PES1UG20CS821

Nikhil T M

WEEK 5: IMPLEMENTATION OF A LOCAL DNS SERVER AND AUTHORITATIVE NAMESERVER

Task 1: Socket programming with UDP

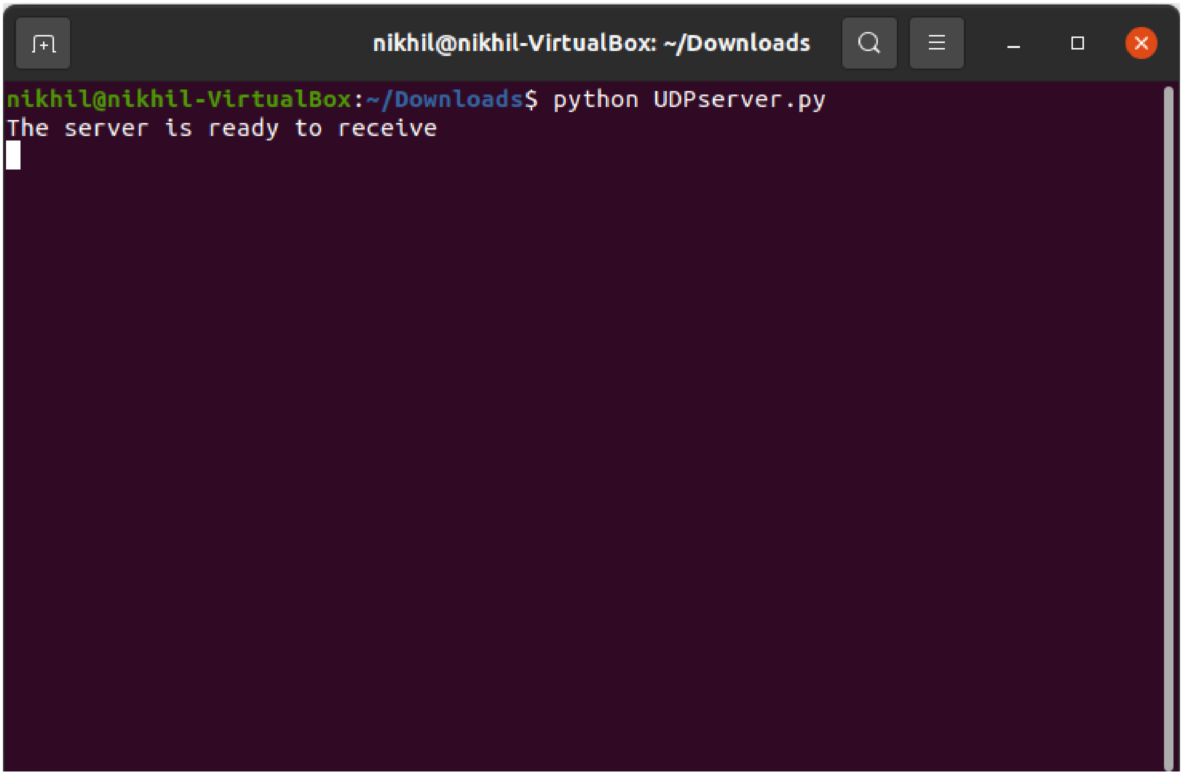
UDPClient.py

```
from socket import *
serverName = "10.0.2.5"
serverPort = 12000
clientSocket = socket(AF_INET,SOCK_DGRAM)
message = raw_input('Input lowercase sentence:')
clientSocket.sendto(message.encode(),(serverName, serverPort))
modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
print(modifiedMessage)
clientSocket.close()
```

UDPServer.py

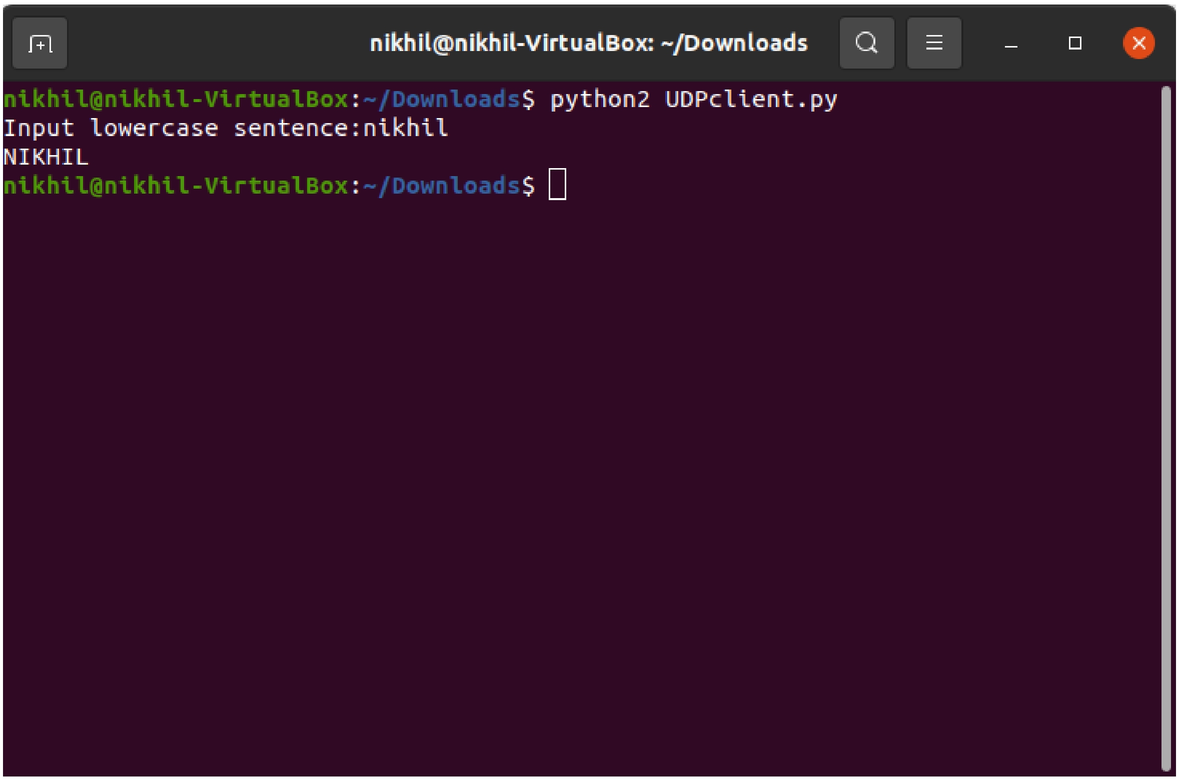
```
from socket import *
server_name = "10.0.2.5"
server_port = 12000
server_socket = socket(AF_INET, SOCK_DGRAM)
server_socket.bind((server_name, server_port))
print("The server is ready to receive")
while True:
    message,client_address = server_socket.recvfrom(2048)
    message = message.upper()
    server_socket.sendto(message, client_address)
```

UDP server in running.



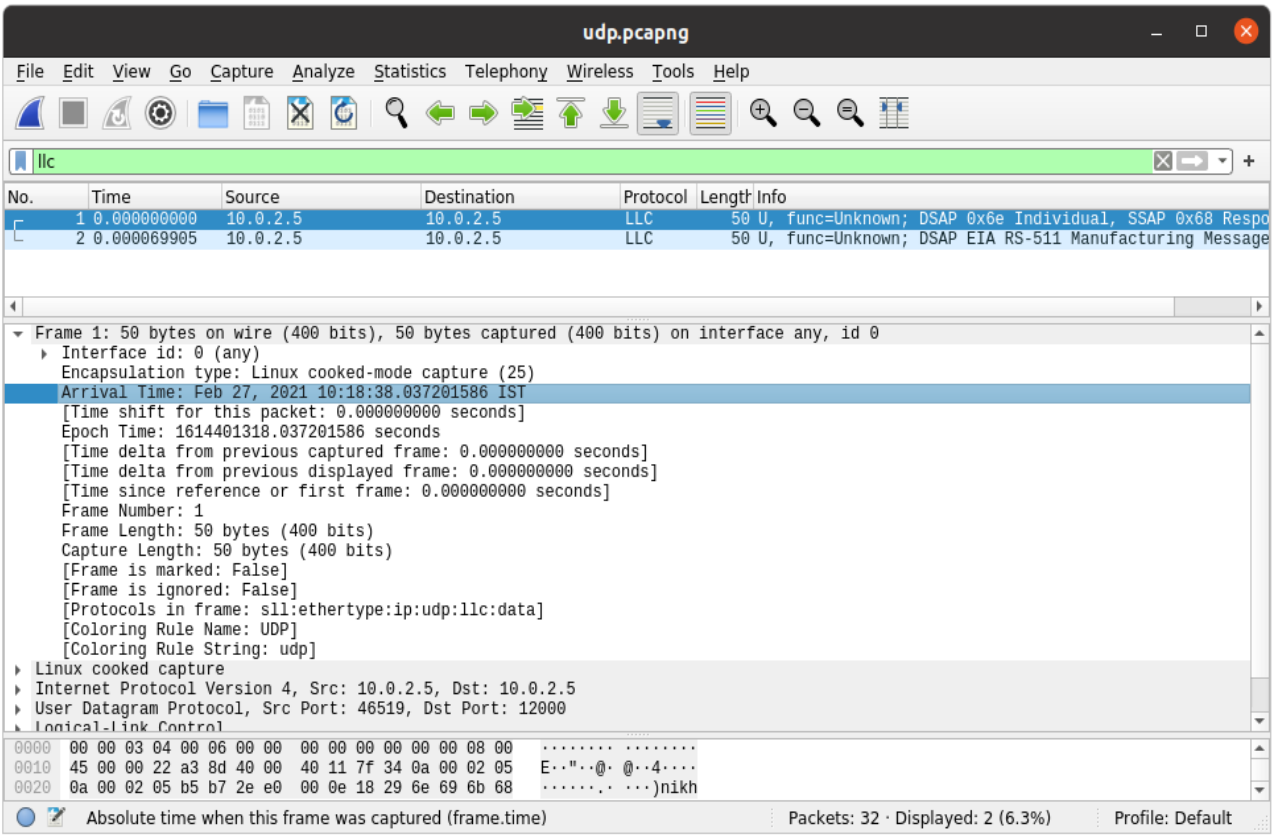
```
nikhil@nikhil-VirtualBox:~/Downloads$ python UDPserver.py
The server is ready to receive
```

UDPClient program compiled and its corresponding output



```
nikhil@nikhil-VirtualBox:~/Downloads$ python2 UDPclient.py
Input lowercase sentence:nikhil
NIKHIL
nikhil@nikhil-VirtualBox:~/Downloads$
```

The wireshark packet capture of UDP packets


udp.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
llc

```
No.   Time          Source      Destination  Protocol Length Info
    1 0.000000000   10.0.2.5    10.0.2.5     LLC          50 U, func=Unknown; DSAP 0x6e Individual, SSAP 0x68 Respo
    2 0.000069905   10.0.2.5    10.0.2.5     LLC          50 U, func=Unknown; DSAP EIA RS-511 Manufacturing Message
```

```
Frame 1: 50 bytes on wire (400 bits), 50 bytes captured (400 bits) on interface any, id 0
   Interface id: 0 (any)
   Encapsulation type: Linux cooked-mode capture (25)
   Arrival Time: Feb 27, 2021 10:18:38.037201586 IST
   [Time shift for this packet: 0.000000000 seconds]
   Epoch Time: 1614401318.037201586 seconds
   [Time delta from previous captured frame: 0.000000000 seconds]
   [Time delta from previous displayed frame: 0.000000000 seconds]
   [Time since reference or first frame: 0.000000000 seconds]
   Frame Number: 1
   Frame Length: 50 bytes (400 bits)
   Capture Length: 50 bytes (400 bits)
   [Frame is marked: False]
   [Frame is ignored: False]
   [Protocols in frame: sll:ethertype:ip:udp:llc:data]
   [Coloring Rule Name: UDP]
   [Coloring Rule String: udp]
Linux cooked capture
Internet Protocol Version 4, Src: 10.0.2.5, Dst: 10.0.2.5
User Datagram Protocol, Src Port: 46519, Dst Port: 12000
Logical-Link Control
```

```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ········ ········
0010  45 00 00 22 a3 8d 40 00  40 11 7f 34 0a 00 02 05   E··"··@· @··4····
0020  0a 00 02 05 b5 b7 2e e0  00 0e 18 29 6e 69 6b 68   ·······.· ···)nikh
```

Absolute time when this frame was captured (frame.time)   Packets: 32 · Displayed: 2 (6.3%)   Profile: Default

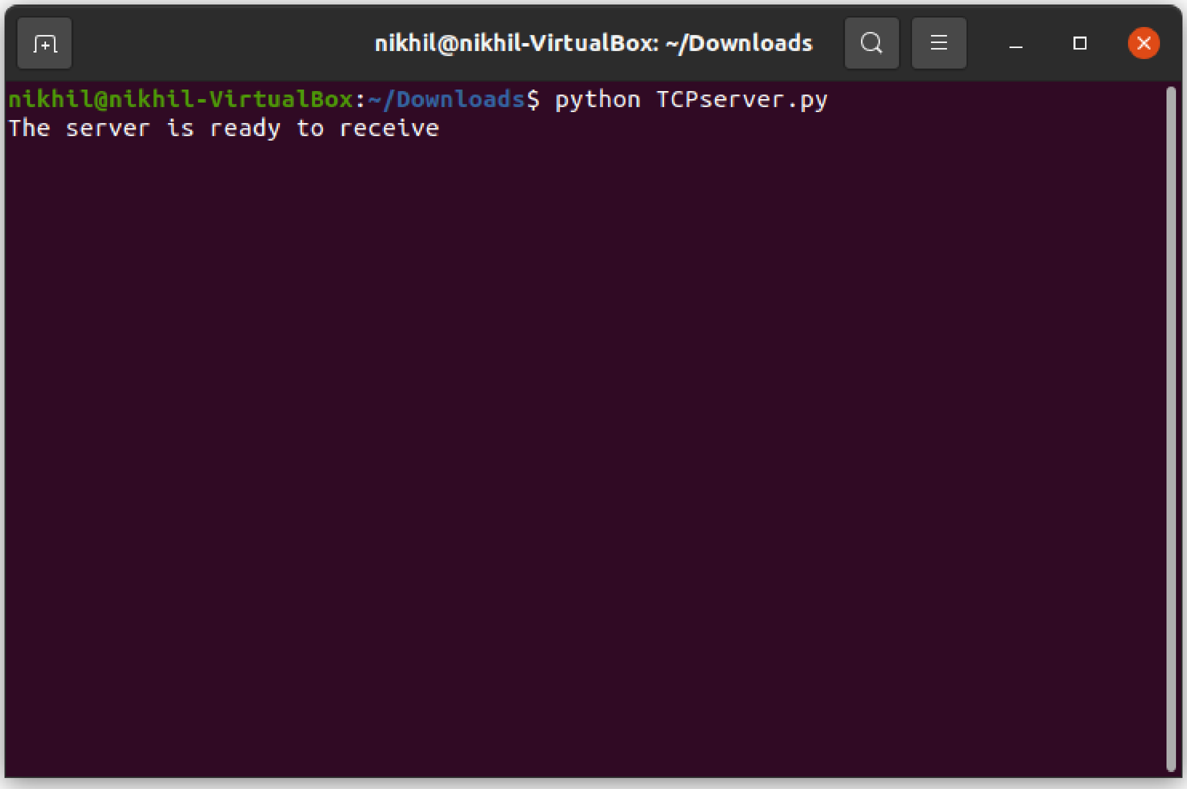
TCPClient.py

```
from socket import *
serverName = "10.0.2.5"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,serverPort))
sentence = raw_input("Input lowercase sentence:")
clientSocket.sendto(sentence.encode(),(serverName, serverPort))
modifiedSentence = clientSocket.recv(1024)
print("From Server:", modifiedSentence)
clientSocket.close()
```

TCPServer.py

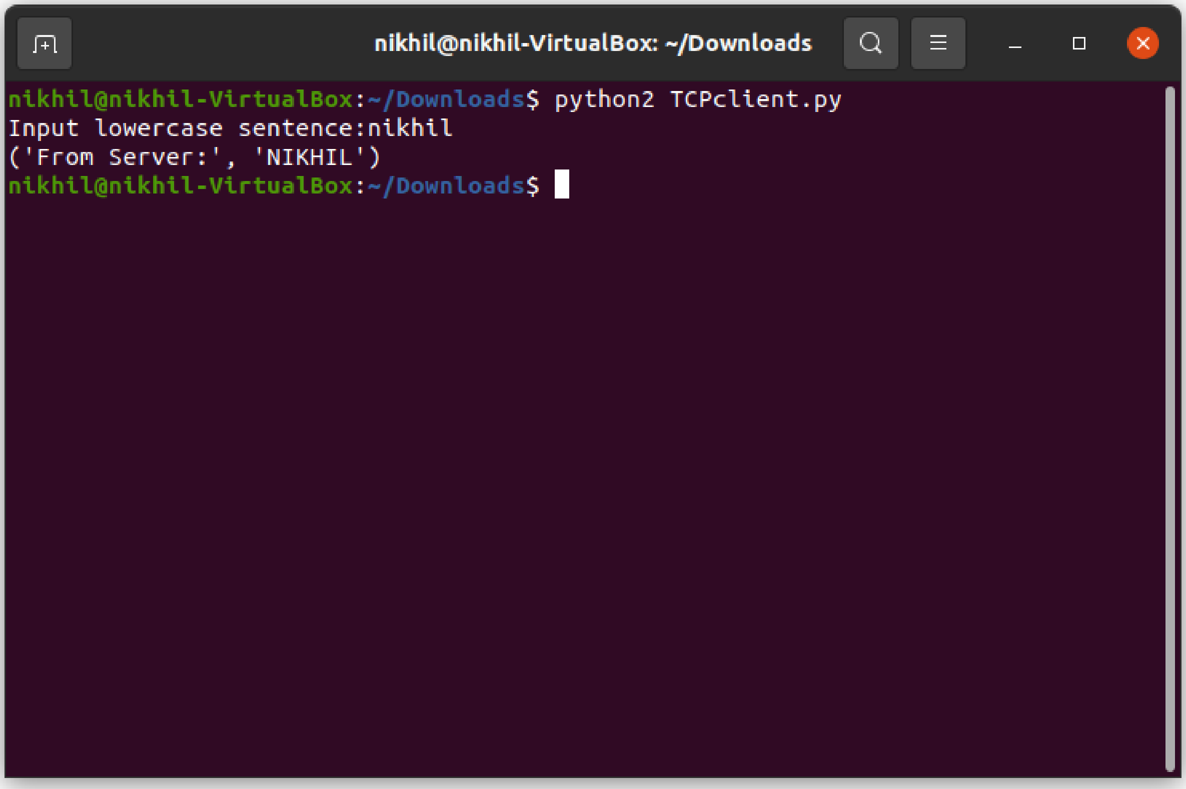
```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind(("10.0.2.5",serverPort))
serverSocket.listen(1)
print("The server is ready to receive")
while 1:
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024)
    capitalizedSentence = sentence.upper()
    connectionSocket.send(capitalizedSentence)
    connectionSocket.close()
```

TCP server in running



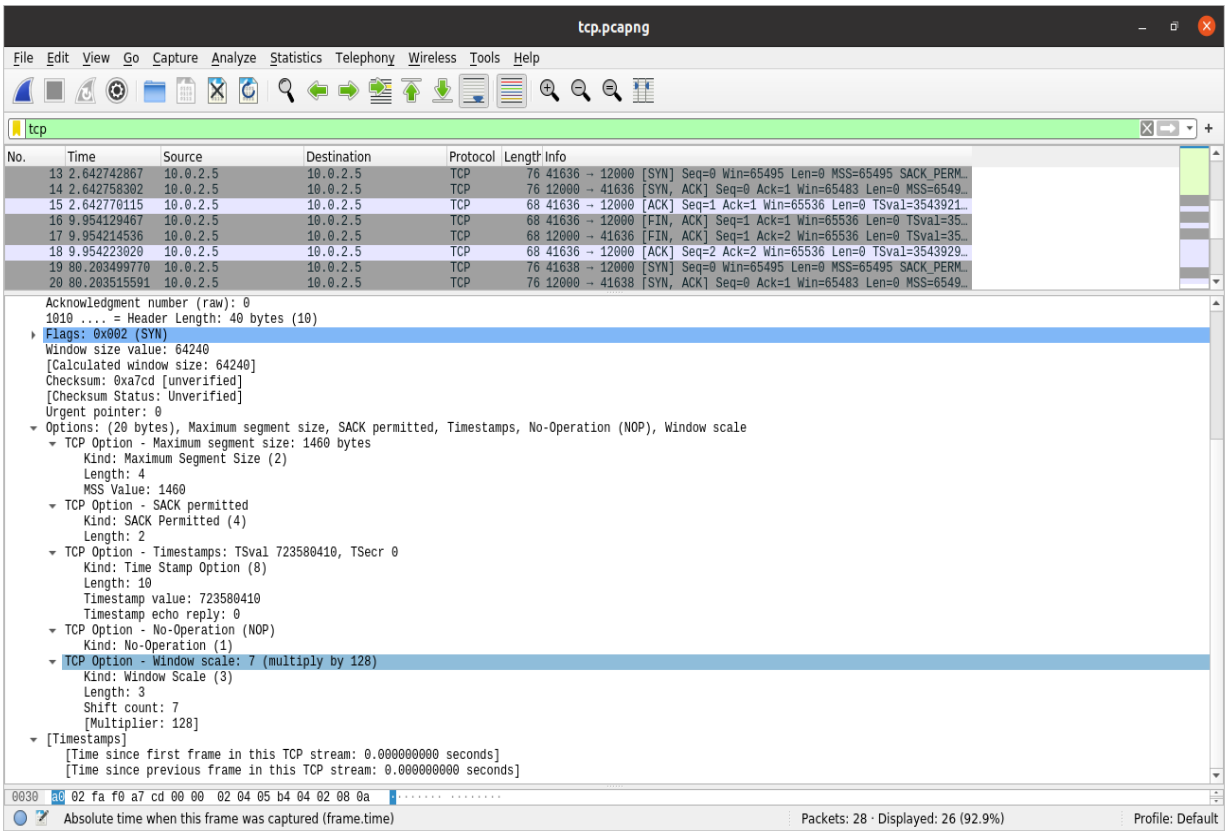
```
nikhil@nikhil-VirtualBox:~/Downloads$ python TCPserver.py
The server is ready to receive
```

TCP client program compiled and its corresponding output



```
nikhil@nikhil-VirtualBox:~/Downloads$ python2 TCPclient.py
Input lowercase sentence:nikhil
('From Server:', 'NIKHIL')
nikhil@nikhil-VirtualBox:~/Downloads$
```

The wireshark capture of TCP packets



tcp

```
No.  Time          Source      Destination  Protocol Length Info
  13 2.642742867   10.0.2.5    10.0.2.5     TCP       76 41636 → 12000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM…
  14 2.642758302   10.0.2.5    10.0.2.5     TCP       76 12000 → 41636 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=6549…
  15 2.642770115   10.0.2.5    10.0.2.5     TCP       68 41636 → 12000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3543921…
  16 9.954129467   10.0.2.5    10.0.2.5     TCP       68 41636 → 12000 [FIN, ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=35…
  17 9.954214536   10.0.2.5    10.0.2.5     TCP       68 12000 → 41636 [FIN, ACK] Seq=1 Ack=2 Win=65536 Len=0 TSval=35…
  18 9.954223020   10.0.2.5    10.0.2.5     TCP       68 41636 → 12000 [ACK] Seq=2 Ack=2 Win=65536 Len=0 TSval=3543929…
  19 80.203499770  10.0.2.5    10.0.2.5     TCP       76 41638 → 12000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM…
  20 80.203515591  10.0.2.5    10.0.2.5     TCP       76 12000 → 41638 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=6549…
```

```
    Acknowledgment number (raw): 0
    1010 .... = Header Length: 40 bytes (10)
  ▸ Flags: 0x002 (SYN)
    Window size value: 64240
    [Calculated window size: 64240]
    Checksum: 0xa7cd [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▾ Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
    ▾ TCP Option - Maximum segment size: 1460 bytes
        Kind: Maximum Segment Size (2)
        Length: 4
        MSS Value: 1460
    ▾ TCP Option - SACK permitted
        Kind: SACK Permitted (4)
        Length: 2
    ▾ TCP Option - Timestamps: TSval 723580410, TSecr 0
        Kind: Time Stamp Option (8)
        Length: 10
        Timestamp value: 723580410
        Timestamp echo reply: 0
    ▾ TCP Option - No-Operation (NOP)
        Kind: No-Operation (1)
    ▾ TCP Option - Window scale: 7 (multiply by 128)
        Kind: Window Scale (3)
        Length: 3
        Shift count: 7
        [Multiplier: 128]
  ▾ [Timestamps]
      [Time since first frame in this TCP stream: 0.000000000 seconds]
      [Time since previous frame in this TCP stream: 0.000000000 seconds]
```

```
0030  a0 02 fa f0 a7 cd 00 00  02 04 05 b4 04 02 08 0a   ........ ........
```

Absolute time when this frame was captured (frame.time)   Packets: 28 · Displayed: 26 (92.9%)   Profile: Default

PROBLEMS:

1) Suppose you run TCPClient before you run TCP server . What happens? Why?

Ans: The TCPClient will not respond if it will be executed first without running the TCP server. This happens because connection will not be established between TCP client and server without Running the server first.

2) Suppose you run UDPClient before you run UDP server . What happens? Why?

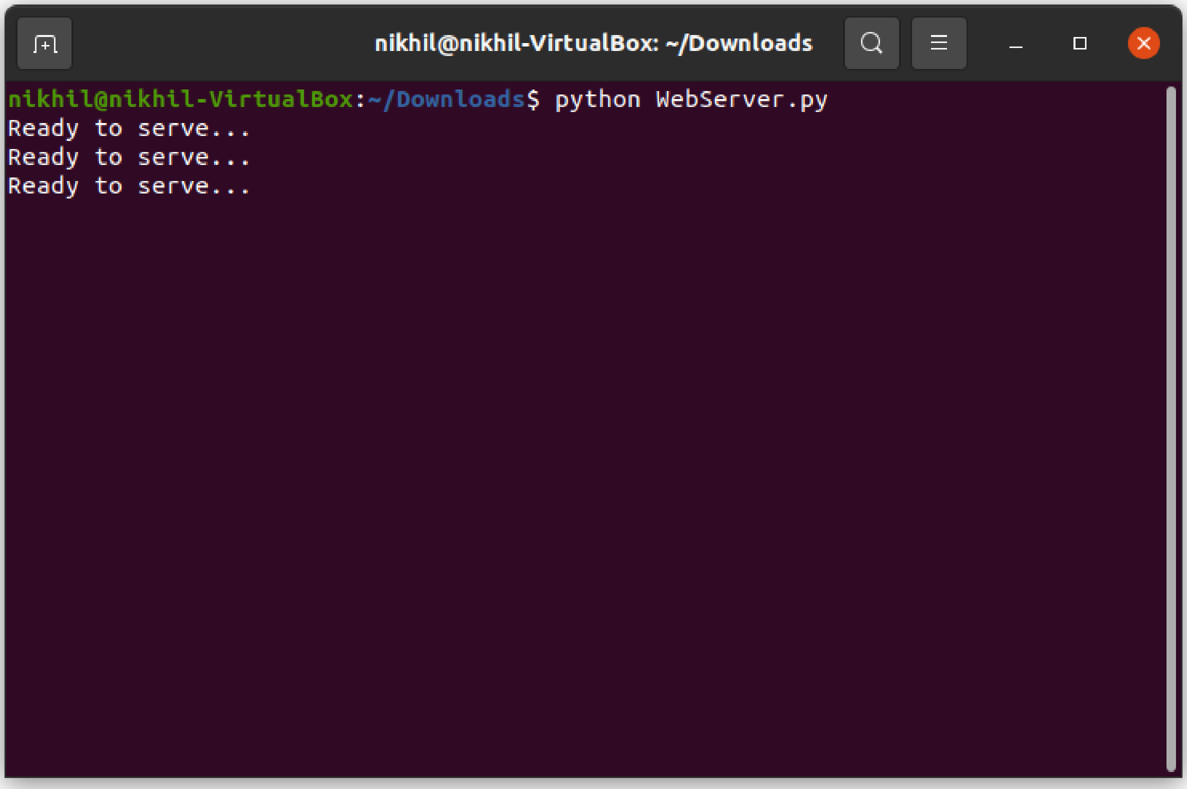
Ans: The UDP Client will respond properly before running UDP server. This happens because UDP is an unreliable protocol unlike the TCP protocol.

3) What happens if you use different port numbers for the client and server sides?

Ans: If different port numbers are used the TCP will not establish connection and errors may occur in Communication.

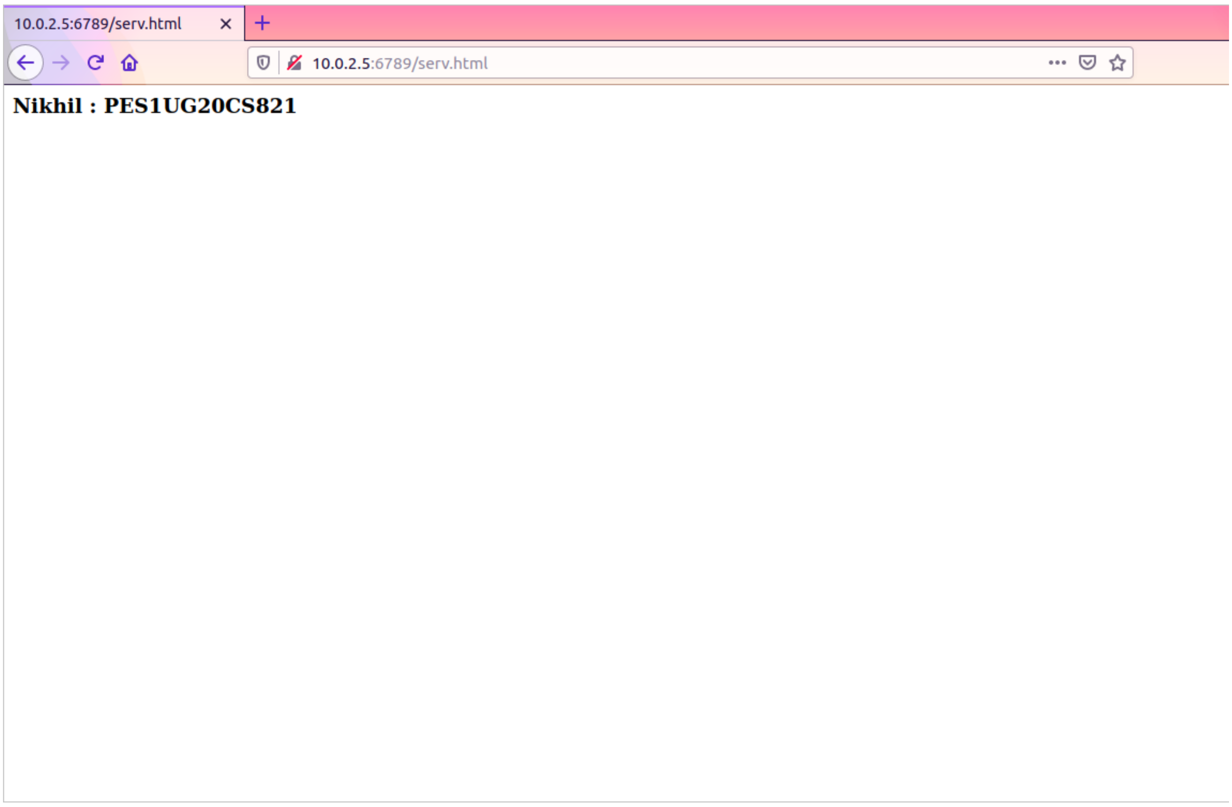
Task 2: Web Server

The web server program in running



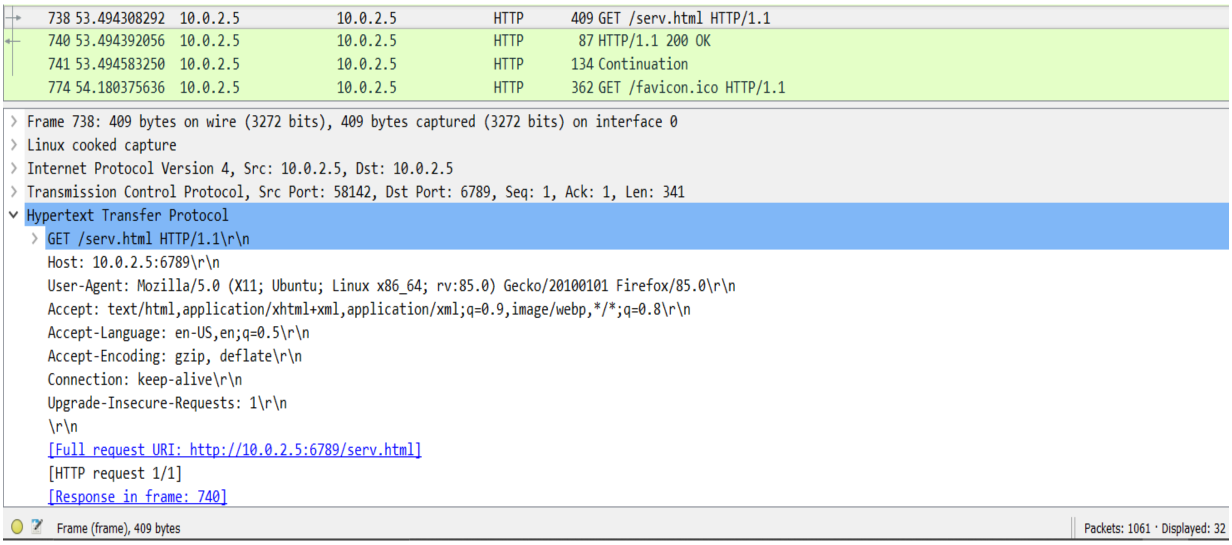
```
nikhil@nikhil-VirtualBox:~/Downloads$ python WebServer.py
Ready to serve...
Ready to serve...
Ready to serve...
```

The webpage with content displayed.



Nikhil : PES1UG20CS821

The wireshark packet capture of the webpage.

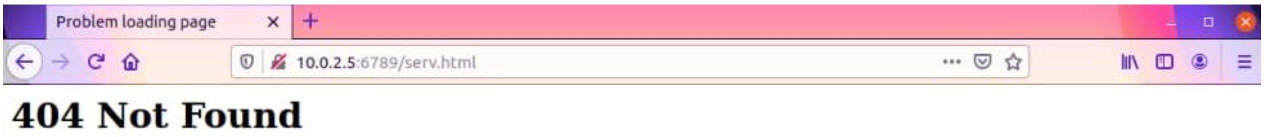


```
738 53.494308292  10.0.2.5          10.0.2.5          HTTP      409 GET /serv.html HTTP/1.1
740 53.494392056  10.0.2.5          10.0.2.5          HTTP       87 HTTP/1.1 200 OK
741 53.494583250  10.0.2.5          10.0.2.5          HTTP      134 Continuation
774 54.180375636  10.0.2.5          10.0.2.5          HTTP      362 GET /favicon.ico HTTP/1.1

> Frame 738: 409 bytes on wire (3272 bits), 409 bytes captured (3272 bits) on interface 0
> Linux cooked capture
> Internet Protocol Version 4, Src: 10.0.2.5, Dst: 10.0.2.5
> Transmission Control Protocol, Src Port: 58142, Dst Port: 6789, Seq: 1, Ack: 1, Len: 341
v Hypertext Transfer Protocol
  > GET /serv.html HTTP/1.1\r\n
    Host: 10.0.2.5:6789\r\n
    User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:85.0) Gecko/20100101 Firefox/85.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://10.0.2.5:6789/serv.html]
    [HTTP request 1/1]
    [Response in frame: 740]

Frame (frame), 409 bytes                                        Packets: 1061 · Displayed: 32
```

The browser responding to a file not present in the system



404 Not Found