# 7 Socket Programming : TCP

## 7.1 Aim

Implement Client-Server communication using Socket Programming and TCP as transport layer protocol.

## 7.2 Theory

### 7.2.1 TCP

Transmission Control Protocol(TCP) is a connection oriented protocol which emphasizes reliabilty of transmission over speed of transmission.TCP & IP work together to create a reliable data transmission network.

### 7.2.2 Client,Server and Socket

- **Server**: A server is a program that processes requests from client programs and replies to the requests accordingly
- **Client**:A client is a program that requests services from the server.The client program sends a request in a predefined format to the server and the server replies accordingly to the request.
- **Socket**:A socket is a way to communicate in a connection using file descriptors.They act as endpoints in a connection.

## 7.3 Algorithm

**Algorithm 1** Server

```
procedure MAIN PROCEDURE
    socketfd=socket(AF_INET,SOCK_STREAM,0)
    server.sin_family = AF_INET
    server.sin_addr.s_addr = htonl(INADDR_ANY)
    server.sin_port = htons(PORT)
    if bind(socketfd,(sockaddr *)&server,sizeof(server)<0 then
        printf("Socket binding failed !")
    end if
    printf("Socket binding successful !")
    if listen(socketfd,5)<0 then
        printf("Socket listening failed !")
        exit(0)
    end if
    printf("Socket listening successful")
    connfd = accept(socketfd,(sockaddr *)&client,&len)
    if connfd < 0 then
        printf("Accept failed !")
    end if
    printf("Accept successful")
    for ;; do
        read(connfd,buffer,sizeof(buffer))
        scanf("%s",buffer)
        write(connfd,buffer,sizeof(buffer))
    end for
end procedure
```

**Algorithm 2** Client

```
procedure MAIN PROCEDURE
    sockfd = socket(AF_INET,SOCK_STREAM,0)
    server.sin_family = AF_INET;
    server.sin_addr.s_addr = inet_addr("127.0.0.1");
    server.sin_port = htons(PORT);
    if connect(sockfd,(sockaddr *)&server,sizeof(server))!= 0 then
        printf("Connection failed !")
        exit(0)
    end if
    printf("Connection successful")
    for ;; do
        scanf("%s",buffer)
        write(sockfd,buffer,sizeof(buffer))
        read(sockfd,buffer,sizeof(buffer))
    end for
end procedure
```

## 7.4 Code

### Server

```c
#include <sys/socket.h>
#include <stdio.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>
#include <unistd.h>
#define PORT 8080
#define MAX 80


void communicator(int connfd){
    char buffer[MAX];

    int n;
    for(;;){
        memset(buffer,0,MAX);
        read(connfd,buffer,sizeof(buffer));

        printf("Client says : %s \n Server says : ",buffer);
        if(strcmp(buffer,"exit") == 0){
            printf("Server exiting..... \n");
            break;
        }
        memset(buffer,0,MAX);
        scanf("%s",buffer);
        write(connfd,buffer,sizeof(buffer));



    }
}

int main(){
    struct sockaddr_in server,client;
    int socketfd,connfd;
    socketfd = socket(AF_INET,SOCK_STREAM,0);
    if(socketfd == -1){
        printf("Socket creation failed !\n");
        exit(0);
    }
    printf("Socket creation successful !\n");
    memset(&server,0,sizeof(server));

    server.sin_family = AF_INET;
    server.sin_addr.s_addr = htonl(INADDR_ANY);
    server.sin_port = htons(PORT);

    if(bind(socketfd,(sockaddr *)&server,sizeof(server)) != 0){
        printf("Socket binding failed  !\n");
        exit(0);
    }
    printf("Socket binding successful !\n");
    if(listen(socketfd,5) < 0){
        printf("Listen failed !\n");
        exit(0);
    }
    printf("Server listening !\n");
    unsigned len = sizeof(client);
    connfd = accept(socketfd,(sockaddr *)&client,&len);
    if(connfd<0){
        printf("Accept failed ! \n");
        exit(0);
    }
    printf("Accept successful ! \n");
    communicator(connfd);
    close(socketfd);
    return 0;
}
```

### Client

```
#include<sys/socket.h>
#include<netinet/in.h>
#include<string.h>
#include<unistd.h>
#include<stdio.h>
#include<stdlib.h>
#include<netdb.h>
#include<arpa/inet.h>
#define PORT 8080
#define MAX 80
void communicator(int sockfd){
    char buffer[MAX];
    for(;;){
        memset(buffer,0,MAX);
```

```c
        scanf("%s",buffer);
        write(sockfd,buffer,sizeof(buffer));
        memset(buffer,0,MAX);
        read(sockfd,buffer,sizeof(buffer));
        printf("Server says : %s",buffer);
        if(strcmp(buffer,"exit")== 0){
            printf("Client exiting \n");
            break;
        }
    }
}

int main(){
    struct sockaddr_in server;
    int sockfd;
    sockfd = socket(AF_INET,SOCK_STREAM,0);
    if(sockfd == -1){
        printf("Socket creation failed ! \n");
        exit(0);
    }
    printf("Socket creation successful ! \n");
    memset(&server,0,sizeof(server));
    server.sin_family = AF_INET;
    server.sin_addr.s_addr = inet_addr("127.0.0.1");
    server.sin_port = htons(PORT);

    if(connect(sockfd,(sockaddr *)&server,sizeof(server))!= 0){
        printf("Connection failed ! \n");
        exit(0);
    }
    printf("Connected to the server ! \n");
    communicator(sockfd);
    close(sockfd);


}
```

## 7.5 Output

- **Server**

```
Socket creation successful !
Socket binding successful !
Server listening !
Accept successful !
Client says : helloworld123
Server says : bye
Client says : exit
Server exiting.....
```

- **Client**

```
Socket creation successful !
Connected to the server !
helloworld123
Server says : bye
exit
Client exiting
```

## 7.6 Result

Server and client was implemented in C++.The socket for the server is created first and the a port is binded to the server.The server then listens for any client connection and when a client connects,it is accepted and data is transmitted between the server and client.