

Name: Preyash

Date: 03-02-2022

Registration Number: 20BPS1022

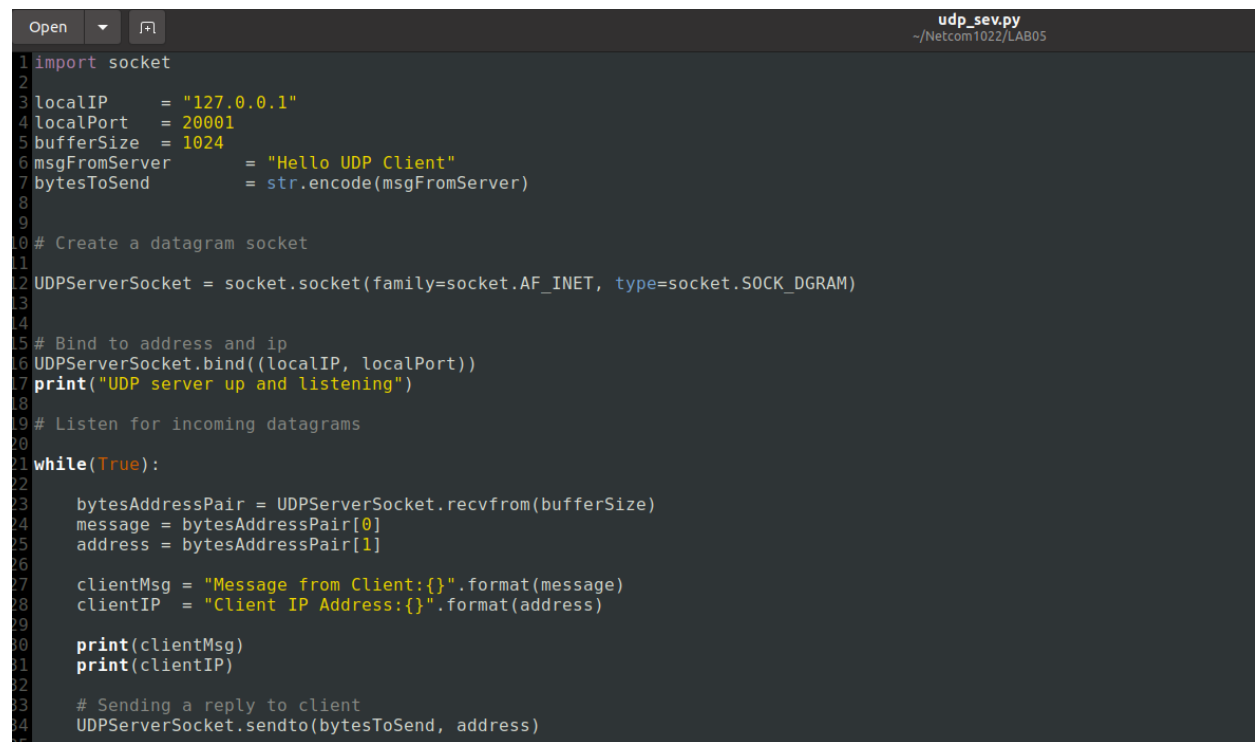
LAB-05

UDP Socket Programming

AIM: To get familiar with UDP socket programming and to build a key server that serves clients with a client-supplied encoded key.

Basic program: Basic program to get familiar with UDP concepts.

Server-side code:



```
udp_serv.py
~/Netcom1022/LAB05

1 import socket
2
3 localIP    = "127.0.0.1"
4 localPort  = 20001
5 bufferSize = 1024
6 msgFromServer = "Hello UDP Client"
7 bytesToSend = str.encode(msgFromServer)
8
9
10 # Create a datagram socket
11
12 UDPServerSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)
13
14
15 # Bind to address and ip
16 UDPServerSocket.bind((localIP, localPort))
17 print("UDP server up and listening")
18
19 # Listen for incoming datagrams
20
21 while(True):
22
23     bytesAddressPair = UDPServerSocket.recvfrom(bufferSize)
24     message = bytesAddressPair[0]
25     address = bytesAddressPair[1]
26
27     clientMsg = "Message from Client:{}".format(message)
28     clientIP  = "Client IP Address:{}".format(address)
29
30     print(clientMsg)
31     print(clientIP)
32
33     # Sending a reply to client
34     UDPServerSocket.sendto(bytesToSend, address)
35
```

Client-side code:

```
Open  udp_cli.py
~/Netcom1022/LAB05
1 import socket
2
3 msgFromClient      = "Hello UDP Server"
4 bytesToSend        = str.encode(msgFromClient)
5 serverAddressPort  = ("127.0.0.1", 20001)
6 bufferSize         = 1024
7
8
9 # Create a UDP socket at client side
10 UDPClientSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)
11
12 # Send to server using created UDP socket
13 UDPClientSocket.sendto(bytesToSend, serverAddressPort)
14
15 msgFromServer = UDPClientSocket.recvfrom(bufferSize)
16
17 msg = "Message from Server {}".format(msgFromServer[0])
18
19 print(msg)
20
21
22
```

Server-side output:

```
preyash-20bps1022@Preyash-20BPS1...  ×  preyash-20bps1022@Preyash-20BPS1...  ×
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB05$ python3 udp_sev.py
UDP server up and listening
Message from Client:b'Hello UDP Server'
Client IP Address:('127.0.0.1', 33391)
█
```

Client-side output:

```
preyash-20bps1022@Preyash-20BPS1...  ×  preyash-20bps1022@Preyash-20BPS1...  ×
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB05$ python3 udp_cli.py
Message from Server b'Hello UDP Client'
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB05$ █
```

TASK:

Send the following statement from the client and do the message encoding on the server side and display it on the client side.

“Please Switch Off the Lights and Fan When Not in Use” replace the vowels with their next vowel for example “a” with “e”, “e” with “i” and so on. Similarly replace the consonants with the next consonant in order. Retain the Capital letters even after encoding.

ALGORITHM

Server-side Algorithm:

- ✓ Connect to a port
- ✓ Wait for the client to connect.
- ✓ Once connected, receive the client's key and encode it before returning the encoded key.
- ✓ Disconnect

Client-Side Algorithm:

- ✓ Connect to port
- ✓ Send key
- ✓ Wait for response and receive it
- ✓ Print the response

Server Program Source Code:

Code window:

```
enDec_sev.py
~/Netcom1022/LAB05

Open [v] [icon]

1 import socket
2 localIP = "127.0.0.1"
3 localPort = 20001
4 bufferSize = 1024
5 # Creating a datagram socket
6 UDPServerSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)
7 # Bind to address and ip
8 UDPServerSocket.bind((localIP, localPort))
9 print("UDP server up and listening")
10 # Listen for incoming datagrams
11 while(True):
12     bytesAddressPair = UDPServerSocket.recvfrom(bufferSize)
13     message = bytesAddressPair[0]
14     address = bytesAddressPair[1]
15     clientMsg = str(message)
16     clientMsg=clientMsg[2:]
17     clientIP = "Client IP Address:{}".format(address)
18     print(clientMsg)
19     print(clientIP)
20     vow = 'a e i o u'.split()
21     temp = dict(zip(vow, vow[1:] + [vow[0]]))
22     res = "".join([temp.get(ele, ele) for ele in clientMsg])
23     cons = 'b c d f g h j k l m n p q r s t v w x y z'.split()
24     temp = dict(zip(cons, cons[1:] + [cons[0]]))
25     fres = "".join([temp.get(ele, ele) for ele in res])
26     msgFromServer = fres
27     bytesToSend = str.encode(msgFromServer)
28     UDPServerSocket.sendto(bytesToSend, address)
29
```

Code:

```
import socket

localIP = "127.0.0.1"

localPort = 20001

bufferSize = 1024

# Creating a datagram socket

UDPServerSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)

# Bind to address and ip

UDPServerSocket.bind((localIP, localPort))

print("UDP server up and listening")

# Listen for incoming datagrams

while(True):

    bytesAddressPair = UDPServerSocket.recvfrom(bufferSize)

    message = bytesAddressPair[0]
```

```

address = bytesAddressPair[1]

clientMsg = str(message)

clientMsg=clientMsg[2:]

clientIP = "Client IP Address:{}".format(address)

print(clientMsg)

print(clientIP)

vow = 'a e i o u'.split()

temp = dict(zip(vow, vow[1:] + [vow[0]]))

res = "".join([temp.get(ele, ele) for ele in clientMsg])

cons = 'b c d f g h j k l m n p q r s t v w x y z'.split()

temp = dict(zip(cons, cons[1:] + [cons[0]]))

fres = "".join([temp.get(ele, ele) for ele in res])

msgFromServer = fres

bytesToSend = str.encode(msgFromServer)

UDPServerSocket.sendto(bytesToSend, address)

```

Output:

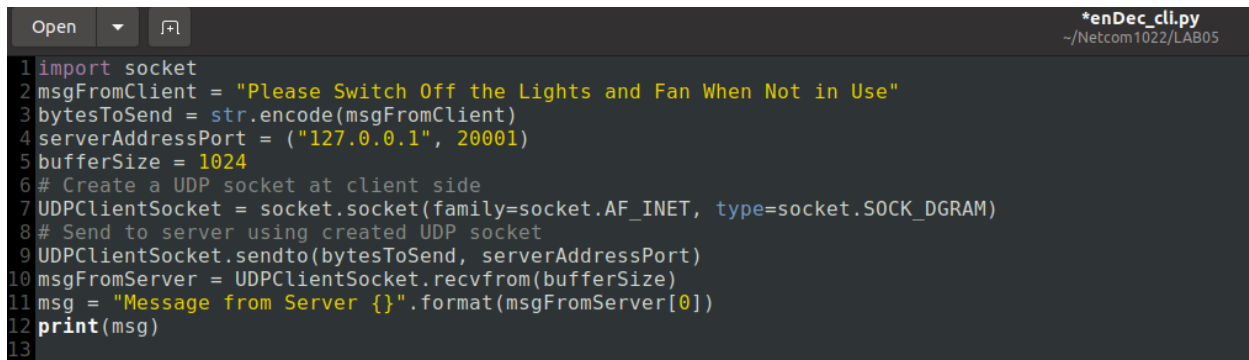
```

preyash-20bps1022@Preyash-20BPS1... x preyash-20bps1022@Preyash-20BPS1... x
preyash-20bps1022@Preyash-20BPS1022:~$ cd Netcom1022/
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022$ cd LAB05
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB05$ ls
a.out  enDec_cli.py  enDec_sev.py  udp_cli.c  udp_cli.py  udp_sev.c  udp_sev.py
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB05$ python3 enDec_sev.py
UDP server up and listening
Please Switch Off the Lights and Fan When Not in Use'
Client IP Address:('127.0.0.1', 34912)

```

Client Program Source Code:

Code window:



```
Open [v] [icon] *enDec_cli.py
~/Netcom1022/LAB05
1 import socket
2 msgFromClient = "Please Switch Off the Lights and Fan When Not in Use"
3 bytesToSend = str.encode(msgFromClient)
4 serverAddressPort = ("127.0.0.1", 20001)
5 bufferSize = 1024
6 # Create a UDP socket at client side
7 UDPClientSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)
8 # Send to server using created UDP socket
9 UDPClientSocket.sendto(bytesToSend, serverAddressPort)
10 msgFromServer = UDPClientSocket.recvfrom(bufferSize)
11 msg = "Message from Server {}".format(msgFromServer[0])
12 print(msg)
13
```

Code:

```
import socket
```

```
msgFromClient = "Please Switch Off the Lights and Fan When Not in Use"
```

```
bytesToSend = str.encode(msgFromClient)
```

```
serverAddressPort = ("127.0.0.1", 20001)
```

```
bufferSize = 1024
```

```
# Create a UDP socket at client side
```

```
UDPClientSocket = socket.socket(family=socket.AF_INET, type=socket.SOCK_DGRAM)
```

```
# Send to server using created UDP socket
```

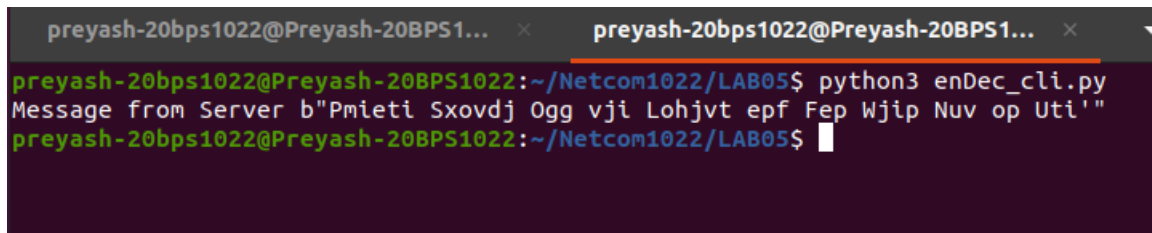
```
UDPClientSocket.sendto(bytesToSend, serverAddressPort)
```

```
msgFromServer = UDPClientSocket.recvfrom(bufferSize)
```

```
msg = "Message from Server {}".format(msgFromServer[0])
```

```
print(msg)
```

Output:



```
preyash-20bps1022@Preyash-20BPS1... x preyash-20bps1022@Preyash-20BPS1... x
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB05$ python3 enDec_cli.py
Message from Server b"Pmieti Sxovdj Ogg vji Lohjvt epf Fep Wjip Nuv op Uti'"
preyash-20bps1022@Preyash-20BPS1022:~/Netcom1022/LAB05$
```

Result: We successfully created a program to perform the required output.