Thapar Institute of Engineering and Technology, Patiala

Computer Science and Engineering

Network Programming Laboratory (UCS413)

Assignment 2

Create an Echo-Server using TCP socket programming in connection- oriented Scenario. Echo Server echo back (return back) the message sent by the client. Also, write the Client side program.

While creating program focus on the use of following socket programming functions.

- *socket(), sockaddr_in, bind(), listen(), accept(), ntohs(), ntohsl(), read()/recv(), write()/send() , connect().*

Steps

Server Side:

- Include appropriate header files.
- Create a TCP Socket.
- Bind the address and port using bind() system call.
- Server executes listen() system call to indicate its willingness to receive connections.
- Accept the next completed connection from the client process by using an accept() system call. At this point, connection is established between client and server, and they are ready to transfer data.
- Receive a message from the Client using recv()/read() system call.
- Send the received message back(echo) to the client using send()/write() system call.
- Close the socket using close() system call

Client Side:

- Include appropriate header files
- Create a TCP Socket.
- Establish connection to the Server using connect() system call.
- Send and recieve messages using send() and recv() system call respectively.
- Close the socket using close() system call

Execution Steps:

- . Save client and server program into two separate file with .c extension.
- . Open two terminal and execute .c files by following commands
- . gcc filename.c –o filename (compilation)
- ./filename (run)

Server-side Program: EchoServer.c

```
/*Required Headers*/

#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <stdio.h>
#include<string.h>

int main()
{
   char str[100];
   int listen_fd, comm_fd;
   struct sockaddr_in servaddr;
   listen_fd = socket(AF_INET, SOCK_STREAM, 0);

   bzero( &servaddr, sizeof(servaddr));

   servaddr.sin_family = AF_INET;
   servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
   servaddr.sin_port = htons(22000);

   bind(listen_fd, (struct sockaddr *) &servaddr, sizeof(servaddr));

   listen(listen_fd, 10);



   while(1)
   {
      comm_fd = accept(listen_fd, (struct sockaddr*) NULL, NULL);
       bzero( str, 100);

       recv(comm_fd,str,100,0);

       printf("Echoing back - %s",str);

        send(comm_fd,str,strlen(str),0);
                close(comm_fd);
   }
}
```

Client Side Program: EchoClient.c

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <stdio.h>
#include<string.h>

int main(int argc,char **argv)
{
   int sockfd,n;
   char sendline[100];
   char recvline[100];
   struct sockaddr_in servaddr;

   sockfd=socket(AF_INET,SOCK_STREAM,0);
   bzero(&servaddr,sizeof servaddr);

   servaddr.sin_family=AF_INET;
   servaddr.sin_port=htons(22000);
   servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
   //inet_pton(AF_INET,"127.0.0.1",&(servaddr.sin_addr));

   connect(sockfd,(struct sockaddr *)&servaddr,sizeof(servaddr));

   while(1)
   {
      bzero( sendline, 100);
      bzero( recvline, 100);
      fgets(sendline,100,stdin); /*stdin = 0 , for standard input */

                send(sockfd,sendline,strlen(sendline),0);
                recv(sockfd,recvline,100,0);
      printf("%s",recvline);
   }

}
```