**PROGRAM 14. IMPLEMENTATION OF FTP CLIENT AND SERVER**

Aim: - Program to implement simple FTP server for file transfer to server.

**Steps involved in writing the Server Process:**

1. Create a socket using socket( ) system call with address family AF\_INET, type SOCK\_STREAM and default protocol.

2. Bind server’s address and port using bind( ) system call.

3. Wait for client connection to complete accepting connections using accept( ) system call.

5. Receive the Clients file using recv() system call .

6. Using \*fgets(char \*str, int n, FILE \*stream) function, we read a line of text from the specified stream and stores it into the string pointed to by str. It stops when either (n-1) characters are read, or when the end-of-file is reached.

7. On successful execution i.e. when file pointer reaches end of file, file transfer “completed” message is sent by the server to the accepted client connection using newsd, socket file descriptor.

**Steps involved in writing the Client Process:**

1. Create a socket system call with address family AF\_INET, type SOCK\_STREAM and default protocol.

2. Enter the client port id

2. Fill in the internet socket address structure (with server information).

3. Connect to the server address using connect() system call.

4. Read the existing and new file name from user.

5. Send existing file to server using send() system call

6. Receive feedback from server “Completed”, regarding file transfer completion.

5. Write “File is transferred” to standard output screen.

6. Close the socket connection and file pointer.

**/\*FTP server\*/**

#include<stdio.h>

#include<sys/types.h>

#include<netinet/in.h>

#include<string.h>

#include<stdlib.h>

#include<unistd.h>

int main()

{

FILE \*fp;

int sd,newsd,ser,n,a,cli,pid,bd,port,clilen;

char name[100],fileread[100],fname[100],ch,file[100],rcv[100];

struct sockaddr\_in servaddr,cliaddr;

printf("Enter the port address\n");

scanf("%d",&port);

sd=socket(AF\_INET,SOCK\_STREAM,0);

if(sd<0)

printf("Cant create\n");

else

printf("Socket is created\n");

servaddr.sin\_family=AF\_INET;

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

servaddr.sin\_port=htons(port);

a=sizeof(servaddr);

bd=bind(sd,(struct sockaddr \*)&servaddr,a);

if(bd<0)

printf("Cant bind\n");

else

printf("Binded\n");

listen(sd,5);

clilen=sizeof(cliaddr);

newsd=accept(sd,(struct sockaddr \*)&cliaddr,&clilen);

if(newsd<0)

{

printf("Cant accept\n");

}

else

printf("Accepted\n");

n=recv(newsd,rcv,100,0);

rcv[n]='\0';

fp=fopen(rcv,"r");

if(fp==NULL)

{

send(newsd,"error",5,0);

close(newsd);

}

else

{

while(fgets(fileread,sizeof(fileread),fp))

{

if(send(newsd,fileread,sizeof(fileread),0)<0)

{

printf("Can’t send file contents\n");

}

sleep(1);

}

if(!fgets(fileread,sizeof(fileread),fp))

{

//when file pointer reaches end of file, file transfer “completed” message is send to accepted client connection using newsd, socket file descriptor.

send(newsd,"completed",999999999,0);

}

return(0);

}

}

**/\*FTP Client\*/**

#include<stdio.h>

#include<stdlib.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<unistd.h>

int main()

{

FILE \*fp;

int csd,n,ser,s,cli,cport,newsd;

char name[100],rcvmsg[100],rcvg[100],fname[100];

struct sockaddr\_in servaddr;

printf("Enter the port");

scanf("%d",&cport);

csd=socket(AF\_INET,SOCK\_STREAM,0);

if(csd<0)

{

printf("Error....\n");

exit(0);

}

else

printf("Socket is created\n");

servaddr.sin\_family=AF\_INET;

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

servaddr.sin\_port=htons(cport);

if(connect(csd,(struct sockaddr \*)&servaddr,sizeof(servaddr))<0)

printf("Error in connection\n");

else

printf("connected\n");

printf("Enter the existing file name\t");

scanf("%s",name);

printf("Enter the new file name\t");

scanf("%s",fname);

fp=fopen(fname,"w");

send(csd,name,sizeof(name),0);

while(1)

{

s=recv(csd,rcvg,100,0);

rcvg[s]='\0';

if(strcmp(rcvg,"error")==0)

printf("File is not available\n");

if(strcmp(rcvg,"completed")==0)

{

printf("File is transferred........\n");

fclose(fp);

close(csd);

break;

}

else

fputs(rcvg,stdout);

fprintf(fp,"%s",rcvg);

return 0;

}

}

**OUTPUT**

*CREATE FILE -*

hello.txt containing “Hello world”

**ftpserver.c**

gcc ftps.c -o fs

./fs

Enter the port address

5030

Socket is created

Binded

Accepted

**ftpclient.c**

gcc ftpc.c -o fc

./fc

Enter the port5030

Socket is created

connected

Enter the existing file name hello.txt

Enter the new file name hi.txt

Hello world

File is transferred........

*AFTER FILE TRANSFER-*

*2 files will appear*

hello.txt containing “Hello world”

hi.txt containing “Hello world”