Afaan Ansari
2019230064
CEL 51, DCCN, Monsoon 2020
Lab 8: Client and Server Configuration

Socket programming in python

(reff : https://www.geeksforgeeks.org/socket-programming-python/)

Python provides two levels of access to network services. At a low level, you can access the basic socket support in the underlying operating system, which allows you to implement clients and servers for both connection-oriented and connectionless protocols.

Python also has libraries that provide higher-level access to specific application-level network protocols, such as FTP, HTTP, and so on.

This chapter gives you an understanding on the most famous concept in Networking - Socket Programming.

Sockets are the endpoints of a bidirectional communications channel. Sockets may communicate within a process, between processes on the same machine, or between processes on different continents.

Sockets may be implemented over a number of different channel types: Unix domain sockets, TCP, UDP, and so on. The socket library provides specific classes for handling the common transports as well as a generic interface for handling the rest.

Client

To write Internet servers, we use the socket function available in socket module to create a socket object. A socket object is then used to call other functions to setup a socket server

Now call connet(hostname, port) function to specify a port for your service on the given host.

Server

Let us write a very simple client program which opens a connection to a given port 8000 and given host. This is very simple to create a socket client using Python's socket module function.

The socket.connect(hosname, port) opens a TCP connection to hostname on the port. Once you have a socket open, you can read from it like any IO object. When done, remember to close it, as you would close a file.

Code for server

- **Server**

```
import socket
 s = socket.socket()
print("Socket successfully created")
port = 12345

s.bind(('', port))
print ("socket binded to %s" %(port))

s.listen(5)
print ("socket is listening")

while True:
  c, addr = s.accept()
  print('Got connection from', addr)
  c.sendall('Thank you for connecting')
  c.close()
```

- **Client**

```
import socket

s = socket.socket()
port = 12345

s.connect(('127.0.0.1', port))
print(s.recv(1024))
s.close()
```

- **Output**

```
n
affanansari@affans-MacBook-Air Downloads % python3 server.py
Socket successfully created
socket binded to 12345
socket is listening
Got connection from ('127.0.0.1', 53235)
```

```
socket.gaierror: [Errno 8] nodename nor servname provided, or not know
n
affanansari@affans-MacBook-Air Downloads % python3 client.py
b'Thank you for connecting'
affanansari@affans-MacBook-Air Downloads %
```

Conclusion :

After completing the above experiment, I have understood that how a client and server works and how do they communicate.