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_sev.py
~/Netcom1022/LAB05

```python
import socket

localIP     = "127.0.0.1"
localPort   = 20001
bufferSize  = 1024
msgFromServer       = "Hello UDP Client"
bytesToSend         = str.encode(msgFromServer)


# Create 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 = "Message from Client:{}".format(message)
    clientIP  = "Client IP Address:{}".format(address)

    print(clientMsg)
    print(clientIP)

    # Sending a reply to client
    UDPServerSocket.sendto(bytesToSend, address)
```

Client-side code:

```
Open                                                    udp_cli.py
                                                        ~/Netcom1022/LAB05
import socket

msgFromClient       = "Hello UDP Server"
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)
```

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:

Open

enDec_sev.py
~/Netcom1022/LAB05

```python
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)
```

Code:

```python
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...   preyash-20bps1022@Preyash-20BPS1...
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                                                    *enDec_cli.py
                                                        ~/Netcom1022/LAB05
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)
```

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...   preyash-20bps1022@Preyash-20BPS1...
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.