19CSE312 Distributed Systems - S6 CSE D 2022

Labsheet 1

Socket Programming

Done By

Adharsh S Mathew
AM.EN.U4CSE19302
CSE-D

Server Program

```
//TCP
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>

int main(void)
{
   int socket_desc, client_sock, client_size;
   struct sockaddr_in server_addr, client_addr;
   char server_message[2000], client_message[2000];

   // Clean buffers:
   memset(server_message, '\0', sizeof(server_message));
   memset(client_message, '\0', sizeof(client_message));

   // Create socket:
   socket_desc = socket(AF_INET, SOCK_STREAM, 0);

   if(socket_desc < 0){
       printf("Error while creating socket\n");
       return -1;
   }
   printf("Socket created successfully\n");

   // Set port and IP:
   server_addr.sin_family = AF_INET;
   server_addr.sin_port = htons(2000);
   server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

   // Bind to the set port and IP:
   if(bind(socket_desc, (struct sockaddr*)&server_addr,
sizeof(server_addr))<0){
       printf("Couldn't bind to the port\n");
       return -1;
   }
   printf("Done with binding\n");
```

```
   // Listen for clients:
   if(listen(socket_desc, 1) < 0){
       printf("Error while listening\n");
       return -1;
   }
   printf("\nListening for incoming connections.....\n");

   // Accept an incoming connection:
   client_size = sizeof(client_addr);
   client_sock = accept(socket_desc, (struct sockaddr*)&client_addr,
&client_size);

   if (client_sock < 0){
       printf("Can't accept\n");
       return -1;
   }
   //printf("Client connected at IP: %s and port: %i\n",
inet_ntoa(client_addr.sin_addr), ntohs(client_addr.sin_port));
   do
   {
       // Clean buffers:
       memset(server_message, '\0', sizeof(server_message));
       memset(client_message, '\0', sizeof(client_message));
       // Receive client's message:
       if (recv(client_sock, client_message, sizeof(client_message), 0) <
0){
           printf("Couldn't receive\n");
           return -1;
       }
       printf("Msg from client: %s\n", client_message);

       // Respond to client:
       // Get input from the user:
       printf("Enter message: ");
       gets(server_message);

       if (send(client_sock, server_message, strlen(server_message), 0) <
0){
           printf("Can't send\n");
           return -1;
```

```
        }

        if(strcasecmp(client_message,"STOP")==0)
        {
            // Closing the socket:
        close(client_sock);
        close(socket_desc);
        exit(0);
        }


    } while (1);

    return 0;
}
```

Output

```
(base) adharsh@adharsh-Inspiron-5570:~/Home-HDD/asm/Github/CN_socket_programming/TCP$ ./server_tcp
Socket created successfully
Done with binding

Listening for incoming connections.....
Msg from client: hi
Enter message: hello
Msg from client: how u
Enter message: good u
Msg from client: noice!!!
Enter message: stop
Msg from client:
Enter message: stop
Msg from client:
Enter message: ^C
(base) adharsh@adharsh-Inspiron-5570:~/Home-HDD/asm/Github/CN_socket_programming/TCP$
```

Client Program

```
//TCP
#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <arpa/inet.h>

int main(void)
{
   int socket_desc;
   struct sockaddr_in server_addr;
   char server_message[2000], client_message[2000];

   // Clean buffers:
   memset(server_message,'\0',sizeof(server_message));
   memset(client_message,'\0',sizeof(client_message));

   // Create socket:
   socket_desc = socket(AF_INET, SOCK_STREAM, 0);

   if(socket_desc < 0){
       printf("Unable to create socket\n");
       return -1;
   }
   printf("Socket created successfully\n");
   // Set port and IP the same as server-side:
   server_addr.sin_family = AF_INET;
   server_addr.sin_port = htons(2000);
   server_addr.sin_addr.s_addr = inet_addr("127.0.0.1");

   // Send connection request to server:
   if(connect(socket_desc, (struct sockaddr*)&server_addr,
sizeof(server_addr)) < 0){
       printf("Unable to connect\n");
       return -1;
   }
   printf("Connected with server successfully\n");

   do
```

```
    {
        // Clean buffers:
        memset(server_message, '\0', sizeof(server_message));
        memset(client_message, '\0', sizeof(client_message));
        // Get input from the user:
        printf("Enter message: ");
        gets(client_message);
        // Send the message to server:
        if(send(socket_desc, client_message, strlen(client_message), 0) <
0){
            printf("Unable to send message\n");
            return -1;
        }
        // Receive the server's response:
        if(recv(socket_desc, server_message, sizeof(server_message), 0) <
0){
            printf("Error while receiving server's msg\n");
            return -1;
        }
        printf("Server's response: %s\n",server_message);

        if(strcasecmp(server_message,"STOP")==0)
        {
            // Close the socket:
        close(socket_desc);
        exit(0);
        }

    } while (1);
    return 0;
}
```

Output

```
(base) adharsh@adharsh-Inspiron-5570:~/Home-HDD/asm/Github/CN_socket_programming/TCP$ ./client_tcp
Socket created successfully
Connected with server successfully
Enter message: hi
Server's response: hello
Enter message: how u
Server's response: good u
Enter message: noice!!!
Server's response: stop
(base) adharsh@adharsh-Inspiron-5570:~/Home-HDD/asm/Github/CN_socket_programming/TCP$
```