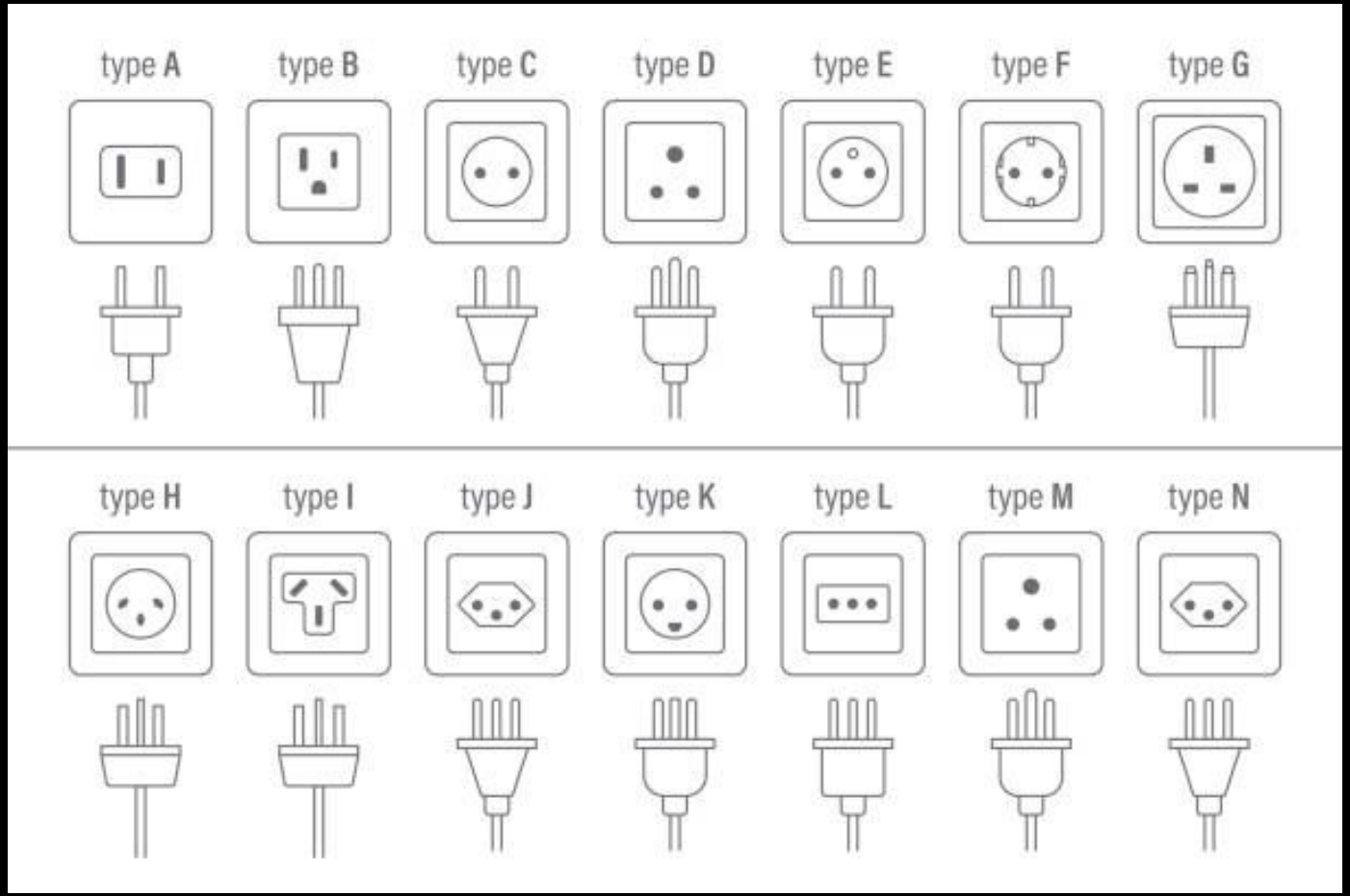
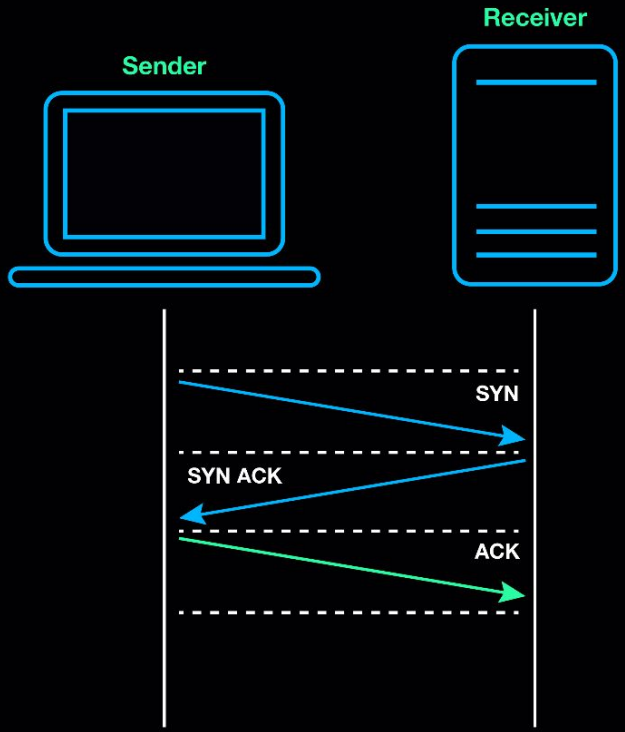
This is

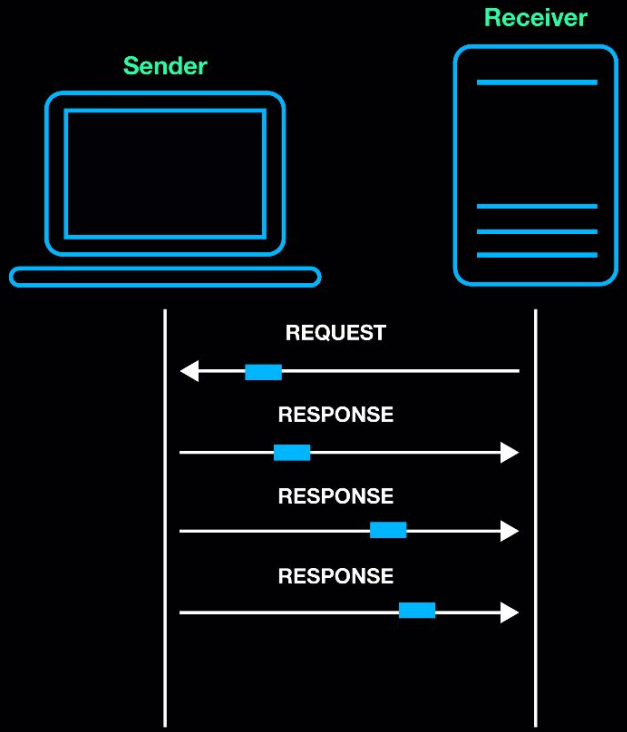
an Introduction to Socket Programming in Python

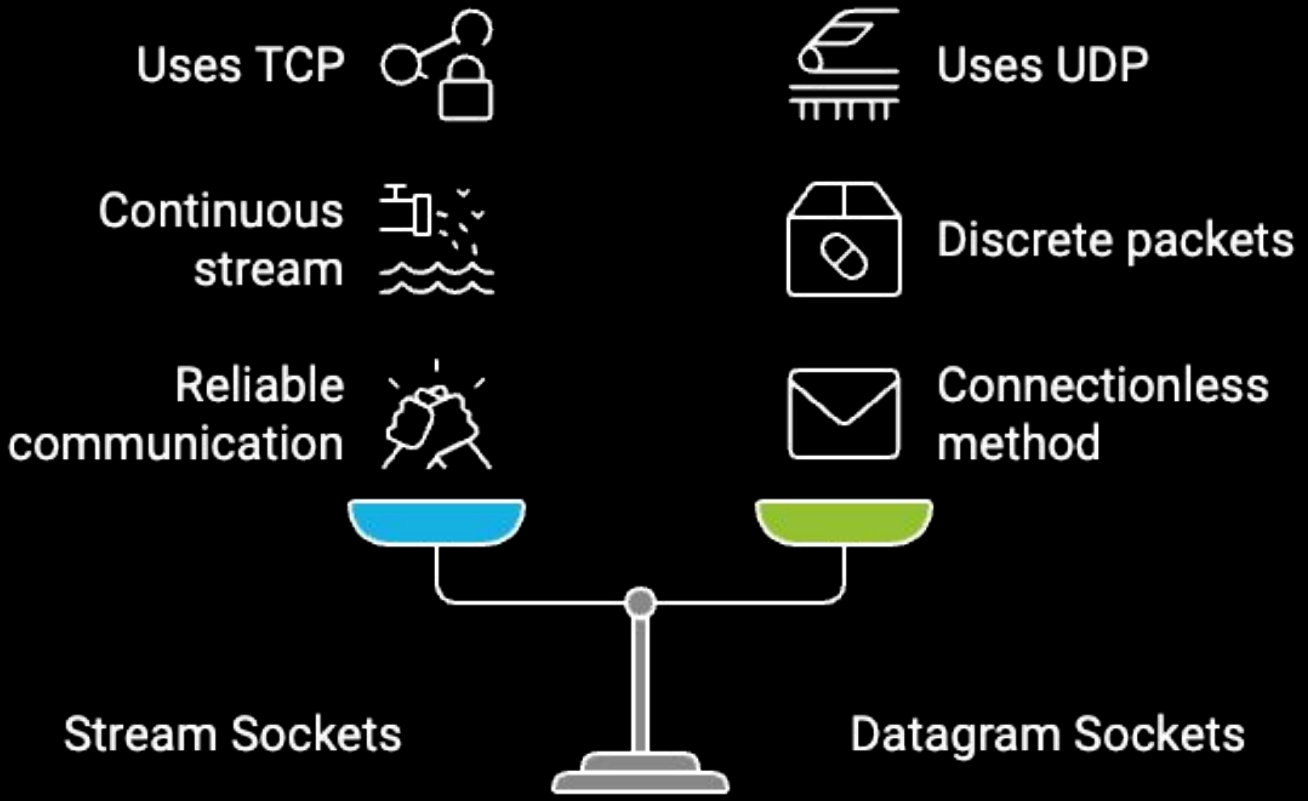

type A
type B
type C
type D
type E
type F
type G
type H
type I
type J
type K
type L
type M
type N

TCP


Receiver
Sender
SYN
SYN ACK
ACK

UDP


Receiver
Sender
REQUEST
RESPONSE
RESPONSE
RESPONSE



Comparing Stream and Datagram Sockets

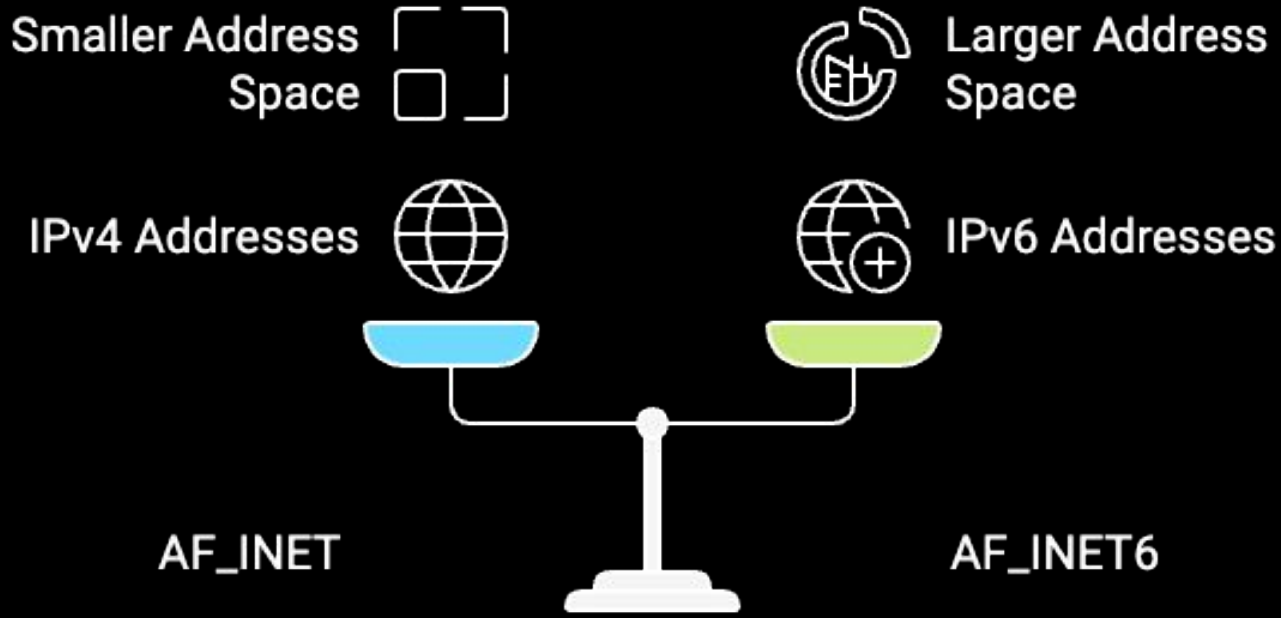
IPv4

123.45.67.89

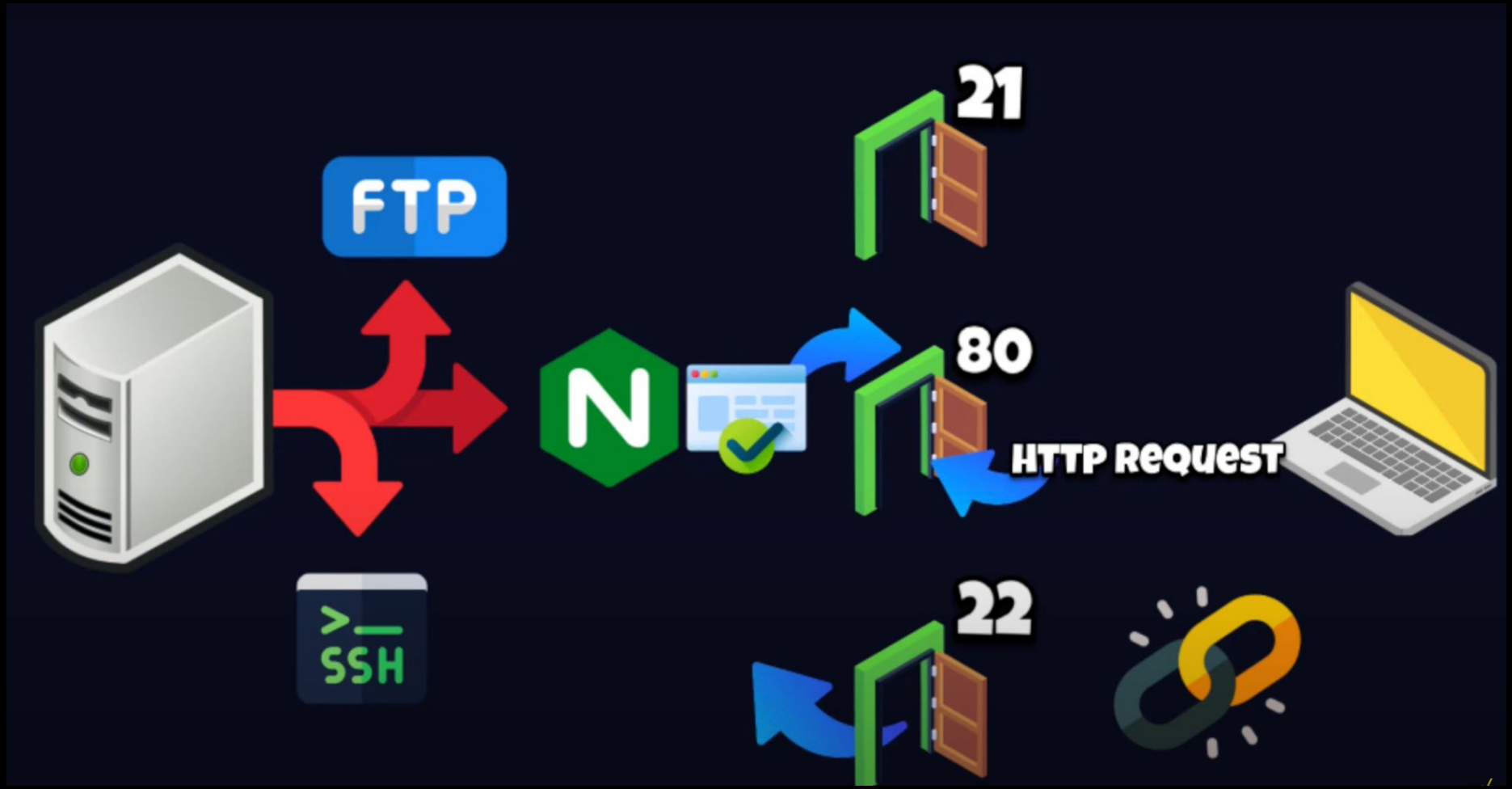
VS

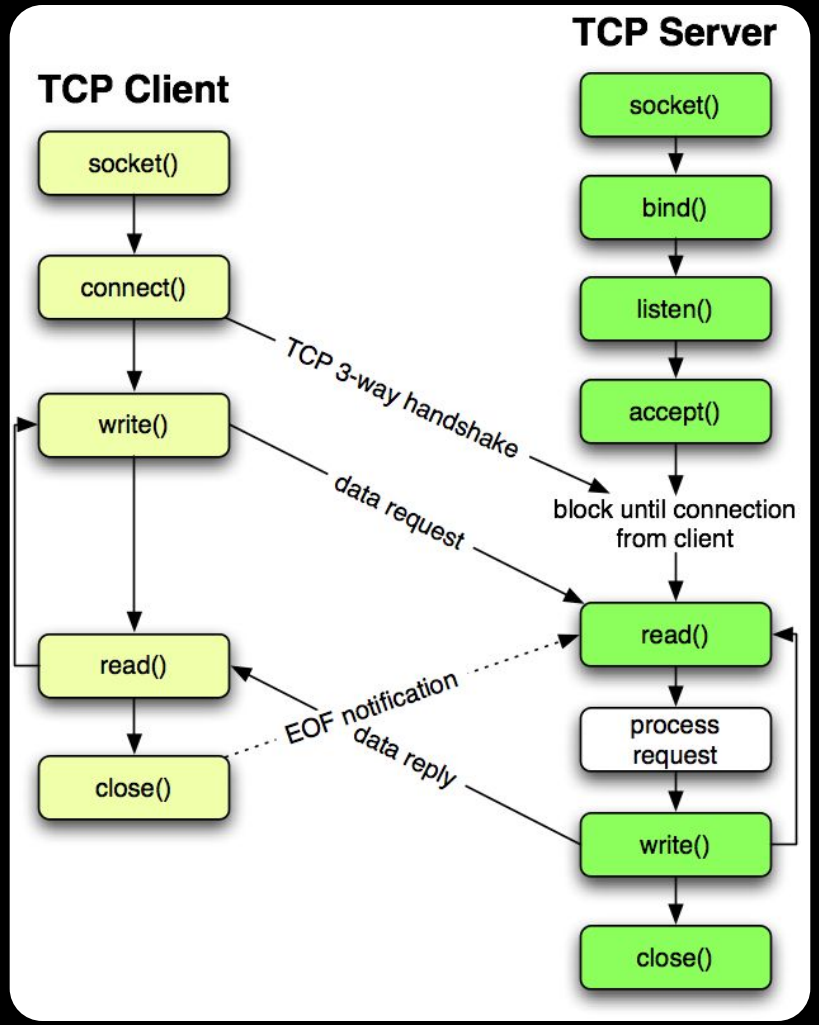
IPv6

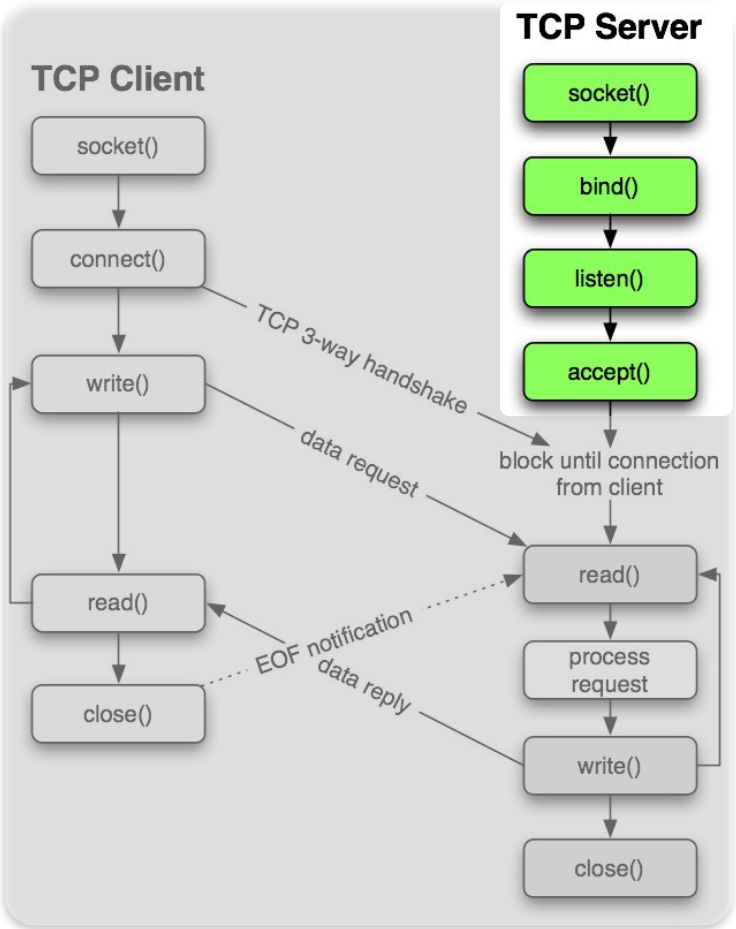
2023:db1::ww:42:6789



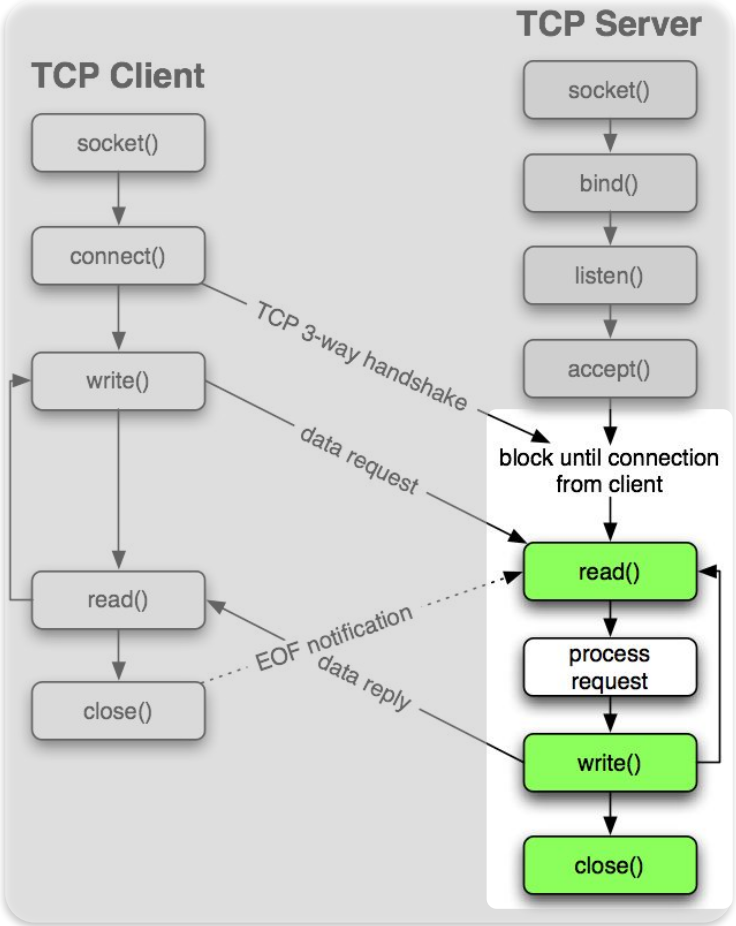
Comparing IPv4 and IPv6 in sockets programming.


FTP
21
80
HTTP Request
SSH
22

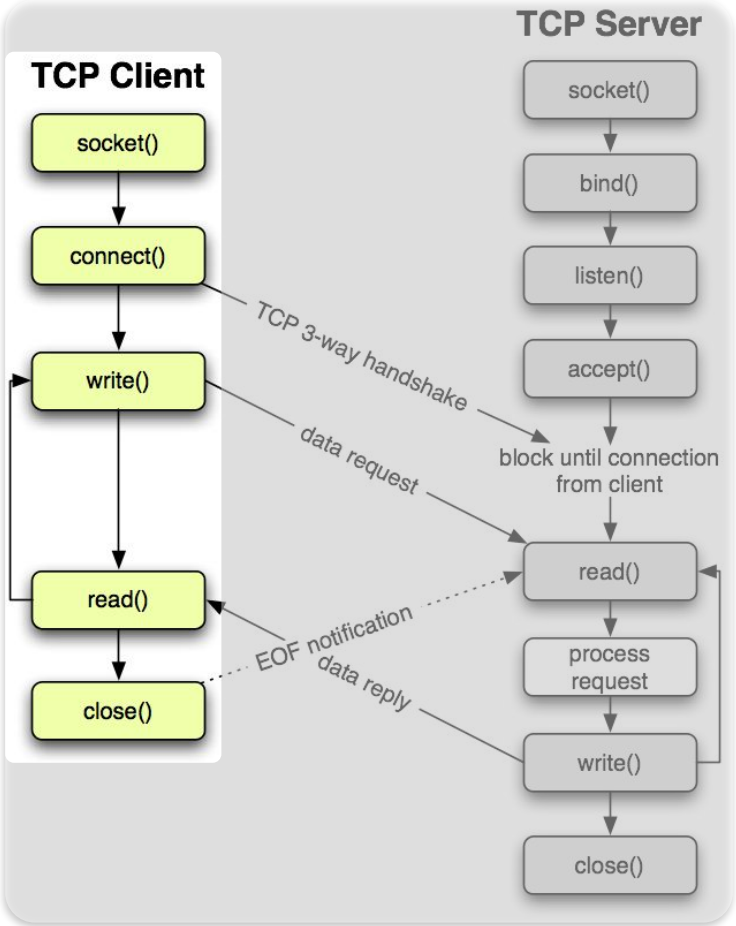

TCP Server
TCP Client
socket()
bind()
listen()
accept()
socket()
connect()
write()
read()
close()
TCP 3-way handshake
data request
block until connection from client
read()
process request
write()
close()
EOF notification
data reply




```python
import socket
import argparse

def start_server(host, port):
    """Initializes and starts the chat server."""
    # Create the TCP/IP socket
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # Bind the socket to the provided address and port
    server_socket.bind((host, port))
    print(f"Server started at {host}:{port}")

    # Enable the server to listen for connections
    server_socket.listen()
    print("Waiting for connections...")

    # Accept the client connection
    client_socket, client_address = server_socket.accept()
    print(f"Connection established with {client_address}")
```




```python
    try:
        # Loop to receive messages
        while True:
            message = client_socket.recv(1024).decode("utf-8")
            if not message:
                break  # If there is no message, close the connection
            print(f"Client: {message}")

            # Send response to the client
            response = input("Server: ")
            client_socket.sendall(response.encode("utf-8"))
    except Exception as e:
        print(f"Error: {e}")
    finally:
        # Close the connections
        client_socket.close()
        server_socket.close()
```




```python
import socket
import argparse

def start_client(host, port):
    """Initializes the chat client and connects to the server."""
    # Create the TCP/IP socket
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    try:
        # Connect to the server
        client_socket.connect((host, port))
        print(f"Connected to the server at {host}:{port}")

        # Loop to send and receive messages
        while True:
            message = input("Client: ")
            if message.lower() == 'exit':
                break  # Exit the chat by typing 'exit'

            # Send message to the server
            client_socket.sendall(message.encode('utf-8'))

            # Receive response from the server
            response = client_socket.recv(1024).decode('utf-8')
            print(f"Server: {response}")
    except Exception as e:
        print(f"Error: {e}")
    finally:
        # Close the connection
        client_socket.close()

if __name__ == "__main__":
    parser = argparse.ArgumentParser(
        description="Start a TCP/IP chat client.",
        usage='%(prog)s <host> <port>'
    )
    parser.add_argument("host", type=str, help="The server address (e.g., 'localhost' or an IP address).")
    parser.add_argument("port", type=int, help="The server port (integer between 1 and 65535).")
    args = parser.parse_args()

    start_client(args.host, args.port)
```