Week: 12 Design TCP Client-server application to transfer file

Aim: Write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.

Client Side:

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<arpa/inet.h>

#include<fcntl.h>

#include<string.h>

#define SERV\_TCP\_PORT 6880

#define SERV\_HOST\_ADDR "127.0.0.1"

int main()

{ int sockfd;

struct sockaddr\_in serv\_addr,cli\_addr;

char filename[100],buf[1000];

int n;

serv\_addr.sin\_family=AF\_INET;

serv\_addr.sin\_addr.s\_addr=inet\_addr(SERV\_HOST\_ADDR);

serv\_addr.sin\_port=htons(SERV\_TCP\_PORT);

if((sockfd=socket(AF\_INET,SOCK\_STREAM,0))<0)

printf("Client:cant open stream socket\n");

else

printf("Client:stream socket opened successfully\n");

if(connect(sockfd,(struct sockaddr \*)&serv\_addr,

sizeof(serv\_addr))<0)

printf("Client:cant connect to server\n");

else

printf("Client:connected to server successfully\n");

printf("\n Enter the file name to be displayed :");

scanf("%s",filename);

write(sockfd,filename,strlen(filename));

printf("\n filename transfered to server\n");

n=read(sockfd,buf,1000);

if(n < 0)

printf("\n error reading from socket");

printf("\n Client : Displaying file content of %s\n",filename);

fputs(buf,stdout);

close(sockfd);

exit(0);

}

Output:

AT CLIENT SIDE

[root@localhost ]# cc tcpc.c

[root@localhost ]# ./a.out

Data Sent

File Content....

Sockets are a mechanism for exchanging data between processes. These processes can

either be on the same machine, or on different machines connected via a network. Once a socket

connection is established, data can be sent in both directions until one of the endpoints closes the

connection.

I needed to use sockets for a project I was working on, so I developed and refined a few

C++ classes to encapsulate the raw socket API calls. Generally, the application requesting the

data is called the client, and the application servicing the request is called the server. I created

two primary classes, ClientSocket and ServerSocket, that the client and server could use to

exchange data.

SERVER SIDE:

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<arpa/inet.h>

#include<fcntl.h>

#include<string.h>

#define SERV\_TCP\_PORT 6880

#define SERV\_HOST\_ADDR "127.0.0.1"

int main()

{ int sockfd,newsockfd,clilen;

struct sockaddr\_in cli\_addr,serv\_addr;

char filename[25],buf[1000];

int n,m=0;

int fd;

if((sockfd=socket(AF\_INET,SOCK\_STREAM,0))<0)

printf("server:cant open stream socket\n");

else

printf("server:stream socket opened successfully\n");

serv\_addr.sin\_family=AF\_INET;

serv\_addr.sin\_addr.s\_addr=htonl(INADDR\_ANY);

serv\_addr.sin\_port=htons(SERV\_TCP\_PORT);

if((bind(sockfd,(struct sockaddr \*)

&serv\_addr,sizeof(serv\_addr)))<0)

printf("server:cant bind local address\n");

else

printf("server:bound to local address\n");

listen(sockfd,5);

printf("\n SERVER : Waiting for client...\n");

for(;;)

{

clilen=sizeof(cli\_addr);

newsockfd=accept(sockfd,(struct sockaddr \*)

&cli\_addr,&clilen);

if(newsockfd<0)

printf("server:accept error\n");

else

printf("server:accepted\n");

n=read(newsockfd,filename,25);

filename[n]='\0';

printf("\n SERVER : %s is found and ready to transfer

\n",filename);

fd=open(filename,O\_RDONLY);

n=read(fd,buf,1000);

buf[n]='\0';

write(newsockfd,buf,n);

printf("\n transfer success\n");

close(newsockfd);

exit(0)

}

}

Output:

[root@localhost ]# cc tcps.c

[root@localhost ]# ./a.out

Received the file name : data.txt

File content sent