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

Amrith M

March 6, 2018

1 About

A client/server application model typically is viewed as a remotely located, high powered computing device that stores a large quantity of data with business logic to access them in a network. The user interface is handled by the client software on a relatively cheap machine. This idea is not distinct because any machine serving the request can potentially be called a server. A TCP connection is like telephone calls where we dial up to connect to the person with whom we want to communicate. The connection remains intact throughout the communication process even if we are not sending any signals. UDP, on the other hand, is more like mail carried through the postal service that assumes they may arrive out of sequence, lost in transit, or duplicate datagrams were detected at the receiving end.

2 Server.py

```
import socket

sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
sock.bind(('127.0.0.1',3000))

while True:
        data,address = sock.recvfrom(4096)
        print(data)
        if data:
                sent = sock.sendto(data,address)
                print("Data Sent Back To ",address)
```

3 Client.py

```
import socket

sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
server = ('127.0.0.1',3000)
n = 0
try:

        while n == 0:
                msg = raw_input("Enter Data: ")
                sent = sock.sendto(msg,server)
                print("Waiting for resp")
                data,server = sock.recvfrom(4096)
                print("Received Data: "+data)
                n=input("1 to exit or 0 to continue")
                if(n == 1):
                        break
finally:
        sock.close()
```