Want to communicate with your friends? Are you in a situation where you didn't have your mobiles to use chat applications? The systems you and your friends using are on the same network? I have come up with a solution, Simple Messaging System Using Python.

How to set up a Simple Messaging System and allow multiple clients to connect to it using a client-side script. This Project uses the concept of sockets and threading.

Sockets:

Sockets can be thought of as endpoints in a communication channel that is bi-directional, and establishes communication between a server and one or more clients. Here, we set up a socket on each end and allow a client to interact with other clients via the server. The socket on the server side associates itself with some hardware port on the server side. Any client that has a socket associated with the same port can communicate with the server socket.

Multi-Threading:

A thread is a sub-process that runs a set of commands individually or any other thread. So, every time a user connects to the server, a separate thread is created for that user and communication from server to client takes place along individual threads based on socket objects created for the sake of identity of each client.

We will require two scripts to establish this chat room. One to keep the serving running, and another that every client should run in order to connect to the server.

Server-Side Script:

The server side script will attempt to establish a socket and bind it to an IP address and port specified by the user (windows users might have to make an exception for the specified port number in their firewall settings, or can rather use a port that is already open). The script will then stay open and receive connection requests, and will append respective socket objects to a list to keep track of active connections. Every time a user connects,

a separate thread will be created for that user. In each thread, the server awaits a message, and sends that message to other users currently on the chat. If the server encountered an error while trying to receive a message from a particular thread, it will exit that thread.

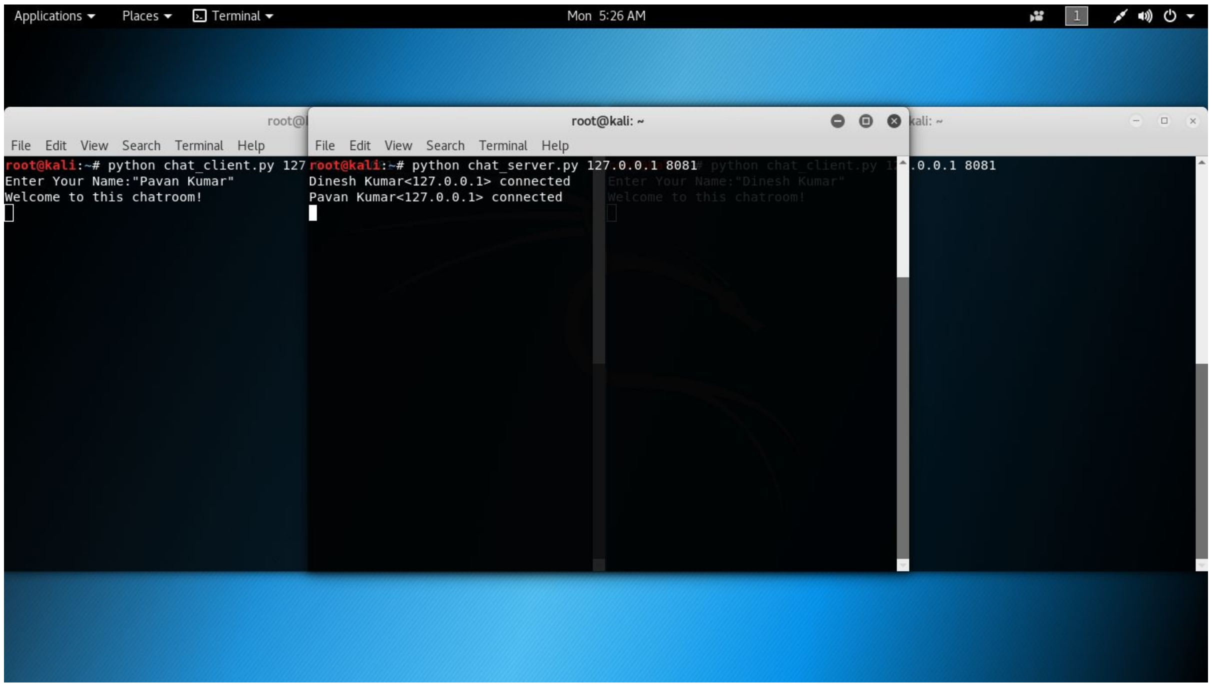
Client-Side Script:

The client side script will simply attempt to access the server socket created at the specified IP address and port. Once it connects, it will continuously check as to whether the input comes from the server or from the client, and accordingly redirects output. If the input is from the server, it displays the message on the terminal. If the input is from the user, it sends the message that the user enters to the server for it to be broadcasted to other users.

Usage:

This server can be set up on a local area network by choosing any on computer to be a server node, and using that computer's private IP address as the server IP address.

For example, if a local area network has a set of private IP addresses assigned ranging from 192.168.1.2 to 192.168.1.100, then any computer from these 99 nodes can act as a server, and the remaining nodes may connect to the server node by using the server's private IP address. Care must be taken to choose a port that is currently not in usage. For example, port 22 is default for ssh, and port 80 is default for HTTP protocols. So these two ports preferably, shouldnt be used or reconfigured to make them free for usage.



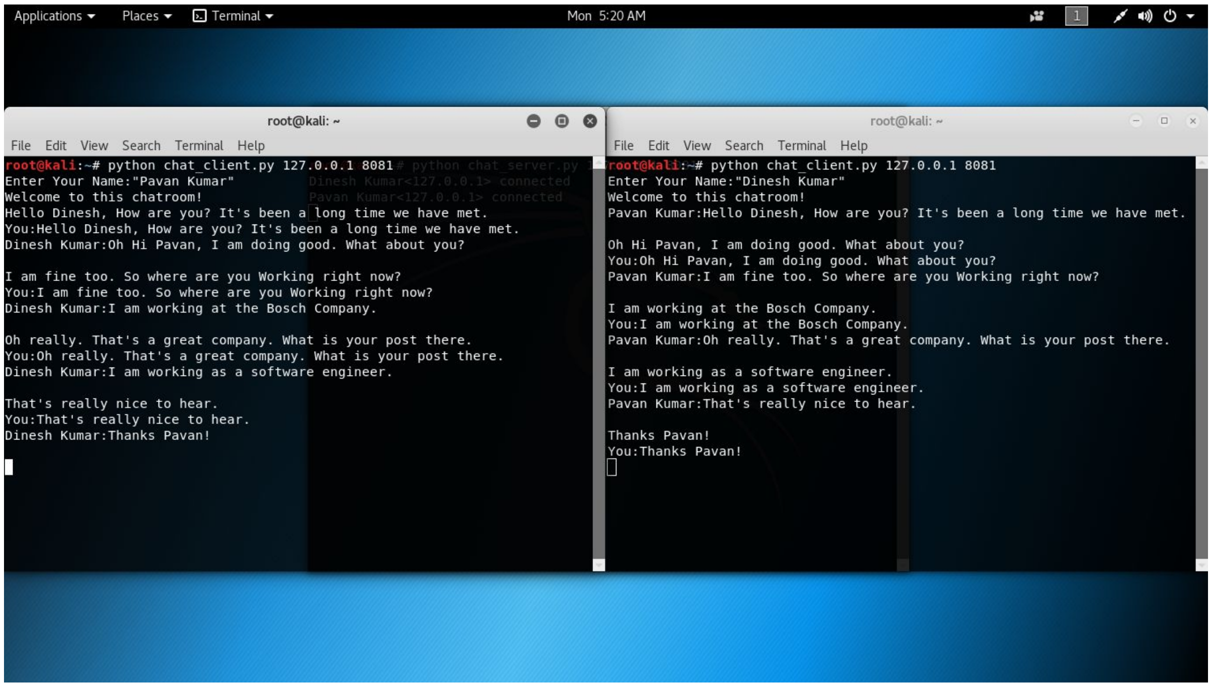
```
root@kali:~# python chat_client.py 127
Enter Your Name:"Pavan Kumar"
Welcome to this chatroom!
```

```
root@kali:~# python chat_server.py 127.0.0.1 8081
Dinesh Kumar<127.0.0.1> connected
Pavan Kumar<127.0.0.1> connected
```

```
# python chat_client.py 1
Enter Your Name:"Dinesh Kumar"
Welcome to this chatroom!
```

```
.0.0.1 8081
```

Snapshot shows the name and status of the clients connected to the server after starting it.



Applications
Places
Terminal
Mon 5:20 AM
root@kali: ~
File Edit View Search Terminal Help
root@kali: ~
File Edit View Search Terminal Help

```
root@kali:~# python chat_client.py 127.0.0.1 8081
Enter Your Name:"Pavan Kumar"
Welcome to this chatroom!
Hello Dinesh, How are you? It's been a long time we have met.
You:Hello Dinesh, How are you? It's been a long time we have met.
Dinesh Kumar:Oh Hi Pavan, I am doing good. What about you?

I am fine too. So where are you Working right now?
You:I am fine too. So where are you Working right now?
Dinesh Kumar:I am working at the Bosch Company.

Oh really. That's a great company. What is your post there.
You:Oh really. That's a great company. What is your post there.
Dinesh Kumar:I am working as a software engineer.

That's really nice to hear.
You:That's really nice to hear.
Dinesh Kumar:Thanks Pavan!
```

```
# python chat_server.py 1
Dinesh Kumar<127.0.0.1> connected
Pavan Kumar<127.0.0.1> connected
```

```
root@kali:~# python chat_client.py 127.0.0.1 8081
Enter Your Name:"Dinesh Kumar"
Welcome to this chatroom!
Pavan Kumar:Hello Dinesh, How are you? It's been a long time we have met.

Oh Hi Pavan, I am doing good. What about you?
You:Oh Hi Pavan, I am doing good. What about you?
Pavan Kumar:I am fine too. So where are you Working right now?

I am working at the Bosch Company.
You:I am working at the Bosch Company.
Pavan Kumar:Oh really. That's a great company. What is your post there.

I am working as a software engineer.
You:I am working as a software engineer.
Pavan Kumar:That's really nice to hear.

Thanks Pavan!
You:Thanks Pavan!
```

Snapshot shows the communication between the two clients.