DAYANANDA SAGAR UNIVERSITY

UNIX PROGRAMMING LABORATORY REPORT

ON

FILE TRANSFER PROTOCOL (RFC959)

BACHELOR OF TECHNOLOGY

IN

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Submitted By:

Bhavanaagamya B S – ENG17CS0051

Bhavya M – ENG17CS0052

C V Vishwas– ENG17CS0054

Chandan Lal P N – ENG17CS0055

VI Semester, 2020

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SCHOOL OF ENGINEERING

DAYANANDA SAGAR UNIVERSITY

KUDLU GATE

BANGALORE - 560068

CERTIFICATE

This is to certify that the Network and UNIX Programming Laboratory Mini-Project report entitled **"FILE TRANSFER PROTOCOL"** being submitted by **Ms. Bhavanaagamya B.S, Ms. Bhavya M, Mr. C V Vishwas, Mr. Chandan Lal** to Department of Computer Science and Engineering, School of Engineering, Dayananda Sagar University, Bangalore, for the 6th semester B.Tech C.S.E of this university during the academic year 2020-2021.

Date:_____ _____

Signature of the Faculty in Charge

_____

Signature of the Chairman

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

We extend our sincere thanks to our Chairman **Dr. Banga M.K** for providing necessary departmental facilities and moral support and encouragement.

We would like to express our deep sense of gratitude to **Dr. Mallanagouda Patil** for initiating us into this project by providing information regarding this project, guidance, support, motivation and patience which helped us for successful completion of our mini- project work.

We have received a great deal of guidance and co-operation from our friends and we wish to thank one and all that have directly or indirectly helped us in the successful completion of this mini-project work.

BHAVANAAGAMYA B.S (ENG17CS0051)

BHAVYA M (ENG17SCS0052)

C V VISHWAS (ENG17CS0054)

CHANDAN LAL (ENG17CS0055)

DECLARATION

We hereby declare the work prescribed in the Mini Project entitled “File Transfer Protocol” has been carried out by us and it has not been submitted for the award of any degree, diploma or the mini project report of any other college or university.

BHAVANAAGAMYA B.S (ENG17CS0051)

BHAVYA M (ENG17SCS0052)

C V VISHWAS (ENG17CS0054)

CHANDAN LAL (ENG17CS0055)

ABSTRACT

FTP stands for the File transfer protocol and is used to transfer files between an FTP server and another computer. In the past, FTP was much more common than it is today and was the dominant file transfer mechanism on the Internet. If you needed to transfer files between two computers, you would use FTP to do so. FTP is still very popular today when a service requires that a lot of files be hosted for other to people to download. FTP also tends to be faster than other contemporary methods of transferring files because it was designed to do so.

The File Transfer Protocol (FTP) is a standard network protocol used for the transfer of computer files between a client and server on a computer network. FTP is built on a client-server model architecture using separate control and data connections between the client and the server. File Transfer Protocol (FTP) is an application layer protocol which moves files between local and remote file systems. It runs on the top of TCP, like HTTP. To transfer a file, 2 TCP connections are used by FTP in parallel: control connection and data connection

CONTENTS

1. INTRODUCTION

File Transfer Protocol (FTP) is a client/server protocol used for transferring files to or exchanging files with a host computer. It may be authenticated with user names and passwords. Anonymous FTP allows users to access files, programs and other data from the Internet without the need for a user ID or password. Web sites are sometimes designed to allow users to use 'anonymous' or 'guest' as a user ID and an email address for a password. Publicly available flies are often found in a directory called pub and can be easily FTPed to a user's computer. FTP is also the Internet standard for moving or transferring files from one computer to another using TCP or IP networks. File Transfer Protocol is also known as RFC 959.

The original FTP specification was written by Abhay Bhushan and published as RFC 114 on April 16, 1971. This was later replaced by RFC 765 (June 1980). The current specification is RFC 959 (October 1985). RFC stands for request for comments.
The first FTP client applications used the DOS command prompt with standardized commands and syntax. Since then, many graphical user interface (GUI) clients have been developed within operating systems, making it easier for the user to upload and download files.

For sending control information like user identification, password, commands to change the remote directory, commands to retrieve and store files, etc., FTP makes use of control connection. The control connection is initiated on port number 21. For sending the actual file, FTP makes use of data connection. A data connection is initiated on port number 20. FTP sends the control information out-of-band as it uses a separate control connection. Some protocols send their request and response header lines and the data in the same TCP connection. For this reason, they are said to send their control information in-band. HTTP and SMTP are such examples.

2. SOCKET PROGRAMMING

Java Socket programming is used for communication between the applications running on different JRE.Java Socket programming can be connection-oriented or connection-less.Socket and ServerSocket classes are used for connection-oriented socket programming and DatagramSocket and DatagramPacket classes are used for connection-less socket programming.

The client in socket programming must know two information:

1. IP Address of Server, and
2. Port number.

3. PROBELM STATEMENT

To implement File Transfer between client and server using socket programming in java.

4. HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE REQUIREMENTS

- Laptop/Personal Computer
- I3or above processor
- 4GB RAM

SOFTWARE REQUIREMENTS

- Ubuntu Operating System
- Editor
- Java Compiler

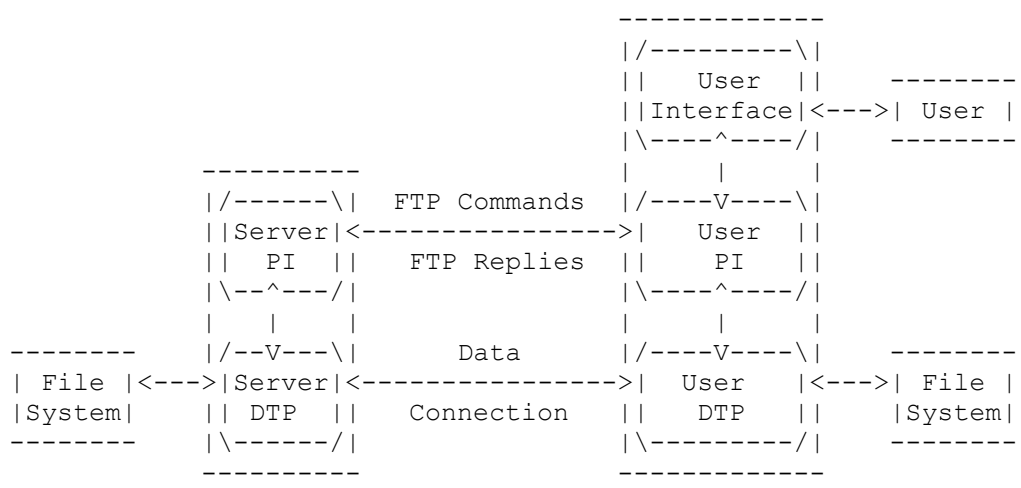
5. OBJECTIVE

- It provides the sharing of files.
- It is used to encourage the use of remote computers.
- It transfers the data more reliably and efficiently.

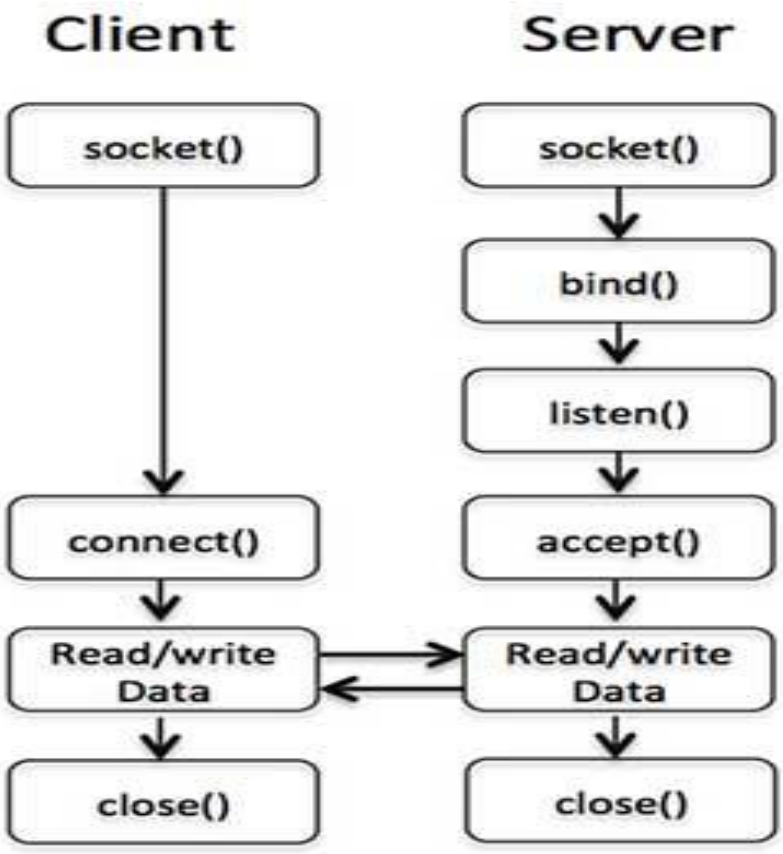
6.WORKING PROCEDURE

FTP protocol falls under the client server model. This means that the client sends the orders and waits for the server to carry out the mentioned instructions. In the server side the server begins execution by creating an object of server socket class and listens to incoming client connections in port 21. When a client connection is established, it creates input and output streams for the connection. If the connection is successful then the server waits for a command from the client. If a send command is sent, a file name is read from the input file stream and the content of the file is requested by the client. Then the file content is sent to the client from the server. The server then waits for the next command from the client. If a receive command is sent then the server requests for the name of the file and permission to read the file contents. After receiving permission, it accepts the file from the client and waits for another command. If it receives a disconnect command it disconnects from the client and shuts the server down. In the client side the client establishes a connection to the server program using Ip address and port number of the server. It then creates input output streams for the established connection. Then it displays the list of commands or options that the user has. The user can then send the get, receive or exit command upon which the server executes them. If a file with the same filename that a user is trying to send already exists on the receiver's side, then the user is asked if he wants to replace the existing file and then the file is sent or received.

7.DESIGN



NOTES: 1. The data connection may be used in either direction.
2. The data connection need not exist all of the time.



8. CODE

Client:

```
import java.net.*;
import java.io.*;
import java.util.*;


class FTPClient
{
   public static void main(String args[]) throws Exception
   {
      Socket soc=new Socket("127.0.0.1",1200);
      transferfileClient t=new transferfileClient(soc);
      t.displayMenu();

   }
}
class transferfileClient
{
   Socket ClientSoc;

   DataInputStream din;
   DataOutputStream dout;
   BufferedReader br;
   transferfileClient(Socket soc)
   {
      try
      {
         ClientSoc=soc;
         din=new DataInputStream(ClientSoc.getInputStream());
         dout=new DataOutputStream(ClientSoc.getOutputStream());
         br=new BufferedReader(new InputStreamReader(System.in));
      }
      catch(Exception ex)
      {
      }
   }
   void SendFile() throws Exception
   {

      String filename;
      System.out.print("Enter File Name :");
      filename=br.readLine();

      File f=new File(filename);
      if(!f.exists())
      {
         System.out.println("File not Exists...");
         dout.writeUTF("File not found");
```

```
return;

 }

      dout.writeUTF(filename);

      String msgFromServer=din.readUTF();
      if(msgFromServer.compareTo("File Already Exists")==0)
      {
         String Option;
         System.out.println("File Already Exists. Want to OverWrite (Y/N) ?");
         Option=br.readLine();
         if(Option=="Y")
         {
            dout.writeUTF("Y");
         }
         else
         {
            dout.writeUTF("N");
            return;
         }
      }

      System.out.println("Sending File ...");
      FileInputStream fin=new FileInputStream(f);
      int ch;
      do
      {
         ch=fin.read();
         dout.writeUTF(String.valueOf(ch));
      }
      while(ch!=-1);
      fin.close();
      System.out.println(din.readUTF());

   }

   void ReceiveFile() throws Exception
   {
      String fileName;
      System.out.print("Enter File Name :");
      fileName=br.readLine();
      dout.writeUTF(fileName);
      String msgFromServer=din.readUTF();

      if(msgFromServer.compareTo("File Not Found")==0)
      {
         System.out.println("File not found on Server ...");
         return;
      }
      else if(msgFromServer.compareTo("READY")==0)
      {
```

```
System.out.println("Receiving File ...");
        File f=new File(fileName);
        if(f.exists())
        {

  String Option;
            System.out.println("File Already Exists. Want to OverWrite (Y/N) ?");
            Option=br.readLine();
            if(Option=="N")
            {
               dout.flush();
               return;
            }
        }
        FileOutputStream fout=new FileOutputStream(f);
        int ch;
        String temp;
        do
        {
            temp=din.readUTF();
            ch=Integer.parseInt(temp);
            if(ch!=-1)
            {
               fout.write(ch);
            }
        }while(ch!=-1);
        fout.close();
        System.out.println(din.readUTF());

     }


   }

   public void displayMenu() throws Exception
   {
      while(true)
      {
         System.out.println("[ MENU ]");
         System.out.println("1. Send File");
         System.out.println("2. Receive File");
         System.out.println("3. Exit");
         System.out.print("\\nEnter Choice :");
         int choice;
         choice=Integer.parseInt(br.readLine());
         if(choice==1)
         {
            dout.writeUTF("SEND");
            SendFile();
         }
```

```
 else if(choice==2)
        {

                    dout.writeUTF("GET");
          ReceiveFile();
        }
        else
        {
          dout.writeUTF("DISCONNECT");
          System.exit(1);

 }
      }
   }
}
```

Server:

```
import java.net.*;
import java.io.*;
import java.util.*;

public class FTPServer
{
   public static void main(String args[]) throws Exception
   {
      ServerSocket soc=new ServerSocket(1200);
      System.out.println("FTP Server Started on Port Number 5217");
      while(true)
      {
         System.out.println("Waiting for Connection ...");
         transferfile t=new transferfile(soc.accept());

      }
   }
}

class transferfile extends Thread
{
   Socket ClientSoc;

   DataInputStream din;
   DataOutputStream dout;

   transferfile(Socket soc)
   {
      try
      {
         ClientSoc=soc;
         din=new DataInputStream(ClientSoc.getInputStream());
         dout=new DataOutputStream(ClientSoc.getOutputStream());
         System.out.println("FTP Client Connected ...");
         start();
```

```
    }
    catch(Exception ex)
    {
    }
}
void SendFile() throws Exception
{
    String filename=din.readUTF();
    File f=new File(filename);
    if(!f.exists())
    {
        dout.writeUTF("File Not Found");
        return;
    }
    else
    {
        dout.writeUTF("READY");
        FileInputStream fin=new FileInputStream(f);
        int ch;
        do
        {
            ch=fin.read();
            dout.writeUTF(String.valueOf(ch));
        }
        while(ch!=-1);
        fin.close();
        dout.writeUTF("File Receive Successfully");
    }
}

void ReceiveFile() throws Exception
{
    String filename=din.readUTF();
    if(filename.compareTo("File not found")==0)
    {
        return;
    }
    File f=new File(filename);
    String option;

    if(f.exists())
    {
        dout.writeUTF("File Already Exists");
        option=din.readUTF();
    }
    else
    {
        dout.writeUTF("SendFile");
        option="Y";
    }

        if(option.compareTo("Y")==0)
```

```
            {
               FileOutputStream fout=new FileOutputStream(f);
               int ch;
               String temp;
               do
               {
                  temp=din.readUTF();


 ch=Integer.parseInt(temp);

if(ch!=-1)
                  {
                     fout.write(ch);
                  }

}while(ch!=-1);
               fout.close();
               dout.writeUTF("File Send Successfully");
            }
            else
            {
               return;
            }
      }
      public void run()
      {
         while(true)
         {
            try
            {
            System.out.println("Waiting for Command ...");
            String Command=din.readUTF();
            if(Command.compareTo("GET")==0)
            {
               System.out.println("\\tGET Command Received ...");
               SendFile();
               continue;
            }
            else if(Command.compareTo("SEND")==0)
            {
               System.out.println("\\tSEND Command Receiced ...");
               ReceiveFile();
               continue;
            }
            else if(Command.compareTo("DISCONNECT")==0)
            {
               System.out.println("\\tDisconnect Command Received ...");
               System.exit(1);
            }
            }
            catch(Exception ex)
            {
```

```
            }
        }
    }
}
```

9. OUTPUT SCREENSHOTS

Server Side:

```
chandan@awat-VirtualBox:~/Desktop/USPProj/server$ ls
server  server.java  t2
chandan@awat-VirtualBox:~/Desktop/USPProj/server$ sudo java server.java
FTP Server Started on Port Number 21
Waiting for Connection ...
FTP Client Connected ...
Waiting for Connection ...
Waiting for Command ...
SEND Command Received ...
Waiting for Command ...
GET Command Received ...
Waiting for Command ...
SEND Command Received ...
Waiting for Command ...
Disconnect Command Received ...
chandan@awat-VirtualBox:~/Desktop/USPProj/server$ ls
server  server.java  t1  t2
chandan@awat-VirtualBox:~/Desktop/USPProj/server$
```

Client Side:

```
chandan@awat-VirtualBox:~/Desktop/USPProj/client$ ls
client.java  t1
chandan@awat-VirtualBox:~/Desktop/USPProj/client$ java client.java
[ MENU ]
1. Send File
2. Receive File
3. Exit
Enter Choice :1
Enter File Name :t1
Sending File ...
File Send Successfully
[ MENU ]
1. Send File
2. Receive File
3. Exit
Enter Choice :2
Enter File Name :t2
Receiving File ...
File Receive Successfully
[ MENU ]
1. Send File
2. Receive File
3. Exit
Enter Choice :1
Enter File Name :t1
File Already Exists. Want to OverWrite (Y/N) ?
y
[ MENU ]
1. Send File
2. Receive File
3. Exit
Enter Choice :3
chandan@awat-VirtualBox:~/Desktop/USPProj/client$ ls
client.java  t1  t2
chandan@awat-VirtualBox:~/Desktop/USPProj/client$
```

10. CONCLUSION

We are introducing the implementation of working of File Transfer Protocol that shows how the files can be transferred between a client and a server. The demo is made more interactive with a keyboard and mouse interaction in the program.

11. FUTURE SCOPE

This project may be useful to implement a server that can handle many clients at the same time. Features like encryption of data, passwords can be added for security of data in the files transferred.

12.REFERENCES

I. https://tools.ietf.org/html/rfc959
II. https://en.wikipedia.org/wiki/File_Transfer_Protocol
III. https://www.geeksforgeeks.org/socket-programming-cc/