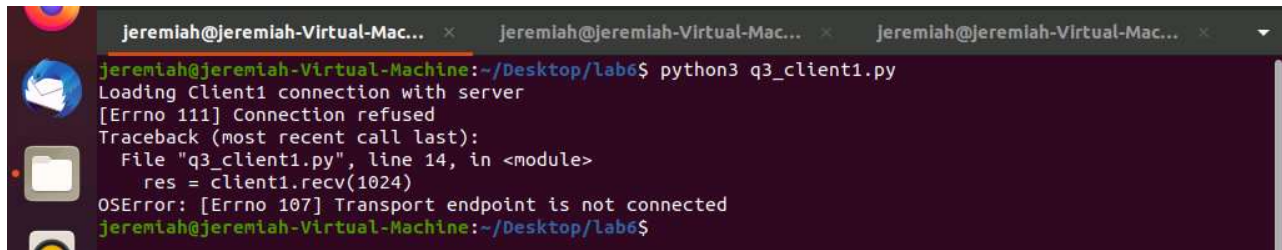
```
L4: eth = '13.3.3.3'
L9: client1.connect((eth, port))
```

(Do Scroll Down)



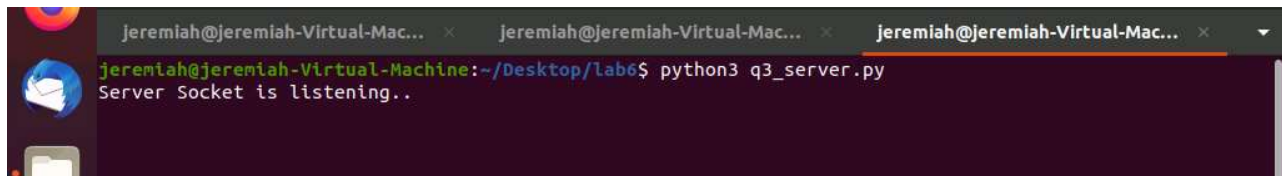
## Output:-

### Client1 Side:-

A terminal window titled 'jeremiah@jeremiah-Virtual-Mac...' shows the execution of 'python3 q3\_client1.py'. The output includes: 'Loading Client1 connection with server', '[Errno 111] Connection refused', a traceback for 'File "q3\_client1.py", line 14, in <module>' showing 'res = client1.recv(1024)', and 'OSError: [Errno 107] Transport endpoint is not connected'.

```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q3_client1.py
Loading Client1 connection with server
[Errno 111] Connection refused
Traceback (most recent call last):
  File "q3_client1.py", line 14, in <module>
    res = client1.recv(1024)
OSError: [Errno 107] Transport endpoint is not connected
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$
```

### Server Side:-

A terminal window titled 'jeremiah@jeremiah-Virtual-Mac...' shows the execution of 'python3 q3\_server.py'. The output is 'Server Socket is listening..'.

```
jeremiah@jeremiah-Virtual-Machine:~/Desktop/lab6$ python3 q3_server.py
Server Socket is listening..
```

- Here as we can see client1 connects to "13.3.3.3" which is the ethernet interface IP in my environment. However, the server socket is based on localhost.
- Thus, there is a clash when client1 requests for a connection and that is why we see the Error, "Connection Refused" and "Transport endpoint is not Connected"
- The server remains idle as seen in the screenshot as it does not handle and its socket is not listening on the IP address, "13.3.3.3"
- To rectify this, we would need to either change the address the server socket is bound to the eth address or change the address from the client side to localhost.

---

THANK YOU!