

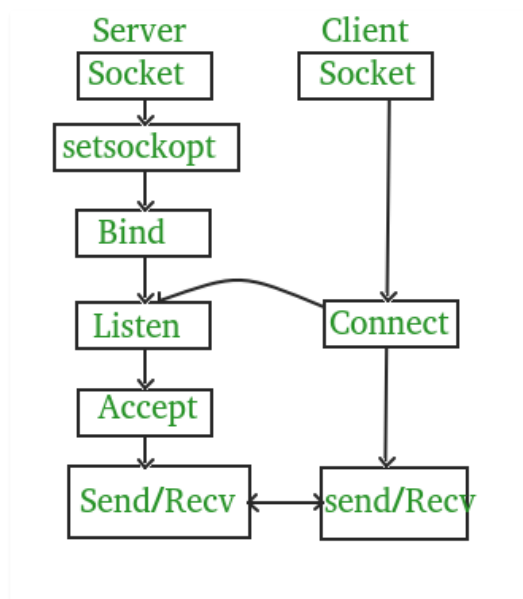


## TCP Server-Client implementation in C

Prerequisites – [Socket Programming in C/C++](#), [TCP and UDP server using select](#), [UDP Server-Client implementation in C](#)

If we are creating a connection between client and server using TCP then it has few functionality like, TCP is suited for applications that require high reliability, and transmission time is relatively less critical. It is used by other protocols like HTTP, HTTPS, FTP, SMTP, Telnet. TCP rearranges data packets in the order specified. There is absolute guarantee that the data transferred remains intact and arrives in the same order in which it was sent. TCP does Flow Control and requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control. It also does error checking and error recovery. Erroneous packets are retransmitted from the source to the destination.

The entire process can be broken down into following steps:



The entire process can be broken down into following steps:

### TCP Server –

1. using `create()`, Create TCP socket.
2. using `bind()`, Bind the socket to server address.
3. using `listen()`, put the server socket in a passive mode, where it waits for the client to approach the server to make a connection
4. using `accept()`, At this point, connection is established between client and server, and they are ready to transfer data.
5. Go back to Step 3.

### TCP Client –

1. Create TCP socket.
2. connect newly created client socket to server.

## TCP Server:

```
#include <netdb.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/types.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr

// Function designed for chat between client and server.
void func(int sockfd)
{
    char buff[MAX];
    int n;
    // infinite loop for chat
    for (;;) {
        bzero(buff, MAX);

        // read the message from client and copy it in buffer
        read(sockfd, buff, sizeof(buff));
        // print buffer which contains the client contents
        printf("From client: %s\t To client : ", buff);
        bzero(buff, MAX);
        n = 0;
        // copy server message in the buffer
        while ((buff[n++] = getchar()) != '\n')
            ;

        // and send that buffer to client
        write(sockfd, buff, sizeof(buff));

        // if msg contains "Exit" then server exit and chat ended.
        if (strcmp("exit", buff, 4) == 0) {
            printf("Server Exit...\n");
            break;
        }
    }
}

// Driver function
int main()
{
    int sockfd, connfd, len;
    struct sockaddr_in servaddr, cli;

    // socket create and verification
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1) {
        printf("socket creation failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully created..\n");
    bzero(&servaddr, sizeof(servaddr));

    // assign IP, PORT
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    servaddr.sin_port = htons(PORT);

    // Binding newly created socket to given IP and verification
    if ((bind(sockfd, (SA*)&servaddr, sizeof(servaddr))) != 0) {
        printf("socket bind failed...\n");
        exit(0);
    }
    ,
```

```

}
else
    printf("Socket successfully binded..\n");

// Now server is ready to listen and verification
if ((listen(sockfd, 5)) != 0) {
    printf("Listen failed...\n");
    exit(0);
}
else
    printf("Server listening..\n");
len = sizeof(cli);

// Accept the data packet from client and verification
connfd = accept(sockfd, (SA*)&cli, &len);
if (connfd < 0) {
    printf("server accept failed...\n");
    exit(0);
}
else
    printf("server accept the client...\n");

// Function for chatting between client and server
func(connfd);

// After chatting close the socket
close(sockfd);
}

```

## TCP Client:

```

// Write CPP code here
#include <netdb.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr
void func(int sockfd)
{
    char buff[MAX];
    int n;
    for (;;) {
        bzero(buff, sizeof(buff));
        printf("Enter the string : ");
        n = 0;
        while ((buff[n++] = getchar()) != '\n')
            ;
        write(sockfd, buff, sizeof(buff));
        bzero(buff, sizeof(buff));
        read(sockfd, buff, sizeof(buff));
        printf("From Server : %s", buff);
        if ((strcmp(buff, "exit", 4)) == 0) {
            printf("Client Exit...\n");
            break;
        }
    }
}

int main()
{
    int sockfd, connfd;
    struct sockaddr_in servaddr, cli;

    // socket create and varification
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1) {
        printf("socket creation failed...\n");
        exit(0);
    }
    else
        printf("Socket successfully created..\n");
    bzero(&servaddr, sizeof(servaddr));

    // assign IP, PORT
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
    servaddr.sin_port = htons(PORT);

    // connect the client socket to server socket
    if (connect(sockfd, (SA*)&servaddr, sizeof(servaddr)) != 0) {
        printf("connection with the server failed...\n");
        exit(0);
    }
    else
        printf("connected to the server..\n");

    // function for chat
    func(sockfd);

    // close the socket
    close(sockfd);
}

```

## Compilation –

Server side:

gcc server.c -o server

./server

Client side:

```
gcc client.c -o client
```

```
./client
```

### Output –

Server side:

```
Socket successfully created..  
Socket successfully binded..  
Server listening..  
server accept the client...  
From client: hi  
    To client : hello  
From client: exit  
    To client : exit  
Server Exit...
```

Client side:

```
Socket successfully created..  
connected to the server..  
Enter the string : hi  
From Server : hello  
Enter the string : exit  
From Server : exit  
Client Exit...
```

## Recommended Posts:

Implementation of ls | wc command

UDP Server-Client implementation in C

Implementation of a Falling Matrix

UDP Client Server using connect | C implementation

Implementation of Diffie-Hellman Algorithm

Peterson's Algorithm for Mutual Exclusion | Set 1 (Basic C implementation)

Java Implementation of Diffie-Hellman Algorithm between Client and Server

8086 program to check whether a string is palindrome or not

Lightweight Directory Access Protocol (LDAP)

8086 program to print a String

8086 program to reverse a string

Difference between Stateless and Stateful Protocol

Difference between pointer to an array and array of pointers

Difference between Bluetooth and Zigbee



**Yogesh Shukla 1**

Check out this Author's [contributed articles](#).

---

If you like GeeksforGeeks and would like to contribute, you can also write an article using [contribute.geeksforgeeks.org](https://contribute.geeksforgeeks.org) or mail your article to [contribute@geeksforgeeks.org](mailto:contribute@geeksforgeeks.org). See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please Improve this article if you find anything incorrect by clicking on the "Improve Article" button below.

Article Tags : [C](#) [Computer Networks](#) [Linux-Unix](#) [c-network-programming](#) [system-programming](#)

Practice Tags : [C](#) [Computer Networks](#)



2

☐ To-do ☐ Done

0

No votes  
yet.

[Feedback/ Suggest Improvement](#)

[Add Notes](#)

[Improve Article](#)

Please write to us at [contribute@geeksforgeeks.org](mailto:contribute@geeksforgeeks.org) to report any issue with the above content.

[Previous](#)

[◀ wmemset\(\) in C/C++ with Examples](#)

[Next](#)

[Computer Network | Open shortest path first \(OSPF\) – Set 2](#)

[▶](#)

Writing code in comment? Please use [ide.geeksforgeeks.org](https://ide.geeksforgeeks.org), generate link and share the link here.

[Load Comments](#)

Starting from **28<sup>th</sup> July 2019**



# PLACEMENT



~~₹14,999~~  
**₹7,499**

**Register Now**

**EARLY BIRD DISCOUNT IS ON\***

employable  
graduates



\* Starting from 28th June  
12:00 pm to 11:59 pm

## Most popular in C

putchar() function in C

C program to sort an array using pointers

Difference between pointer to an array and array of pointers

OpenMP | Hello World program

### Most visited in Computer Networks

IEEE 802.11 Mac Frame

Cryptography | One Time Password (OTP) algorithm

Computer Network | Introduction To Subnetting

Computer Network | Classless Inter Domain Routing (CIDR)

Computer Network | Finding Network ID of a Subnet (using Subnet Mask)



**FREE** Starting from **10<sup>th</sup> June 2019**

# SUDO

## PLACEMENT

2019

**Register Now**

The banner features a dark teal background with a large, stylized illustration of a person in a red cape and white shirt, holding a briefcase, standing on a large white paper airplane. A hand is shown launching the paper airplane. The background includes a large globe and several white stars. A dashed line connects the 'Register Now' button to the 'SUDO' logo. In the bottom right corner, there is a logo consisting of the letters 'GG' in a stylized font.



# GeeksforGeeks

A computer science portal for geeks

5th Floor, A-118,  
Sector-136, Noida, Uttar Pradesh - 201305  
[feedback@geeksforgeeks.org](mailto:feedback@geeksforgeeks.org)

## COMPANY

[About Us](#)  
[Careers](#)  
[Privacy Policy](#)  
[Contact Us](#)

## LEARN

[Algorithms](#)  
[Data Structures](#)  
[Languages](#)  
[CS Subjects](#)  
[Video Tutorials](#)

## PRACTICE

[Courses](#)  
[Company-wise](#)  
[Topic-wise](#)  
[How to begin?](#)

## CONTRIBUTE

[Write an Article](#)  
[Write Interview Experience](#)  
[Internships](#)  
[Videos](#)