Socket Programming : UDP

Aim

To implement Client-Server communication using Socket Programming and UDP as transport layer protocol.

Theory

UDP (User Datagram Protocol) is an alternative communications protocol to Transmission Control Protocol (**TCP**) used primarily for establishing low latency and loss-tolerating connections between applications on the internet. **UDP** enables process-to-process communication. **UDP** sends messages, called datagrams, and is considered a best-effort mode of communications. It is considered a connectionless protocol because it doesn't require a virtual circuit to be established before any data transfer occurs.

Server & Client - Since the **UDP** is a connectionless protocol, they do not require a connection to get established prior to data transmission or reception. Hence data can be sent between them directly.

Code

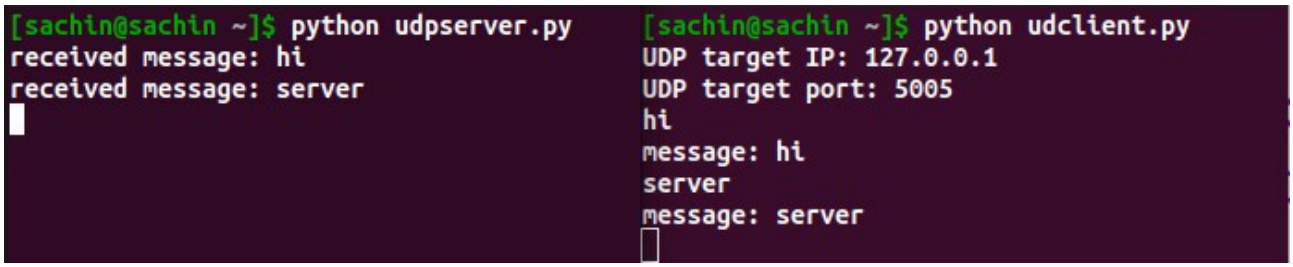
udpserver.cpp

```
import socket
UDP_IP = "127.0.0.1"
UDP_PORT = 5005
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # UDP
sock.bind((UDP_IP, UDP_PORT))
while True:
  data, addr = sock.recvfrom(1024) # buffer size is 1024 bytes
  print "received message:", data
```

udpclient.py

```
import socket
UDP_IP = "127.0.0.1"
UDP_PORT = 5005
print "UDP target IP:", UDP_IP
print "UDP target port:", UDP_PORT
while True:
 MESSAGE= raw_input()
 print "message:", MESSAGE
 sock = socket.socket(socket.AF_INET, # Internet
 socket.SOCK_DGRAM) # UDP
 sock.sendto(MESSAGE, (UDP_IP, UDP_PORT))
```

Output


[sachin@sachin ~]$ python udpserver.py
received message: hi
received message: server
[sachin@sachin ~]$ python udclient.py
UDP target IP: 127.0.0.1
UDP target port: 5005
hi
message: hi
server
message: server