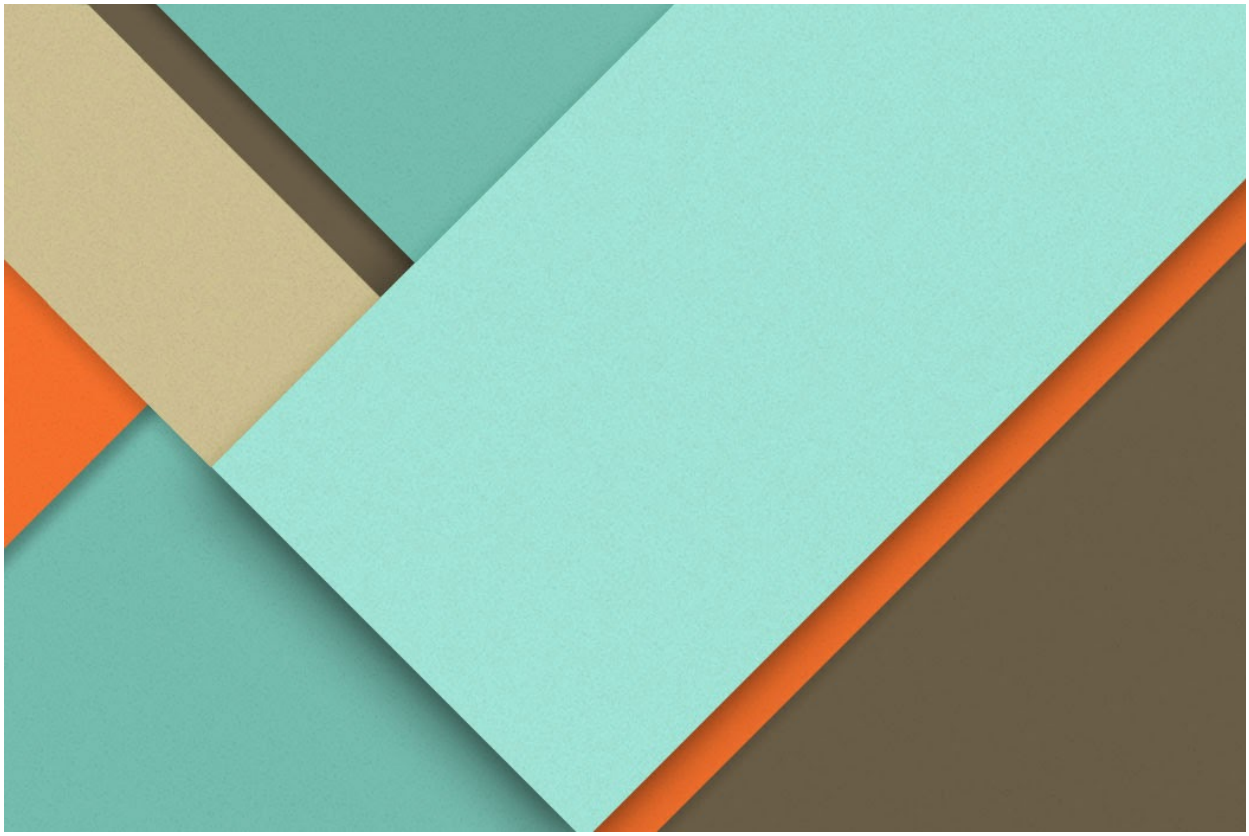

Different IP Address communication using socket programming in C

Group - 5

Overview

Try to establish a client server communication using TCP between the devices operating in two different networks. Let's take an example: Computer A operates with Jio network and Computer B operates with different Jio network or Vodafone or any network service providers.Try to establish a TCP connection between computer A and computer B.

Ngrok and SSH

Ngrok is a very lightweight tool that creates a secure tunnel on your local machine along with a public URL you can use for browsing your local site.

We use TCP protocol for connecting different hosts.

The below command is used by the host.

```
./ngrok <protocol> <port-number>
```

- **./ngrok:** The executable that starts the ngrok service.
- **protocol:** The protocol that needs to be used. Here, we use the **tcp** protocol.
- **port-number:** The port to be used for the connection. Here, we used the port **22.**

The below command is used by the client.

```
ssh <host-user-name>@<ip-address> -p<port-number>
```

- **ssh:** It provides a secure encrypted connection between two devices over an insecure network.

- **host-user-name:** The username of the host we are trying to connect.
- **ip-address:** The IP address given by the ngrok at the host side. It is of the pattern “0.tcp.ngrok.io” or “2.tcp.ngrok.io”.
- **port-number:** The virtual port number used to establish the connection. It is a 5-digit number.

socket

It creates a socket. It returns a socket description, similar to a file handler.

```
int socket(domain, type, protocol);
```

- **domain:** communication domain e.g., AF_INET (IPv4 protocol) , AF_INET6 (IPv6 protocol)
- **type:** communication type
  ➔ SOCK_STREAM: TCP(reliable, connection oriented)
  ➔ SOCK_DGRAM: UDP(unreliable, connectionless)
- **protocol:** communication protocol. Protocol value for Internet Protocol(IP), which is 0.

sockaddr_in

Structure describing an Internet socket address.

```
struct sockaddr_in{
   __SOCKADDR_COMMON (sin_);
   in_port_t sin_port;
```

```
    struct in_addr sin_addr;


    unsigned char sin_zero[sizeof (struct sockaddr) -
                __SOCKADDR_COMMON_SIZE -
                sizeof (in_port_t) -
                sizeof (struct in_addr)];
};
```

- **sin_family:** It is the communication domain e.g., AF_INET (IPv4 protocol) , AF_INET6 (IPv6 protocol)
- **sin_port:** It stores the port number for connection.
- **sin_addr:** It stores the IP address for connection. In server.c, we use INADDR_ANY, so that any client can connect. In client.c, we enter the public IP Address of the console, the server is running on.

bind

It binds the socket to the address and port number specified in addr(custom data structure).

```
int bind(int sockfd, const struct sockaddr *addr, socklen_t
addrlen);
```

- **sockfd:** It stores the socket description.
- **addr:** It is the structure that stores the address and port number to which the socket must bind to.
- **addrlen:** It is the size of the aforementioned structure.

listen

It puts the server socket in a passive mode, where it waits for the client to approach the server to make a connection.

```
int listen(int sockfd, int backlog);
```

- **sockfd:** It stores the socket description.
- **backlog:** It is the maximum number of clients that can wait in the queue for connection.

accept

It extracts the first client connection in the queue.

```
int new_socket= accept(int sockfd, struct sockaddr *addr,
socklen_t *addrlen);
```

- **new_socket:** It stores the socket description of the incoming client.
- **sockfd:** It stores the server socket description.
- **addr:** It is the structure that stores the address and port number to which the socket must accept.
- **addrlen:** It is the size of the aforementioned structure.

connect

It sends a connection request to the server.

```
int connect(int sockfd, struct sockaddr *addr, socklen_t
addrlen);
```

- **sockfd:** It stores the socket description.
- **addr:** It is the structure that stores the address and port number of the client.
- **addrlen:** It is the size of the aforementioned structure.

send

It sends a message to another socket.

```
int send(int sockfd, void *message, int size, int flags);
```

- **sockfd:** It stores the socket description.
- **message:** It is a pointer to the message that needs to be received. It can be any datatype, or a custom structure.
- **size:** It is the size of the buffer.
- **flags:** It is used to send any flags.

recv

It receives a message from another socket.

```
int recv(int sockfd, void *message, int size, int flags);
```

- **sockfd:** It stores the socket description.
- **message:** It is a pointer to the message that needs to be received. It can be any datatype, or a custom structure.
- **size:** It is the size of the buffer.
- **flags:** It is used to receive any flags from the sender.