Bhuvana Kanakam
SE21UCSE035
CS4101 Lab 01
08.23.2024

Introduction to Inter-Process Communica¡on Using Sockets

Challenges Faced

I encountered some challenges while working on a socket-based client-server application for inter-process communication (IPC), particularly when attempting to establish a connection between the client and the server.

0.1 Port Conflict

What Happened: My initial attempt to bind the server to a port I believed would work was unsuccessful. I kept receiving an error message that said `"Address already in use."` It appears that another service on my computer was already using the port I was attempting to utilize.

How I Fixed It: I selected a different port number, one that wasn't in use. Specifically, after realizing this, I used port 9999, which was available and didn't interfere with any other services, thus it worked flawlessly.

0.2 IP Address Configuration

What Happened: I thought it would be enough to set up the server using the IP address of the local machine, which is 127.0.0.1 or localhost. However, because localhost only permits connections from the same machine, my attempt to connect from a different device was unsuccessful.

How I Fixed It: I realized that in order for clients on other computers to establish a connection, I had to either bind the server to my computer's IP address or use 0.0.0.0 to allow connections from any network interface. This modification allowed external clients to connect successfully.

Design and Implementation

Before diving into the code, here are a few key points explaining the client-server chat application:

- The client and server communicate using TCP sockets, which ensure reliable data transmission.
- The server listens on a specific IP address and port, waiting for incoming connections from clients.
- Once a connection is established, the client can send messages to the server, and the server can respond back.
- Both the client and server are designed to handle continuous communication until either side sends a "QUIT" message or a timeout occurs.

Client Code - Chat Implementation

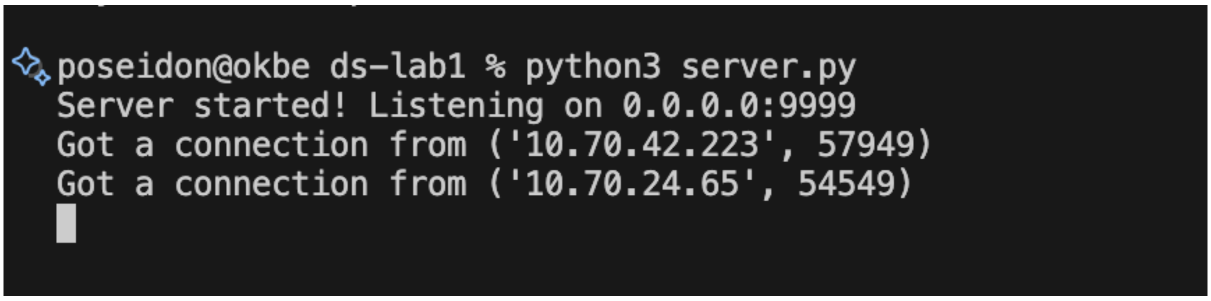
```
import socket
import time

HOST='10.70.34.14'
PORT=9990

client_socket=socket.socket(socket.AF_INET,socket.SOCK_STREAM)

client_socket.connect((HOST,PORT))

while True:
    message=input(f"Client: ")
    start_time=time.time()
    client_socket.sendall(message.encode())
    if(message=="QUIT" or time.time()-start_time >10):
        break
    response=client_socket.recv(1024).decode()
    print(f"Server : {response}")
    if(response=="QUIT"):
        break

client_socket.close()
```

Server Code - Chat Implementation

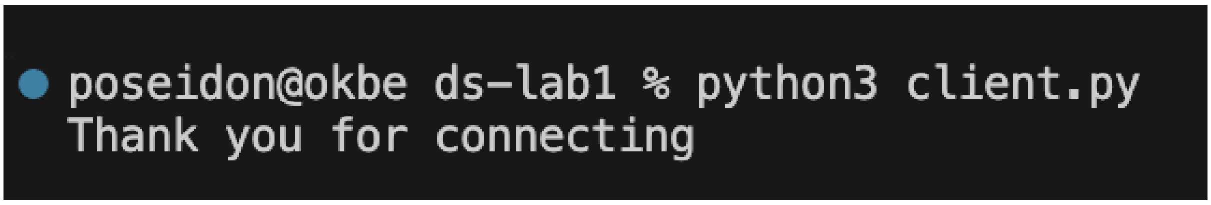
```
import socket

def start_server():
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    host = '0.0.0.0'
    port = 9990

    server_socket.bind((host, port))

    server_socket.listen(5)
    print(f"Server started! Listening on {host}:{port}")

    while True:
        client_socket, addr = server_socket.accept()
        print(f"Got a connection from {addr}")

        try:
            while True:
                client_message = client_socket.recv(1024).decode('ascii')
                if client_message.lower() == 'exit':
                    print("Client has disconnected.")
                    break

                print(f"Client: {client_message}")

                server_message = input("Server: ")
                client_socket.send(server_message.encode('ascii'))

                if server_message.lower() == 'exit':
                    print("Server is shutting down the connection.")
                    break
        except Exception as e:
            print(f"An error occurred: {e}")
        finally:
            client_socket.close()
            print("Connection closed.")

if __name__ == "__main__":
    start_server()
```

Outputs And Interactions

Connection Establishment with Multiple Clients

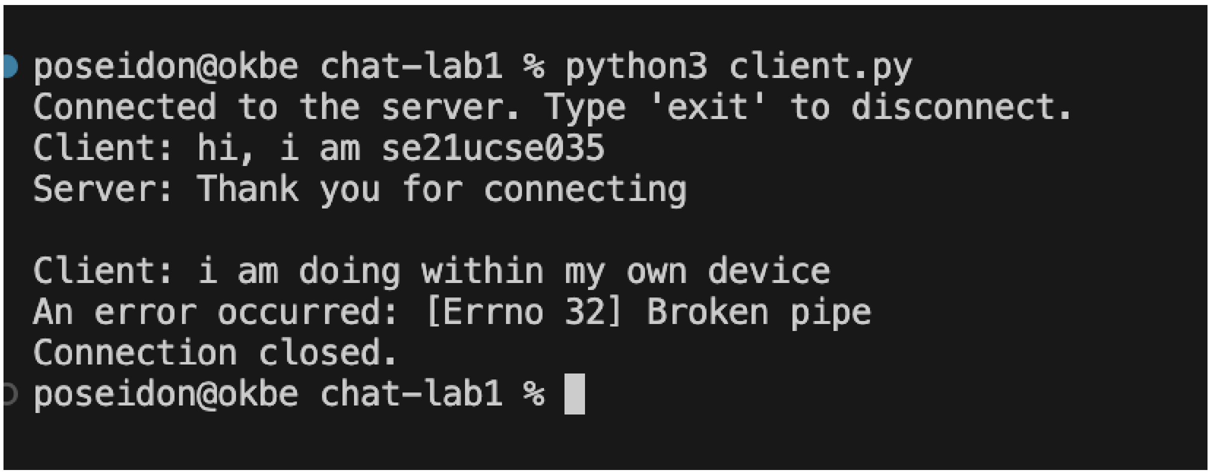


```
poseidon@okbe ds-lab1 % python3 server.py
Server started! Listening on 0.0.0.0:9999
Got a connection from ('10.70.42.223', 57949)
Got a connection from ('10.70.24.65', 54549)
```



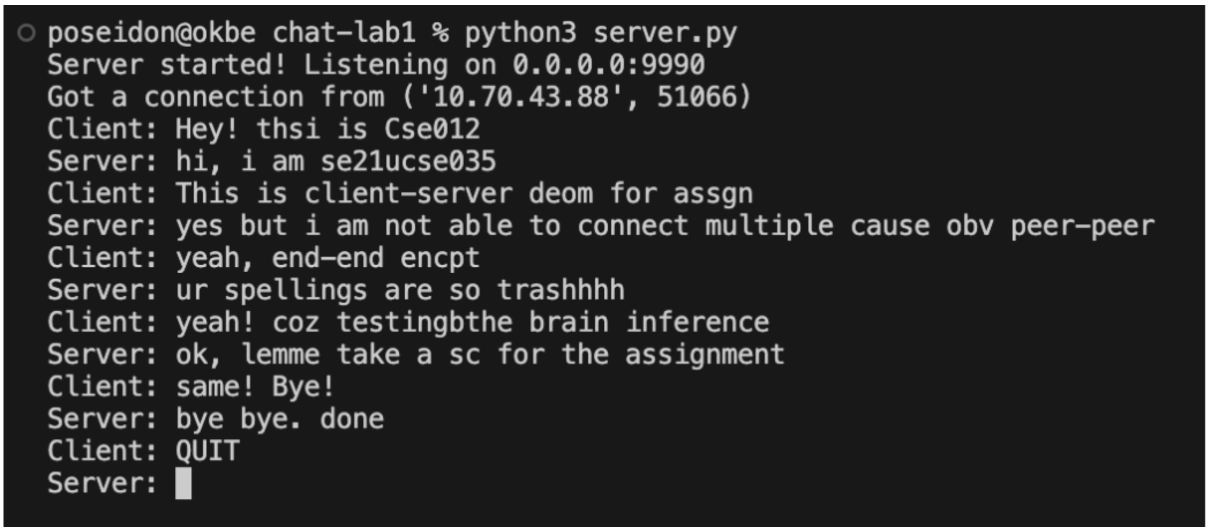
```
poseidon@okbe ds-lab1 % python3 client.py
Thank you for connecting
```

Chat Within My Own Device

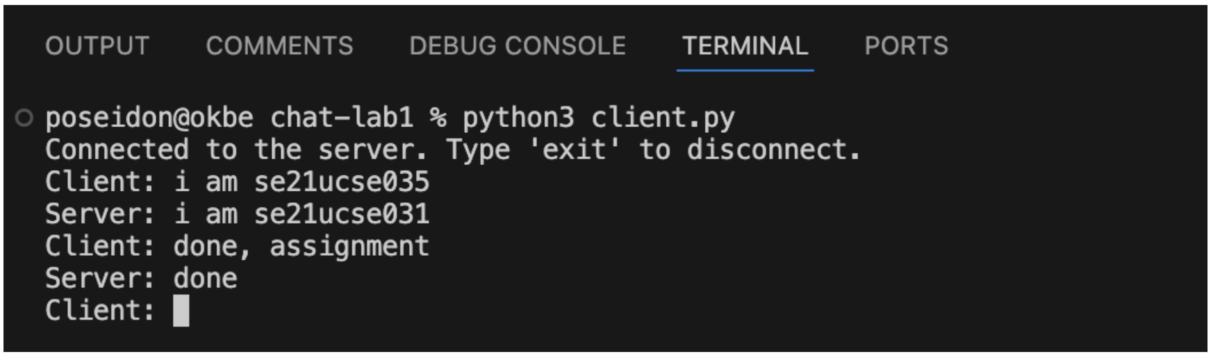


```
poseidon@okbe chat-lab1 % python3 client.py
Connected to the server. Type 'exit' to disconnect.
Client: hi, i am se21ucse035
Server: Thank you for connecting

Client: i am doing within my own device
An error occurred: [Errno 32] Broken pipe
Connection closed.
poseidon@okbe chat-lab1 %
```

Chat With Another Client



```
poseidon@okbe chat-lab1 % python3 server.py
Server started! Listening on 0.0.0.0:9990
Got a connection from ('10.70.43.88', 51066)
Client: Hey! thsi is Cse012
Server: hi, i am se21ucse035
Client: This is client-server deom for assgn
Server: yes but i am not able to connect multiple cause obv peer-peer
Client: yeah, end-end encpt
Server: ur spellings are so trashhhh
Client: yeah! coz testingbthe brain inference
Server: ok, lemme take a sc for the assignment
Client: same! Bye!
Server: bye bye. done
Client: QUIT
Server:
```



```
poseidon@okbe chat-lab1 % python3 client.py
Connected to the server. Type 'exit' to disconnect.
Client: i am se21ucse035
Server: i am se21ucse031
Client: done, assignment
Server: done
Client:
```