

**Name : Md. Sohag Hossain**

**ID : IT-17060**

**Lab report no : 03**

**Name of the lab report:** Socket programming

**Objectives:**

- ✓ learn server and client
- ✓ learn socket programming
- ✓ learn TCP,UDP.

**Theory:**

Python Socket programming is used for communication between the applications running on different JRE. Python Socket programming can be connection-oriented or connection-less.

Socket and Server Socket classes are used for connection-oriented socket programming and Datagram Socket and Datagram Packet classes are used for connection-less socket programming.

The client in socket programming must know two information:

1. IP Address of Server, and
2. Port number.

The primary socket API functions and methods in this module are:

- `socket()`
- `bind()`
- `listen()`
- `accept()`
- `connect()`
- `connect_ex()`
- `send()`
- `recv()`
- `close()`

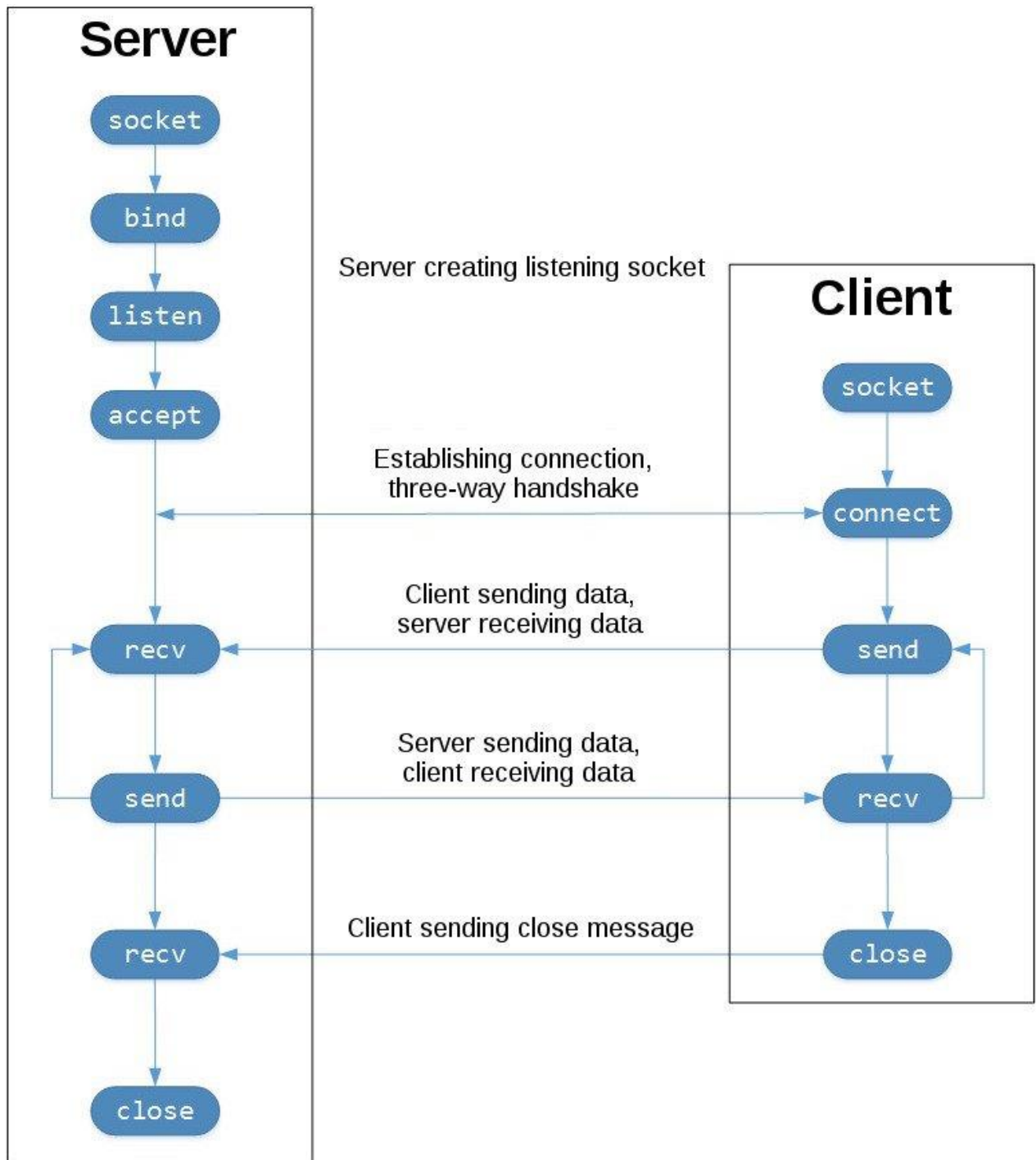


Fig: TCP Socket Flow

## Import Socket Library:

```
1 . import socket
```

## Build Socket Objects:

```
1 . sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

output:

```
In [2]: import socket
...: sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
...: print(sock)
<socket.socket fd=2088, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM,
proto=0>
```

## Server Connection Code:

```
import socket
import sys
try:
    s = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
    print ("Socket successfully created")
except socket.error as err:
    print ("socket creation failed with error %s"
%(err) )
port = 80
try:
    host_ip = socket.gethostbyname('www.google.com')
except socket.gaierror:
    print("there was an error resolving the
host",sys.exit())
s.connect((host_ip, port))
print ("the socket has successfully connected to google
\
on port == %s" %(host_ip))
```

output:

```
"C:\Users\sohag\PycharmProjects\computer network\venv\Scripts\python.exe"  
Socket successfully created  
the socket has successfully connected to google on port == 172.217.31.36
```

```
C:\Users\sohag>ping 8.8.8.8  
  
Pinging 8.8.8.8 with 32 bytes of data:  
Reply from 8.8.8.8: bytes=32 time=273ms TTL=113  
Reply from 8.8.8.8: bytes=32 time=86ms TTL=113  
Reply from 8.8.8.8: bytes=32 time=103ms TTL=113  
Reply from 8.8.8.8: bytes=32 time=77ms TTL=113  
  
Ping statistics for 8.8.8.8:  
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),  
Approximate round trip times in milli-seconds:  
    Minimum = 77ms, Maximum = 273ms, Average = 134ms
```

```
C:\Users\sohag>ping 9.9.9.9

Pinging 9.9.9.9 with 32 bytes of data:
Reply from 9.9.9.9: bytes=32 time=268ms TTL=56
Reply from 9.9.9.9: bytes=32 time=174ms TTL=56
Reply from 9.9.9.9: bytes=32 time=177ms TTL=56
Reply from 9.9.9.9: bytes=32 time=74ms TTL=56

Ping statistics for 9.9.9.9:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 74ms, Maximum = 268ms, Average = 173ms
```

**Conclusion:** The processes that use a socket can reside on the same system or different systems on different networks. Sockets are useful for both stand-alone and network applications. Sockets allow you to exchange information between processes on the same machine or across a network, distribute work to the most efficient machine, and they easily allow access to centralized data. Socket application program interfaces (APIs) are the network standard for TCP/IP. A wide range of operating systems support socket APIs. i5/OS™ sockets support multiple transport and networking protocols. Socket system functions and the socket network functions are threadsafe.