Assignment :- D-12

Title:- Client -Server Application

Aim:- Write a server application to be deployed on Raspberry-Pi /Beagle board. Write client applications to get services from the server application..

Hardware Requirement :-

1) Raspberry Pi Board module
2) DHT-11 Sensor

Software Requirement:-

1) Raspbian OS (IDLE)

Theory:-

Sockets:-

1) Sockets are the endpoints of a bidirectional communications channel.
2) Sockets may communicate within a process, between processes on the same machine, or between processes on different continents.
3) Sockets may be implemented over a number of different channel types: Unix domain sockets, TCP, UDP, and so on.
4) The socket library provides specific classes for handling the common transports as well as a generic interface for handling the rest.
5) To create a socket, you must use the socket.socket() function in socket module, which has the general syntax:

   s = socket.socket (socket_family,socket_type, protocol=0)

   Where,

   socket_family: This is either AF_UNIX or AF_INET.

   socket_type: This is either SOCK_STREAM or SOCK_DGRAM.

   protocol: This is usually left out, defaulting to 0.

6) Once you have socket object, then you can use required functions to create your client or server program.

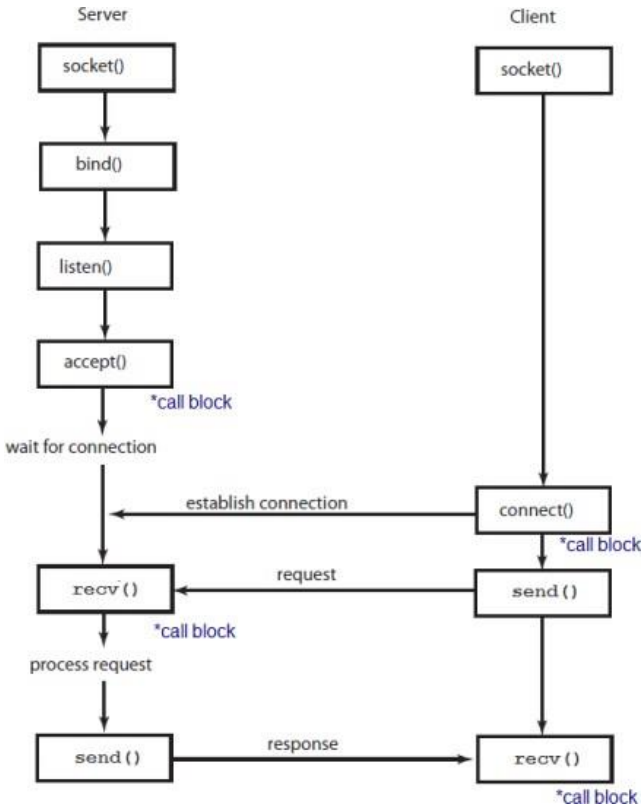
Server Socket Methods :-

1) s.bind() This method binds address (hostname, port number pair) to socket.
2) s.listen() This method sets up and start TCP listener.
3) s.accept() This passively accept TCP client connection, waiting until connection arrives (blocking).
4) s.connect() This method actively initiates TCP server connection.

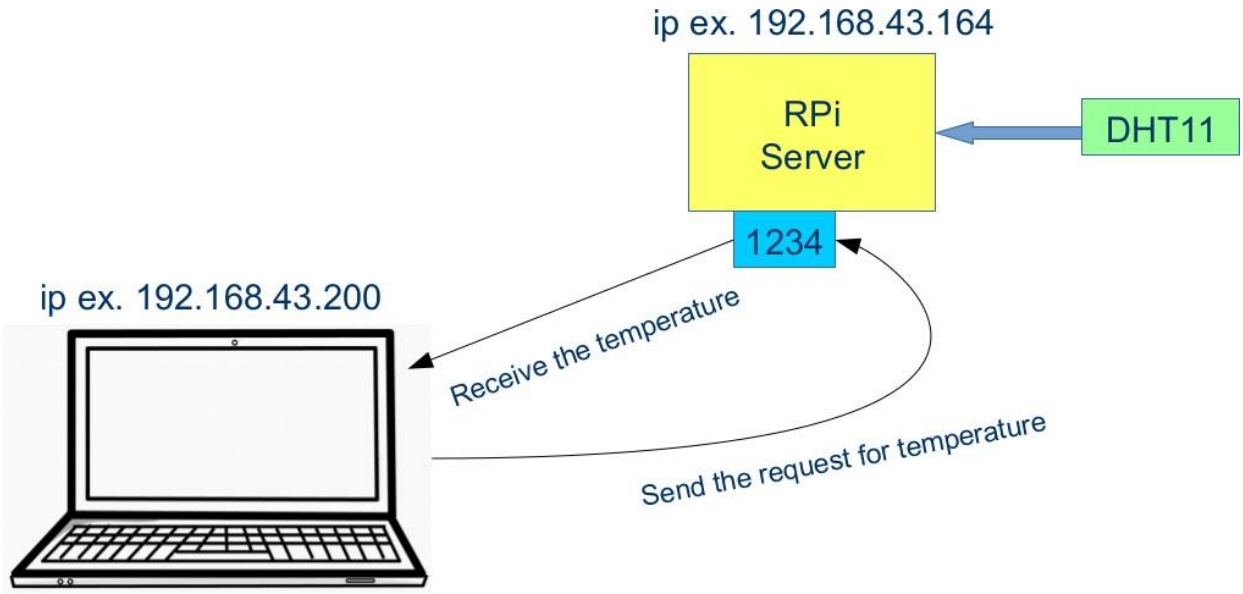
General Socket Methods:-

1) s.recv() This method receives TCP message
2) s.send() This method transmits TCP message
3) s.recvfrom() This method receives UDP message
4) s.sendto() This method transmits UDP message
5) s.close() This method closes socket
6) socket.gethostname() Returns the hostname.

How the communication is taken place?



Raspberry Pi Server Application:-



Program for Temperature Server:-

```
import socket
import Adafruit_DHT

s = socket.socket()

s.bind(("192.168.43.164",1234))
s.listen(5)
while True:
    print "Waiting for client connection..."
    c, addr = s.accept()

    hum, temp = Adafruit_DHT.read_retry(11, 4)
    print 'Got connection from', addr
    c.send(str(temp))
    c.close()
```

Program for Temperature Server:-

```
import socket

s = socket.socket()
# Create a socket object

s.connect(("192.168.43.164", 1234))
print "The temperature is: " s.recv(1024)
s.close()
```

Temperature Server UDP:-

```
import socket, time
import Adafruit_DHT

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
while True:
    hum, temp = Adafruit_DHT.read_retry(11, 4)
    print "Broadcasting...", temp
    sock.sendto(str(temp), ('192.168.43.164', 1234))
    time.sleep(1)
```

Temperature Client UDP:-

```
import socket

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.bind(('192.168.43.164', 1234))

while True:
    data, addr = sock.recvfrom(100)
    print "The temperature is:", data, "from", addr
```

Conclusion:

We have successfully developed a server application to be deployed on Raspberry-Pi /Beagle board and client applications to get services from the server application..