Bhushan Asodi

Roll NO :- 53 Sub :- CNSL

Experiment NO 01

Lab assignment an Unit I (mandatory Assigment)

* Title :- study of existing LAN

* objective :-

① To understand the structure and working of various network including the interconnecting device used in them.

② To get hands on experience of making and testing cables.

* problem statement :- Part A :- setup a wired LAN using Layer 2 switch and the IP switch of minimum for computer It includes prepration of cable, testing of cable using the line tester, configuration machine using IP addresses testing using PING utility and demonstrate the PING packets captured traces using wireshark packet Analyser tools.

* Types of Network :- Common example of area network types are.

LAN :- Local Area Network
WLAN :- wireless Local Area Network.
WAN :- Wide Area Network
MAN :- Metropolitan Area Network
SAN :- Storage Area Network System Area Network, Server Area Network.
or sometimes smay Area Network.

Scanned with CamScanner

CAN :- Campus Area Network / Controller Area Network
or sometimes Cluster Area Network.

PAN :- Personal Area Network.

DAN :- Desk Area Network

* Types of cables :- Cable is the medium through which information usually moves from one network device to another. There are several types of cables which are commonly used with LAN & In some cases a network will utilize only one type of cable other network will use a variety of cable types The type of cable chosen for a network is related to the network topology, protocol and size Understanding the characteristics of different types of cables and how they related to other aspects of a network is necessary for the development of a successful network.

The following seetions discuss the types of cables used in network and other related topics.

* unshielded Twisted pair (UTP) cable
* Shielded Twisted pair (STP) cable.
* Coaxial cable
* fiber optics cable
* cable Installation Guides.

Scanned with CamScanner

* Wireless LAN

* Network Devices :- ① Gateway ② Router ③ Bridge ④ Switch ⑤ Hub ⑥ Repeater.

* Network topology :- The study of Network topology recognizes server. basic topology.

- Point-to-point topology
- Bus (point-to-multipoint) topology
- Star topology
- Ring topology
- Tree topology
- mesh topology
- Hybrid topology

Application :- ① most network in the real world use the above mentioned devices to create network.

Experiment No 02

Title :— Study of CRC and Hamming code.

objective :— To understand the structure and working of CRC and Hamming code.

problem statement :— write a program for error detection and correction for 7/8 bits ASCII coder using Hamming codes or CRC. Demonstrate the packet captured Traces using wireshark packet Analyzer tool for peer to peer mode (so, student will perform Hamming code and other will perform (CRC).

theroy :— Introduction :— All communication system try to make sure that the transmitted messages reach the ~~sentination~~ destination without any problem so they intend to implement different algorithm in order to satisfy this requirement

② What is CRC :— CRC stands for cyclic Redundancy check ~~w~~ which means that is based on cyclic algorithm that generates redundant information.

~~In general redundant information.~~

In general CRC codes are able to detect :—

- All single - and double bit errors
- All odd number of errors
- All burst errors less than or equal to the degree of the polynomial used.

Scanned with CamScanner

- most burst error greater than the degree of the polynomial used.

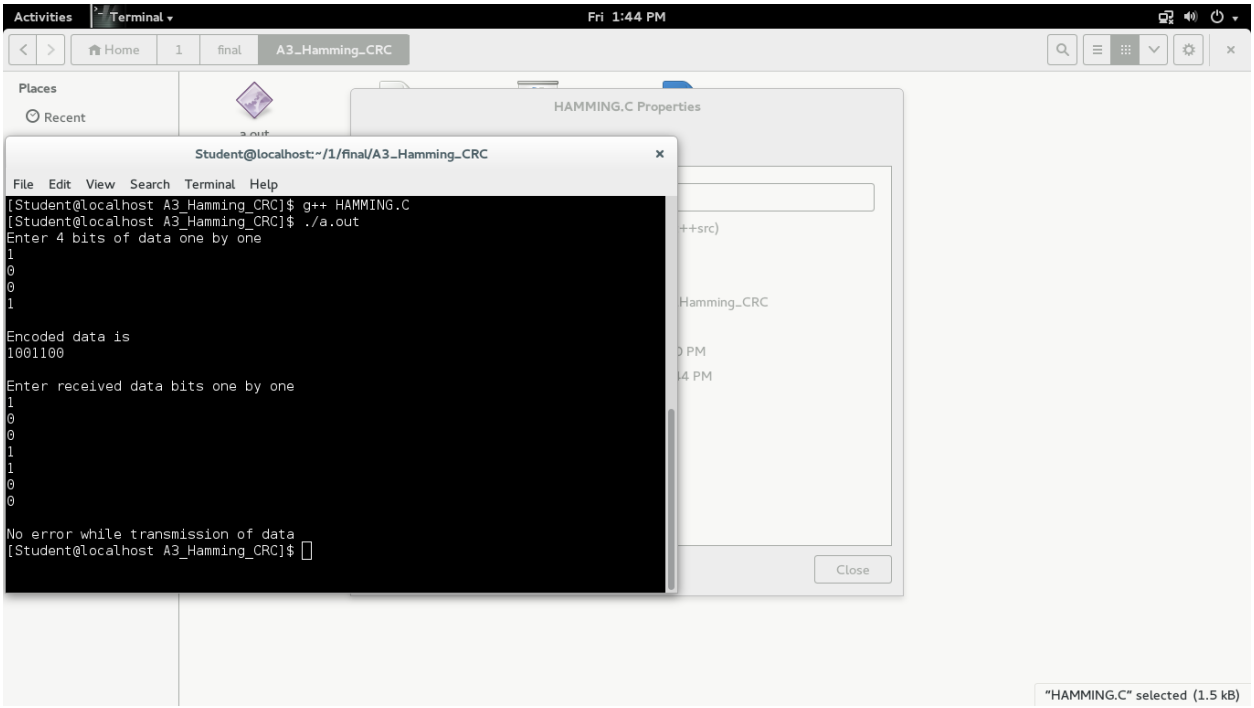
* CRC idea :— The main idea of CRC is to treat the message as binary numbers and divide it by fixed binary number The remainder from this division is considered the checksum The recipient of the message performs the same division and compare the remainder with the "checksum" (transmitted remainder)

* Theory of operation :— As stated in the previous section the CRC is simple binary division and subtraction. The only difference is that these operations are done on modulo arithmetic based on mod 2. For example the addition and substraction are replaced with XOR operation that do the sum and subtraction without carry.

* Conclusion :— Hence we studied and implement program for error detection and correction for 7/8 bits ASCII code using CRC.

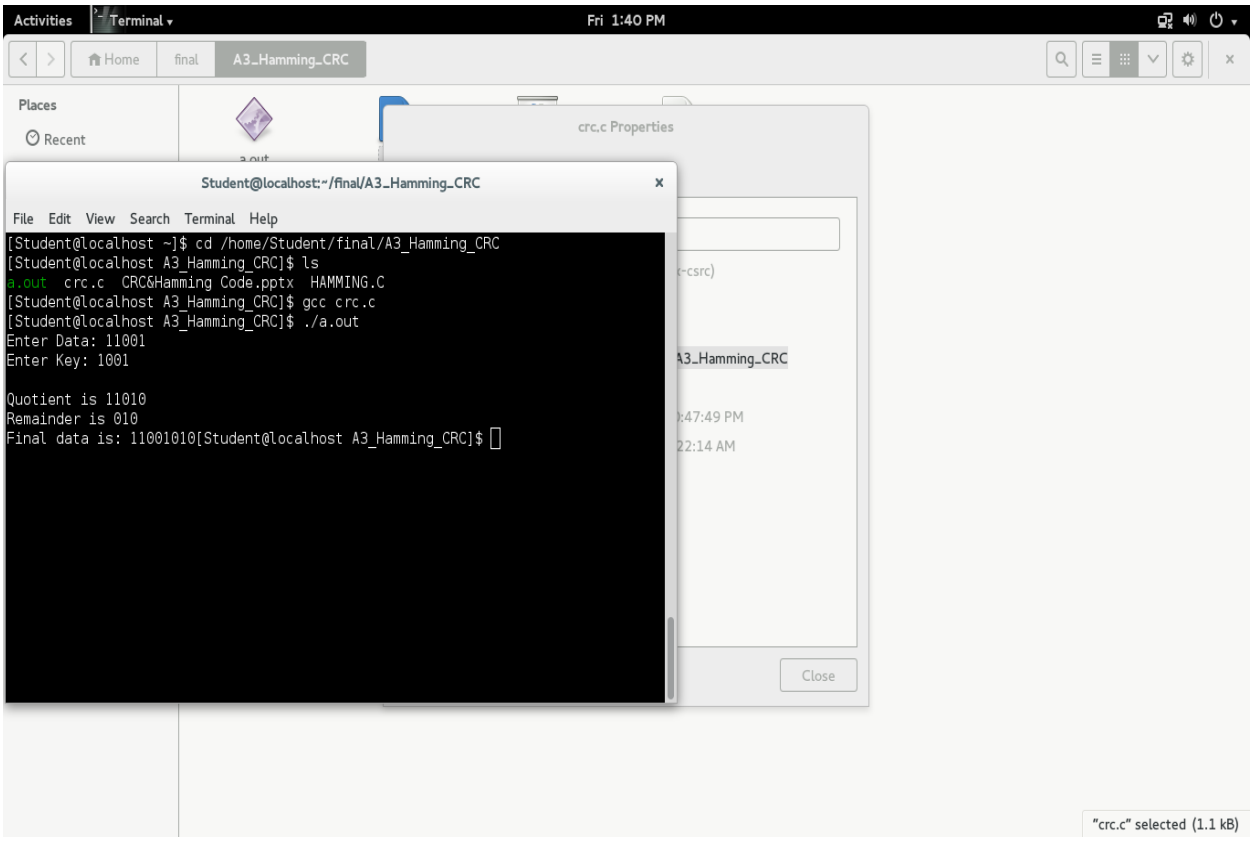
```
/******                A4 CRC          ******/
#include<stdio.h>

 #include <string.h>
void main() {
      int i,j,keylen,msglen;
      char input[100], key[30],temp[30],quot[100],rem[30],key1[30];

      printf("Enter Data: ");
      scanf("%s",input);
      printf("Enter Key: ");
      scanf("%s",key);
      keylen=strlen(key);
      msglen=strlen(input);
      strcpy(key1,key);
      for (i=0;i<keylen-1;i++) {
            input[msglen+i]='0';
      }
      for (i=0;i<keylen;i++)
      temp[i]=input[i];

      for (i=0;i<msglen;i++) {
            quot[i]=temp[0];
            if(quot[i]=='0')
            for (j=0;j<keylen;j++)
            key[j]='0';
else
            for (j=0;j<keylen;j++)
            key[j]=key1[j];
            for (j=keylen-1;j>0;j--) {
                  if(temp[j]==key[j])
                  rem[j-1]='0'; else
                  rem[j-1]='1';
            }
            rem[keylen-1]=input[i+keylen];
            strcpy(temp,rem);
      }
      strcpy(rem,temp);
      printf("\nQuotient is ");
      for (i=0;i<msglen;i++)
      printf("%c",quot[i]);
      printf("\nRemainder is ");
      for (i=0;i<keylen-1;i++)
      printf("%c",rem[i]);
      printf("\nFinal data is: ");
      for (i=0;i<msglen;i++)
      printf("%c",input[i]);
      for (i=0;i<keylen-1;i++)
      printf("%c",rem[i]);


}
```

*** OUTPUT of CRC ***



```
/******                A4 Hamming          ******/


#include<stdio.h>


int main()
{
int data[10];
int dataatrec[10],c,c1,c2,c3,i;

printf("Enter 4 bits of data one by one\n");
scanf("%d",&data[0]);
scanf("%d",&data[1]);
scanf("%d",&data[2]);
scanf("%d",&data[4]);

    //Calculation of even parity
data[6]=data[0]^data[2]^data[4];
data[5]=data[0]^data[1]^data[4];
data[3]=data[0]^data[1]^data[2];

printf("\nEncoded data is\n");
for(i=0;i<7;i++)
printf("%d",data[i]);

printf("\n\nEnter received data bits one by one\n");
for(i=0;i<7;i++)
scanf("%d",&dataatrec[i]);

    c1=dataatrec[6]^dataatrec[4]^dataatrec[2]^dataatrec[0];
    c2=dataatrec[5]^dataatrec[4]^dataatrec[1]^dataatrec[0];
    c3=dataatrec[3]^dataatrec[2]^dataatrec[1]^dataatrec[0];
    c=c3*4+c2*2+c1 ;

if(c==0)
     {
printf("\nNo error while transmission of data\n");
    }
else
     {
printf("\nError on position %d",c);

printf("\nData sent : ");
for(i=0;i<7;i++)
printf("%d",data[i]);

printf("\nData received : ");
for(i=0;i<7;i++)
printf("%d",dataatrec[i]);

printf("\nCorrect message is\n");
```

```
        //if errorneous bit is 0 we complement it else vice versa
if(dataatrec[7-c]==0)
dataatrec[7-c]=1;
else
dataatrec[7-c]=0;

for (i=0;i<7;i++)
           {
printf("%d",dataatrec[i]);


      }
    }
return 0;
}
*** OUTPUT ***
```



Experiment No :- 05.

* Title :— Study of Go Back - N and selective Repeat mode of Sliding Window protocol.

* objective :— ① To understand Go Back N and selective repeat mode of sliding window protocol.

② Designing simple client or server applications for stream.

* Problem statement :- write a program to simulate Go back N and selective Repeat models of sliding windows protocol in peer to peer mode and demonstrate the packets captured traces using wireshark packet analyzer tool per peer to peer mode.

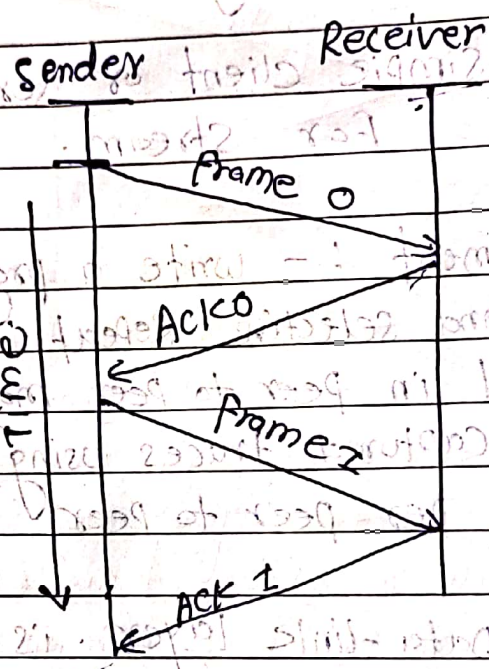
* Theory :— Data-link layer is responsible for implementation of point to point flow and error control mechanism.

* Flow control :— when a data frame (layer-2 data) is sent from one host to another over a single medium it is required that the sender and receiver should work at the same speed. that is sender sends at a speed on which the receiver can process and accept. the data what if the speed (hardware software) of the sender or receiver differ if sender is sending too fast the receiver may be overloaded and data may be lost.

Two types of mechanisms can be deployed to control the flow.

Scanned with CamScanner

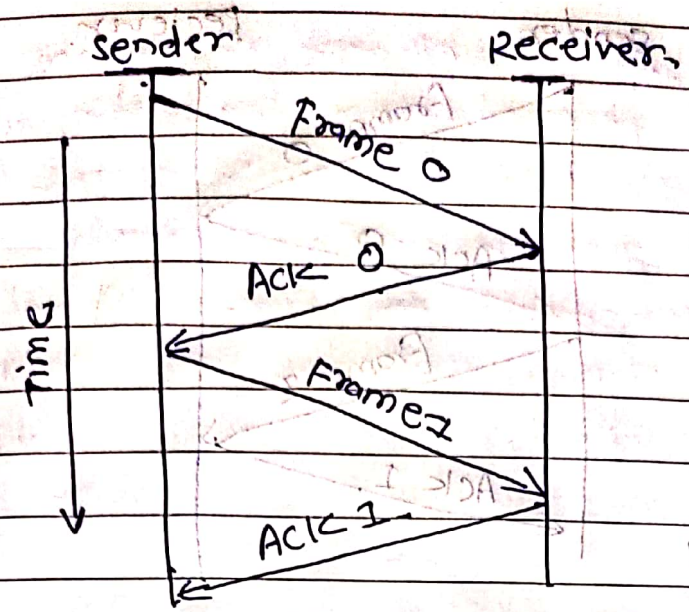
① stop and wait :- This flow control mechanism forces the sender after transmitting a data fram to stop and wait until the acknowledge of the data-frame sent is received.



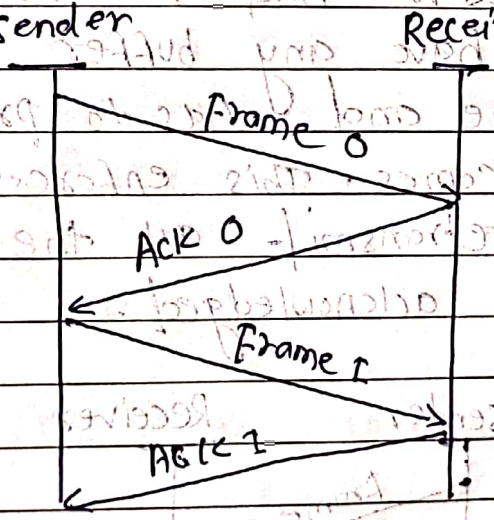
② stop-and-wait ARQ

- The following transmition may occur in stop-and-wait ARQ.
- The sender maintains a timeout counter.
- When a frame is sent the sender starts the timeout counter.
- If acknowledgement of frame comes in time the sender transmits the next frame in queue.

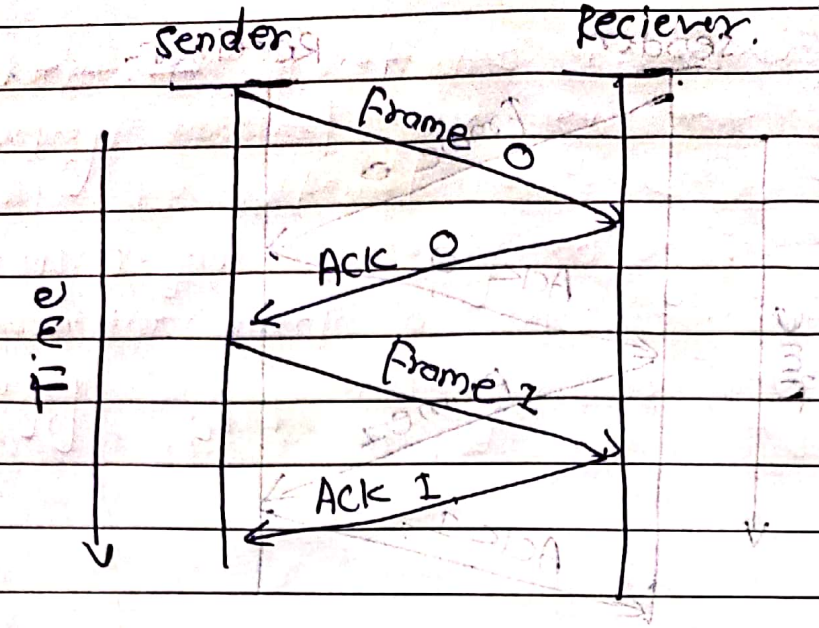
Scanned with CamScanner



③ Go-Back-N ARQ.

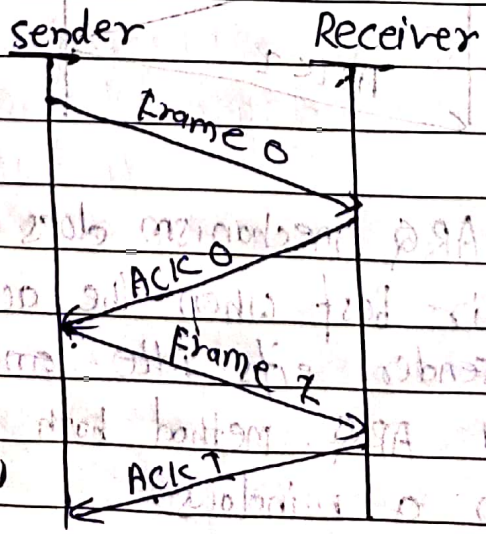


Stop and wait ARQ mechanism does not utilize the resources at their best when the acknowledgment is received the sender sits idle and does nothing In Go-Back-N ARQ. method both sender and receiver maintain a windows.



④ selective Repeat ARQ. :-

In Go-back N. ARQ it is assumed that the receiver does not have any buffer space for its window size and has to process each frame as it comes. This enforces the sender to retransmit all the frames which are not acknowledged.



Conclusion :- Hence we studied and implement program Go-back n and selective repeat mode of sliding window protocol.

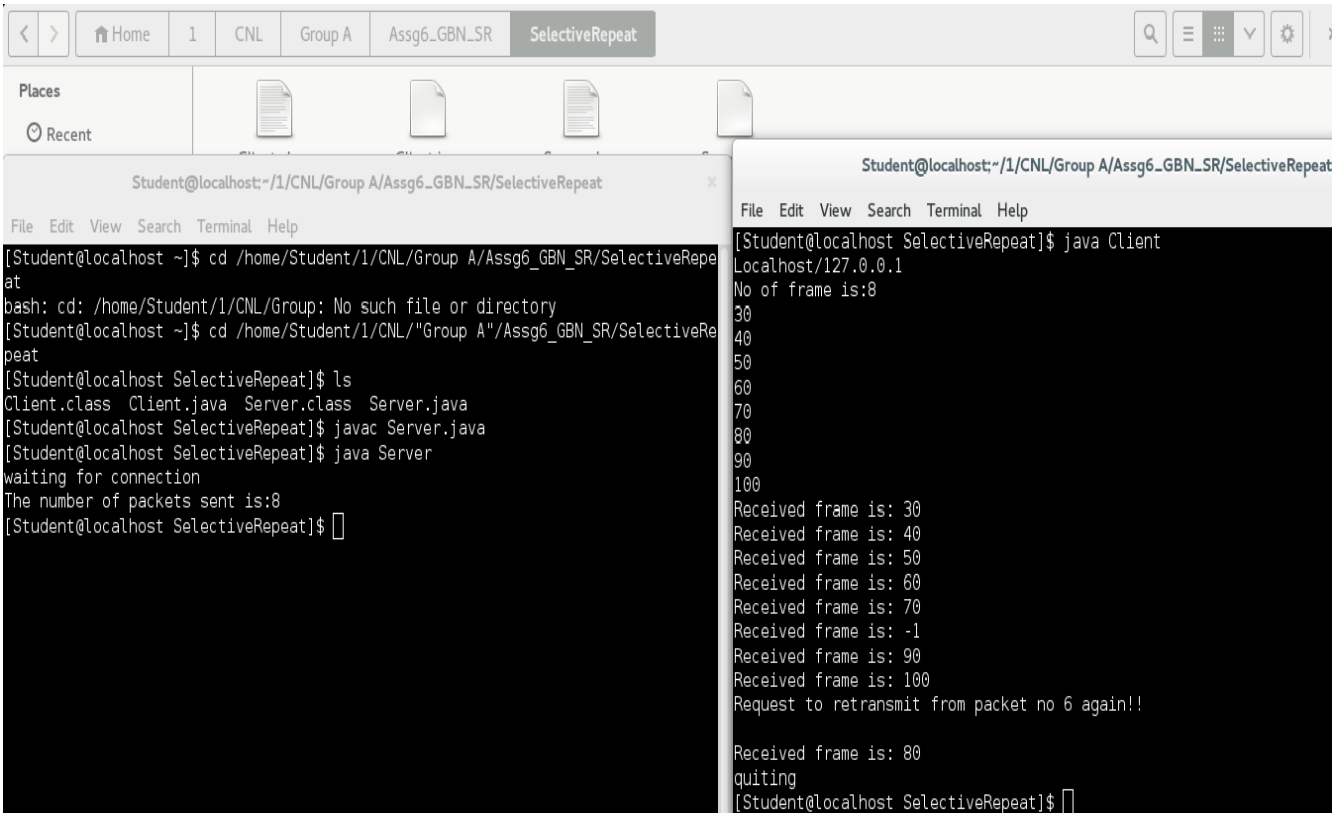
Scanned with CamScanner

```
/***** A5.selective repeat  *****/


//Client

import java.lang.System;
import java.net.*;
import java.io.*;

public class Client {
     static Socket connection;

     public static void main(String a[]) throws SocketException {
          try {
               int v[] = new int[8];
               //int g[] = new int[8];
               int n = 0;
               InetAddress addr = InetAddress.getByName("Localhost");
               System.out.println(addr);
               connection = new Socket(addr, 8011);
               DataOutputStream out = new DataOutputStream(
                         connection.getOutputStream());
               DataInputStream in = new DataInputStream(
                         connection.getInputStream());
               int p = in.read();
               System.out.println("No of frame is:" + p);

               for (int i = 0; i < p; i++) {
                    v[i] = in.read();
                    System.out.println(v[i]);
                    //g[i] = v[i];
               }
               v[5] = -1;
               for (int i = 0; i < p; i++)
                {
                         System.out.println("Received frame is: " +
v[i]);

                    }
               for (int i = 0; i < p; i++)
                    if (v[i] == -1) {
                         System.out.println("Request to retransmit
from packet no "
                                   + (i+1) + " again!!");
                         n = i;
                         out.write(n);
                         out.flush();
                    }

               System.out.println();

                    v[n] = in.read();
```

```
                        System.out.println("Received frame is: " + v[n]);


                    System.out.println("quiting");
                } catch (Exception e) {
                    System.out.println(e);
                }

        }
}
```

```
//Server

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.net.SocketException;

public class Server {
	static ServerSocket Serversocket;
	static DataInputStream dis;
	static DataOutputStream dos;

	public static void main(String[] args) throws SocketException {

		try {
			int a[] = { 30, 40, 50, 60, 70, 80, 90, 100 };
			Serversocket = new ServerSocket(8011);
			System.out.println("waiting for connection");
			Socket client = Serversocket.accept();
			dis = new DataInputStream(client.getInputStream());
			dos = new DataOutputStream(client.getOutputStream());
			System.out.println("The number of packets sent is:" +
a.length);
			int y = a.length;
			dos.write(y);
			dos.flush();

			for (int i = 0; i < a.length; i++) {
				dos.write(a[i]);
				dos.flush();
			}

			int k = dis.read();

			dos.write(a[k]);
			dos.flush();

		} catch (IOException e) {
			System.out.println(e);
		} finally {
			try {
				dis.close();
				dos.close();
			} catch (IOException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
```

```
                    }
                }
            }
}
```


Home
1
CNL
Group A
Assg6_GBN_SR
SelectiveRepeat
Places
Recent
Student@localhost:~/1/CNL/Group A/Assg6_GBN_SR/SelectiveRepeat
File Edit View Search Terminal Help
[Student@localhost ~]$ cd /home/Student/1/CNL/Group A/Assg6_GBN_SR/SelectiveRepeat
bash: cd: /home/Student/1/CNL/Group: No such file or directory
[Student@localhost ~]$ cd /home/Student/1/CNL/"Group A"/Assg6_GBN_SR/SelectiveRepeat
[Student@localhost SelectiveRepeat]$ ls
Client.class Client.java Server.class Server.java
[Student@localhost SelectiveRepeat]$ javac Server.java
[Student@localhost SelectiveRepeat]$ java Server
waiting for connection
The number of packets sent is:8
[Student@localhost SelectiveRepeat]$
Student@localhost:~/1/CNL/Group A/Assg6_GBN_SR/SelectiveRepeat
File Edit View Search Terminal Help
[Student@localhost SelectiveRepeat]$ java Client
Localhost/127.0.0.1
No of frame is:8
30
40
50
60
70
80
90
100
Received frame is: 30
Received frame is: 40
Received frame is: 50
Received frame is: 60
Received frame is: 70
Received frame is: -1
Received frame is: 90
Received frame is: 100
Request to retransmit from packet no 6 again!!
Received frame is: 80
quiting
[Student@localhost SelectiveRepeat]$

```
/***** A5.sliding window go back N*****/

//Client

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

class testclient
{

public static void main(String args[])throws IOException
{
InetAddress addr=InetAddress.getByName("Localhost");
System.out.println(addr);

Socket connection=new Socket(addr,5000);

BufferedInputStream in=new
BufferedInputStream(connection.getInputStream());
DataOutputStream out=new
DataOutputStream(connection.getOutputStream());
Scanner scr=new Scanner(System.in);// this will be used to accept i/p
from console

System.out.println(" client is Connected to server" + addr);
System.out.println("Enter the number of frames to be requested to the
server");
int c=scr.nextInt();

out.write(c); // write no of frames on client socket
out.flush();

System.out.println("Enter the type of trans. Error=1 ; No Error=0");
int choice=scr.nextInt();
out.write(choice);  //write choice on socket

int check=0;
int i=0;
int j=0;

if(choice==0)
{
for(j=0;j<c;++j)
{
i=in.read();  //read all frames one by one from server
System.out.println("received frame no: "+i);
System.out.println("Sending acknowledgement for frame no: "+i);
out.write(i); //write ack  to socket
```

```
out.flush();
}
out.flush();
}
else
{
for(j=0;j<c;++j)
{
i=in.read();  //read 0,1,2,3 frame
if(i==check)
{
System.out.println("received frame no: "+i);
System.out.println("Sending acknowledgement for frame no: "+i);
out.write(i); //sent ack of frame 0,1
++check;
}
else
{
--j;
System.out.println("Discarded frame no: "+i);
System.out.println("Sending NEGATIVE ack");
out.write(-1);
}
out.flush();
}
}//end of else for error

in.close();
out.close();
System.out.println("Quiting");

}// end of main method
}// end of main class
```

```
/***** A5.sliding window go back N *****/

//server

import java.io.*;
import java.net.*;
import java.util.*;
class testserver
{
public static void main(String args[])throws IOException
{
System.out.println("server Waiting for connection....");
InetAddress addr=InetAddress.getByName("Localhost");
ServerSocket ss=new ServerSocket(5000);

Socket client=new Socket();
client=ss.accept();

BufferedInputStream in=new
BufferedInputStream(client.getInputStream());
DataOutputStream out=new DataOutputStream(client.getOutputStream());

System.out.println("Received request for sending frames");
int p=in.read();  //read no of frames sent by client

boolean f[]=new boolean[p];

int pc=in.read();  //read choice sent by client
System.out.println("Sending....");

if(pc==0)
{
for(int i=0;i<p;++i)
{
System.out.println("sending frame number "+i);
out.write(i); //send frame on server socket
out.flush();
System.out.println("Waiting for acknowledgement");
try
{
Thread.sleep(7000);
}
catch(Exception e){}

int a=in.read(); //read ack on servers socket from client
System.out.println("received acknowledgement for frame "+i+" as "+a);
}
out.flush();
```

```
}
else
{
for(int i=0;i<p;++i)
{
if(i==2)
{
System.out.println("sending frame no "+i);  //sent frame 2
}
else
{
System.out.println("sending frame no "+i);
out.write(i);  //write 0 and 1 and 3 frame
out.flush();
System.out.println("Waiting for acknowledgement ");
try
{
Thread.sleep(7000);
}
catch(Exception e){}

int a=in.read(); //Read NACK

if(a!=255)
{
System.out.println("received ack for frame no: "+i+" as "+a);
f[i]=true;
}
}// end of inner else
}// end of for

// check which frames have not been ack

for(int a=0;a<p;++a)
{
if(f[a]==false)
{
System.out.println("Resending frame "+a);
out.write(a);
out.flush();
System.out.println("Waiting for ack ");
try
{
Thread.sleep(5000);
}
catch(Exception e){}

int b=in.read();
System.out.println("received ack for frame no: "+a+" as "+b);
f[a]=true;
}
}
```

```
out.flush();
}// end of else which is for error

in.close();
out.close();
client.close();
ss.close();
System.out.println("Quiting");

}// end main method
}// end main class
```

Experiement No :- 6

*Title :- study of IP Address and subnetting.

*Objective :-

① To understand the structure of IP addresses and subnet mask.

② To understand the concept of subnetting and create subnet of given IP Address.

* problem statement :- write a program to demonstrate public and find the subnet mask.

* Theory :- Introduction :- If definition are helpful to you use these vocabilary terms in order to get ~~your~~ you started.

* Address :- The unique number ID assigned to one host or interface in a network.

- Subnet :- A portion of a network that shares a particular subnet Address.

- Subnet mask :- A 32-bit combination used to describe which portion of an address refers to the subnet and which part returns to the host.

Scanned with CamScanner

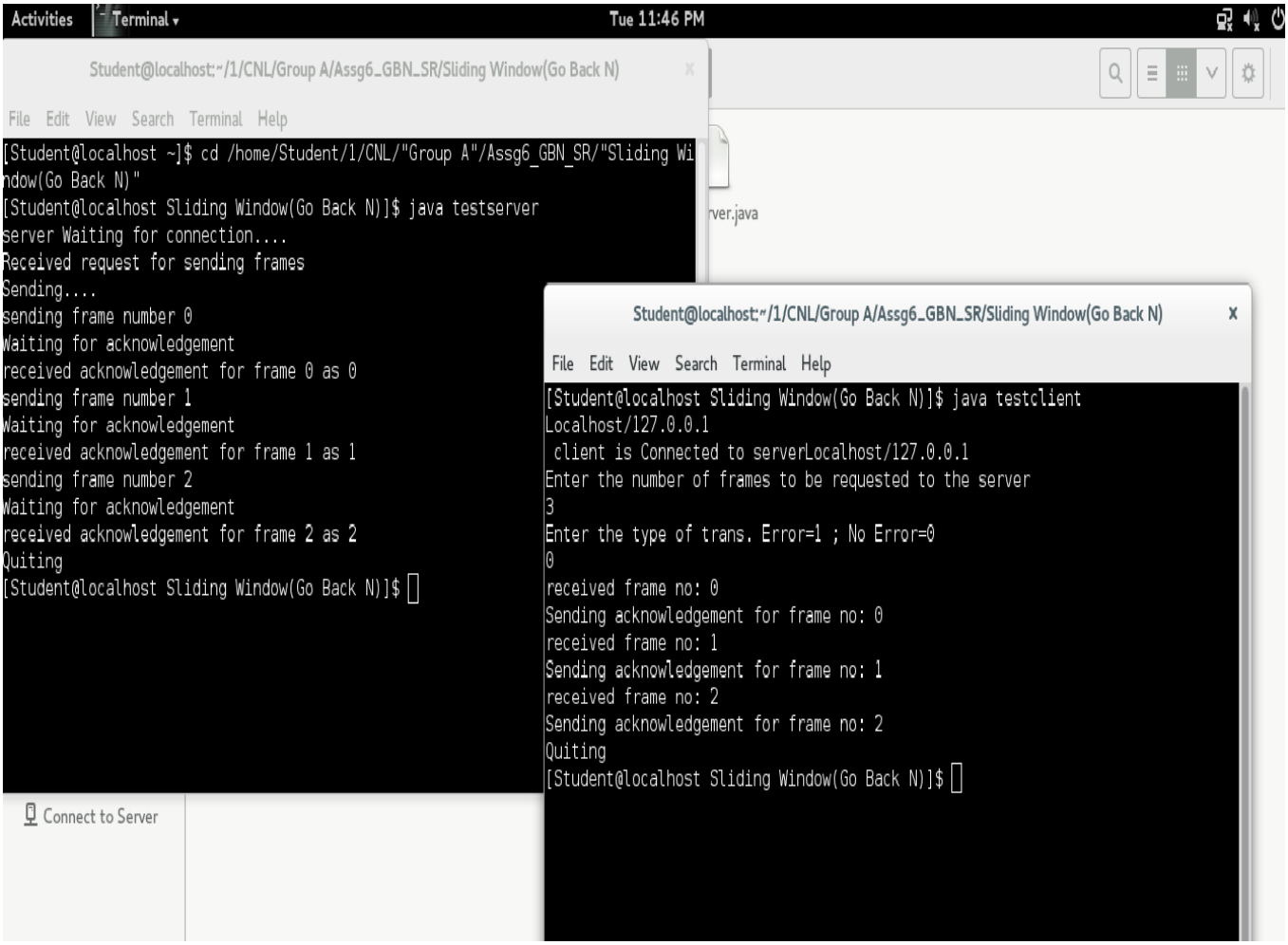
Interface :- A network connection if you have already received your legitimate address from the Internet network Information Center (InterNIC) you are ready to begin. If you do not plan to connect to the Internet Cisco strongly suggest that you use reserved addresses from RFC 1918.

* Understand IP address :- An IP address is an address used in order to uniquely identify a device on an IP network. The address is mode up of 32 binary bits, which can be divisible into a network portion and part portion with the help of a subnet mask. The 32 binary bits are broken into four octets (1 octet = 8 bits)

* understand subnetting :- Subnetting allows you to create multiple logical network that exits within a single class A, B, or C network. If you do not subnet, you are only able to use one network from your class A, B, or C network which is unrealistic.

* Conclusion :- Hence we studied and implement program to demonstrate subnetting & find the subnet-mask.

Scanned with CamScanner


Activities
Terminal
Tue 11:46 PM
Student@localhost:~/1/CNL/Group A/Assg6_GBN_SR/Sliding Window(Go Back N)
File Edit View Search Terminal Help
[Student@localhost ~]$ cd /home/Student/1/CNL/"Group A"/Assg6_GBN_SR/"Sliding Wi
ndow(Go Back N)"
[Student@localhost Sliding Window(Go Back N)]$ java testserver
server Waiting for connection....
Received request for sending frames
Sending....
sending frame number 0
Waiting for acknowledgement
received acknowledgement for frame 0 as 0
sending frame number 1
Waiting for acknowledgement
received acknowledgement for frame 1 as 1
sending frame number 2
Waiting for acknowledgement
received acknowledgement for frame 2 as 2
Quiting
[Student@localhost Sliding Window(Go Back N)]$
rver.java
Student@localhost:~/1/CNL/Group A/Assg6_GBN_SR/Sliding Window(Go Back N)
File Edit View Search Terminal Help
[Student@localhost Sliding Window(Go Back N)]$ java testclient
Localhost/127.0.0.1
client is Connected to serverLocalhost/127.0.0.1
Enter the number of frames to be requested to the server
3
Enter the type of trans. Error=1 ; No Error=0
0
received frame no: 0
Sending acknowledgement for frame no: 0
received frame no: 1
Sending acknowledgement for frame no: 1
received frame no: 2
Sending acknowledgement for frame no: 2
Quiting
[Student@localhost Sliding Window(Go Back N)]$
Connect to Server

```
/***** B1.Subneting  *****/

import java.util.Scanner;
class Subnet{
public static void main(String args[])
{
Scanner sc = new Scanner(System.in);
System.out.print("Enter the ip address: ");
String ip = sc.nextLine();
String split_ip[] = ip.split("\\."); //SPlit the string after every .
String split_bip[] = new String[4]; //split binary ip
String bip = "";
String mask1="";

int firstoctet = Integer.parseInt(split_ip[0]);
if(firstoctet<=127)
                                {
                                                mask1 = "255.0.0.0";
System.out.println("Class A IP Address");
System.out.println("Default mask is: "+mask1);
                                }
else if(firstoctet>=128 && firstoctet<=191)
                                {
                                                mask1 = "255.255.0.0";
System.out.println("Class B IP Address");
System.out.println("Default mask is: "+mask1);
                                }
else if(firstoctet>=192 && firstoctet<=223)
                                {
                                                mask1 =
"255.255.255.0";
System.out.println("Class C IP Address");
System.out.println("Defaultt mask is: "+mask1);
                                }
for(int i=0;i<4;i++)
{
split_bip[i] =
appendZeros(Integer.toBinaryString(Integer.parseInt(split_ip[i]))); //
"18" => 18 => 10010 => 00010010
bip += split_bip[i];
}
System.out.println("IP in binary is "+bip);
System.out.print("Enter the number of subnets: ");
int n = sc.nextInt();

//Calculation of mask
double bits = (int)Math.ceil(Math.log(n)/Math.log(2)); /*eg if address
= 120, log 120/log 2 gives log to the base 2 => 6.9068, ceil gives us
upper integer */
int y=(int) bits;
System.out.println("Number of bits borrowd from host to network are =
"+y);
```

Experiment No 08

Title :- Study of RIP, OSPF and BGP using Packet Tracer

Objective :-
- Student should be able to understand and design different Network devices such machine router, switches using Packet Tracer.

Problem Statement :-

Configure RIP / OSPF / BGP using Packet Tracer

Theory :- Packet Tracer is a powerful network simulator that can be utilized in training for CCNA and CCNP certification exam by allowing student to create network with an almost unlimited number of devices and to experience troubleshooting without having to buy real Cisco routers or switches. The tool is created by Cisco System The purpose of Packet tracer is to offer student a tool to learn the principals of networking as well as develop Cisco technology specific skill. However it is not be used as a replacement for Routers or Switches.

Now we will have to connect these devices and for that we use Cables. To understand the Cables, we have connect the devices further we will go to the router CLI mode and enter the following commands step by step we will have to do the Following things.

Scanned with CamScanner

i) Access the interfaces one by one.
ii) Assign IP addresses to interfaces.
iii) change the status of the interfaces i.e. from Down to up
iv) Assign IP addresses to PC,s
v) Assign Default Gateway to Pcs. FYI fast ethernet address is the getway address to the PC. NOW Commonds of the Router CLI mode are as follows.

Conclusion :— Hence we studied and Configure RIP OSPF and BGP using tracer.

Scanned with CamScanner

Experiment No :- 10

Title :- Study of UDP socket programming for wired network.

Objective :- ① Getting Familiar with the client-server communication model.

② Learning the most important library function (the UNIX and Internet socket used for the design of the client-server application.

③ Designing simple client or server application for datagram.

* Problem statement :- Write a program using UDP sockets to cable file transfer (Script, text, Audio and video one file each) between two machine, Demonstrate the packets cap using Wireshark Packet Analyzer tool for peer to peer mode.

* Theory :-

Program :- A program is an executable file residing on a disk in a directory A program is read into memory and is executed by the kernel as result of an exec() function. The exec() how six variants but we only consider the simplest one (execl()) in this course.

Scanned with CamScanner

* Process :- An executing instance of a program is called a process sometime task is used instead of process with the same meaning. UNIX go that every process has a unique identifier called the process ID The process ID is always non-negative integer.

* File descriptor :- file descripter are normally small non-negative integer that the kernal used to identify the files being accessed by a particular process.

* The client-server model :-

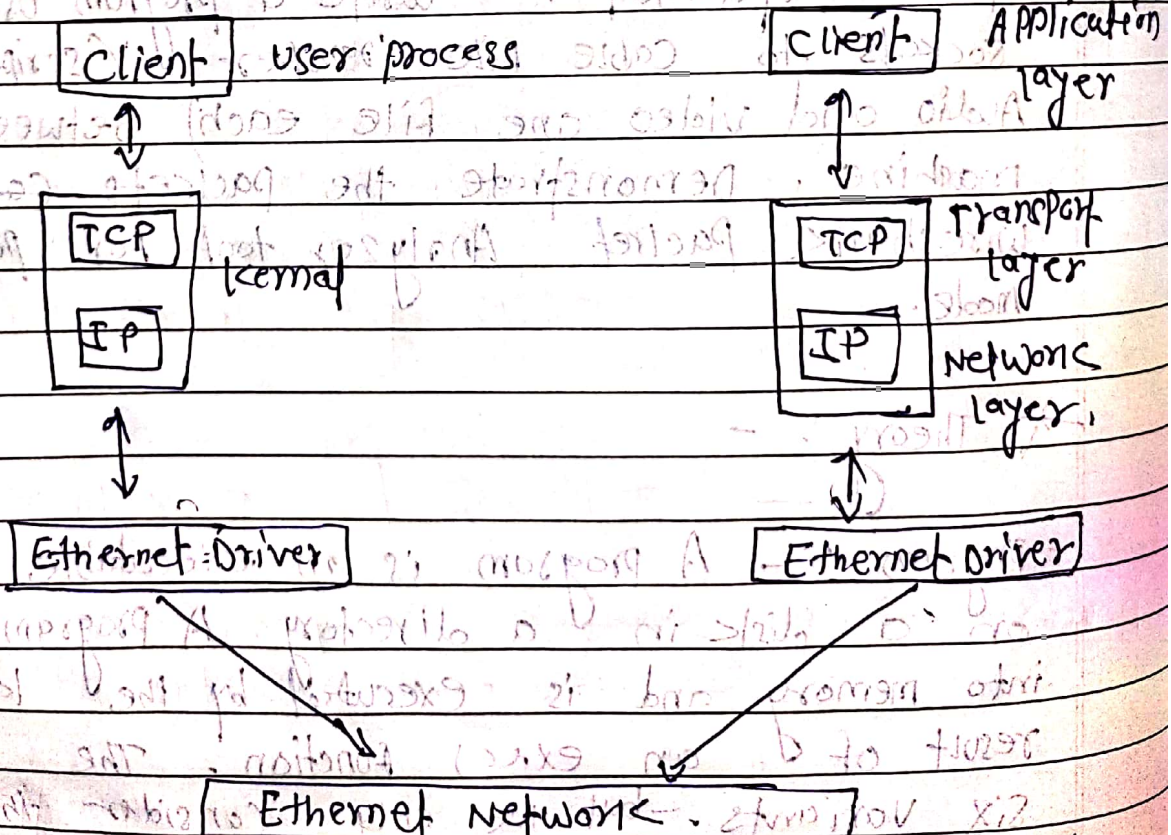


Fig :- Client and server on the same Ethernet communicating using TCP/IP

Scanned with CamScanner

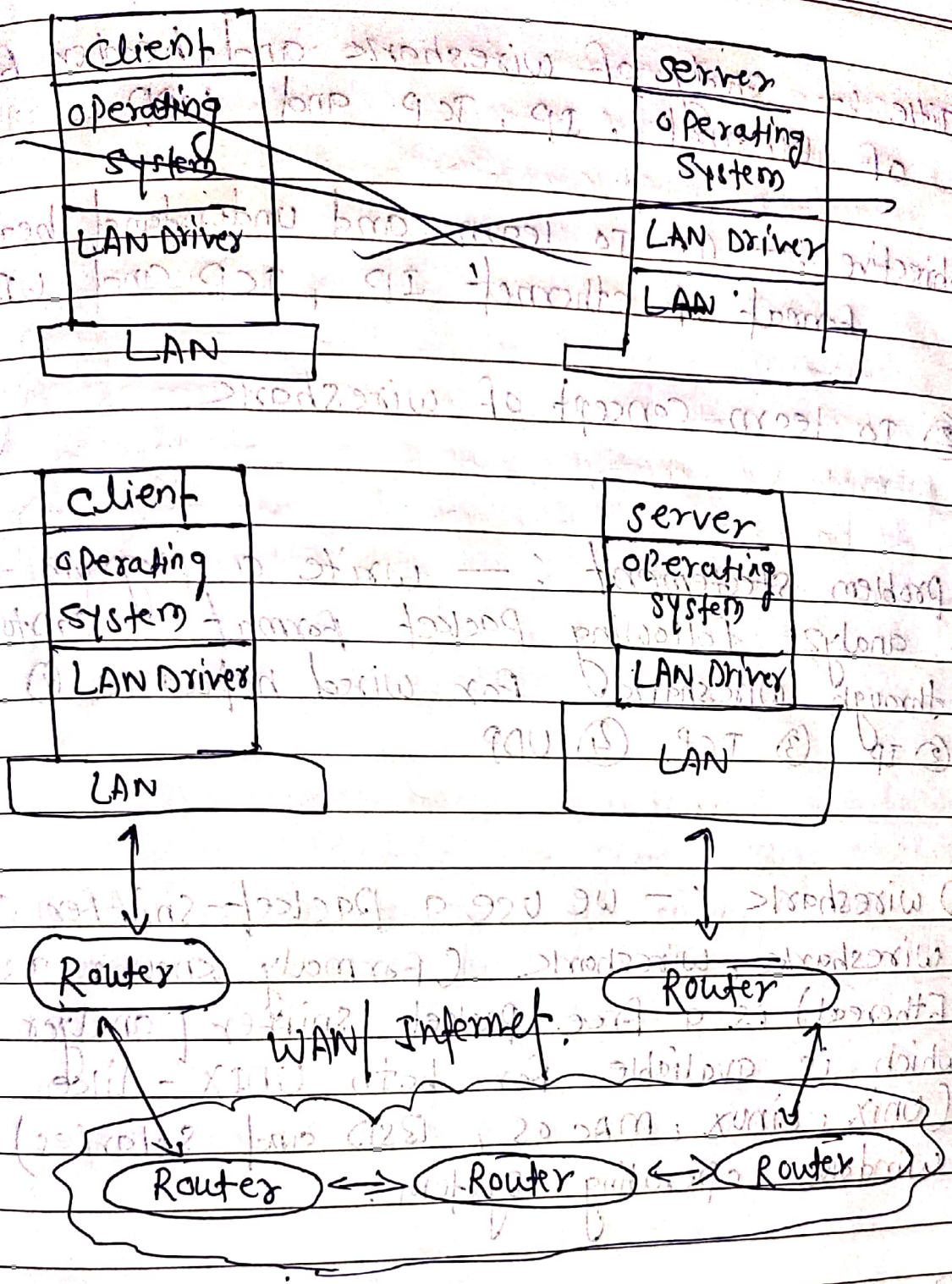


Fig :- client and server different LAN connected through WAN Internet.

Conclusion :- Hence we studied and implement program to demonstrate UCP socket programming for wired network.

Scanned with CamScanner

/*** B10. UDP Socket client ***/

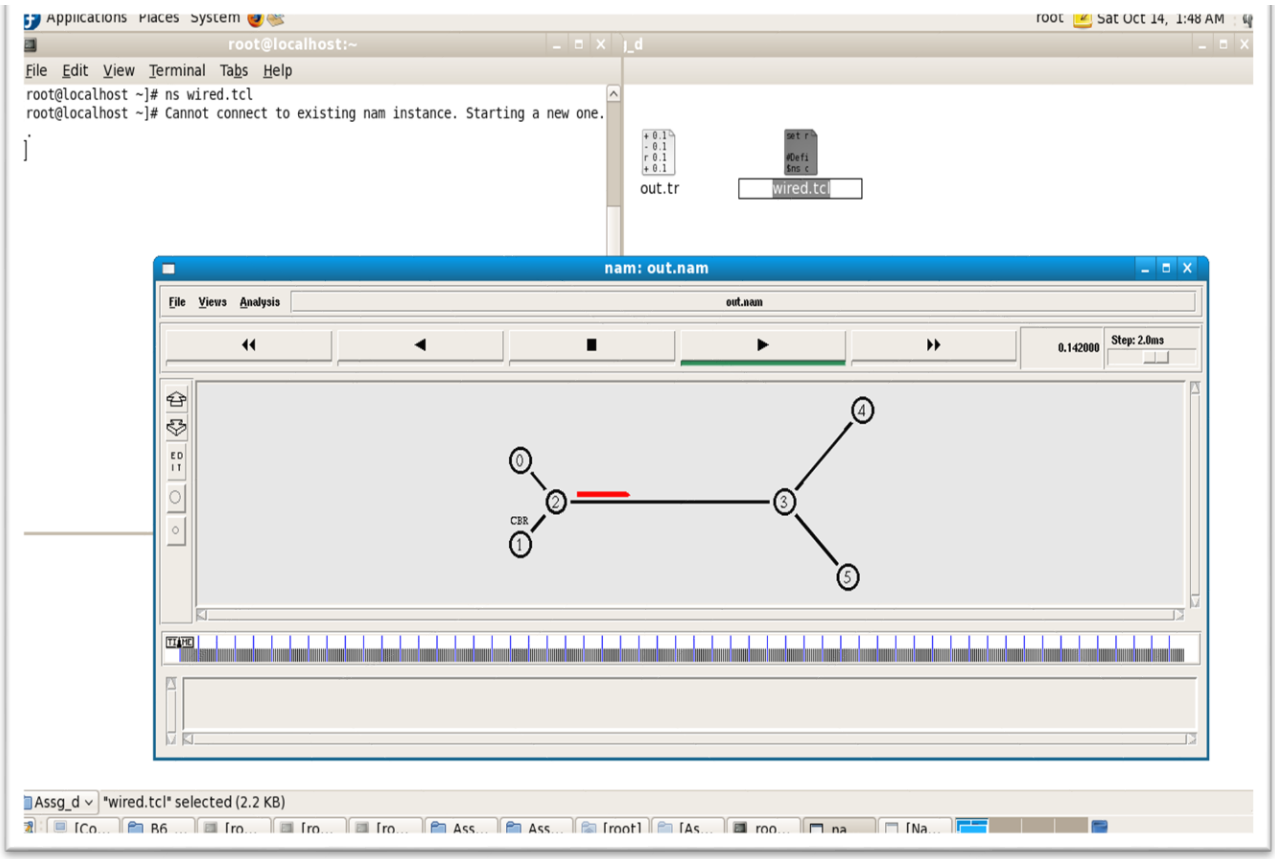
```
import java.io.BufferedOutputStream;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.net.InetAddress;
import java.net.Socket;


public class FileTransferClient {

public static void main(String[] args) throws Exception{

        //Initialize socket
        Socket socket = new Socket(InetAddress.getByName("localhost"),
5000);
byte[] contents = new byte[10000];

        //Initialize the FileOutputStream to the output file's full
path.
        FileOutputStream fos = new
FileOutputStream("/home/Student/1/final/A8_udp_socket/file2.txt");
        BufferedOutputStream bos = new BufferedOutputStream(fos);
        InputStream is = socket.getInputStream();

        //No of bytes read in one read() call
int bytesRead = 0;

while((bytesRead=is.read(contents))!=-1)
bos.write(contents, 0, bytesRead);

bos.flush();
socket.close();

System.out.println("File saved successfully!");
    }
}
```

/*** B10. UDP socket server ***/

```
import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.OutputStream;
import java.net.InetAddress;
import java.net.ServerSocket;
import java.net.Socket;

public class FileTransferServer {

public static void main(String[] args) throws Exception {
        //Initialize Sockets
        ServerSocket ssock = new ServerSocket(5000);
        Socket socket = ssock.accept();

        //The InetAddress specification
        InetAddress IA = InetAddress.getByName("localhost");

        //Specify the file
        File file = new
File("/home/Student/1/final/A8_udp_socket/file.txt");
        FileInputStream fis = new FileInputStream(file);
        BufferedInputStream bis = new BufferedInputStream(fis);

        //Get socket's output stream
        OutputStream os = socket.getOutputStream();

        //Read File Contents into contents array
byte[] contents;
long fileLength = file.length();
long current = 0;

long start = System.nanoTime();
while(current!=fileLength){
int size = 10000;
if(fileLength - current >= size)
current += size;
else{
size = (int)(fileLength - current);
current = fileLength;
            }
contents = new byte[size];
bis.read(contents, 0, size);
os.write(contents);
System.out.print("Sending file ... "+(current*100)/fileLength+"%
complete!");
        }

os.flush();
        //File transfer done. Close the socket connection!
```

```
socket.close();
ssock.close();
System.out.println("File sent succesfully!");
    }
}
```

*** OUTPUT***


Applications Places System
root@localhost:~
File Edit View Terminal Tabs Help
root@localhost ~]# ns wired.tcl
root@localhost ~]# Cannot connect to existing nam instance. Starting a new one.
out.tr
wired.tcl
nam: out.nam
File Views Analysis
out.nam
Step: 2.0ms
CBR
Assg_d
"wired.tcl" selected (2.2 KB)

Experiment No :- 03.

* Title :- Study of wireshark and header format of Ethernet, IP, TCP and UDP

* Objective :- ① To learn and understand header format of ethernet IP, TCP and UDP

② To learn concept of wireshark

* Problem statement :- write a program to analyze following packet format captured through wireshark for wired network ① Ethernet ② IP ③ TCP ④ UDP

① wireshark :- we use a packet-sniffer called wireshark, wireshark (formaly knowns as Ethereal) is a free packet sniffer / analyzer which is avaliable for both UNIX-like (Unix, Linux, MAC OS, BSD and Solaries) and windows operating system.

② Downloading and Installing :-

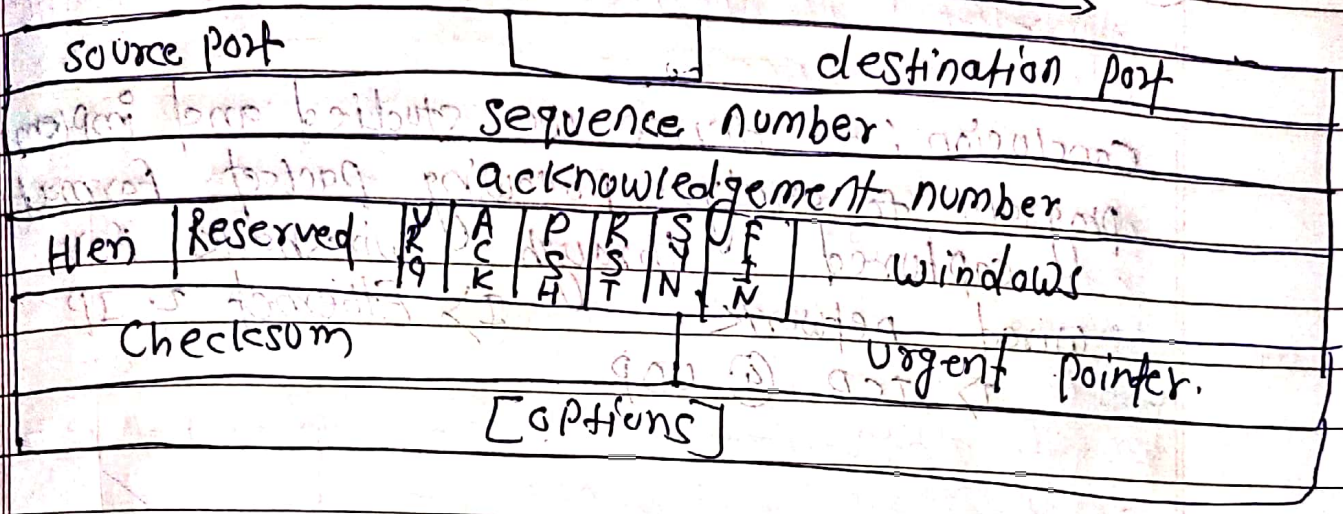
To Download the wireshark software connect to the Internet using website:

http://www.wireshark.org/download/

Scanned with CamScanner

* TCP Header format :-

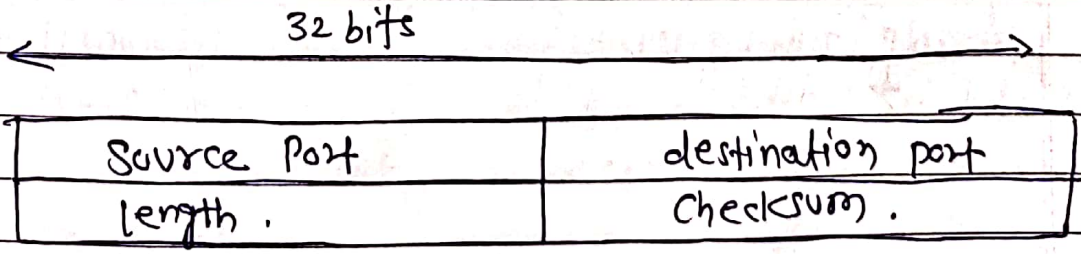
TCP Header format



Each TCP header has ten required fields totaling 20 bytes (160 bits) in size. They can also optionally includes an additional data section up to 40 bytes in size.

* UDP Header format :-

UDP header format



Because UDP is significantly more limited in capability than TCP, its headers are much smaller. A UDP header contain 8 bytes, divided into the following four required fields.

Scanned with CamScanner

① Source port number (2 bytes)
② Destination port number (2 bytes)
③ Length of data (2 bytes)
④ UDP checksum (2 bytes)

Conclusion :— Hence we studied and implemented program to analyze following packet formats captured through wireshark for wired network : I> Ethernet 2 IP I> TCP ④ UDP

Experiment No 11

Title :- Study of DNS Lookup

Objective :-

1. To learn and understand DNS lookup
2. To learn and understand concept of IP protocol.

Problem statement :-

Write a program of DNS lookup. Given an IP address input it should return URL and vice-versa.

Theory :-

What is DNS ?

- The Domain name system (DNS) is the phonbook of the Internet. Humans access information online through domain name like nytimes.com or espn.com web browsers interact through Internet protocol (IP) address. DNS translate domain names to "HYPERLINK "https://www.cloudflare.com/learning/dns/glossary/what-is-my-ip-address/" \h IP address so browsers can load Internet resources.

Each device connected to the internet to the internet has a unique IP address which other machines use to find devices. DNS servers eliminate the need for humans to memorize IP addresses such as 192.168.1.1 (in IPv4) or more complex newer alphanumeric IP addresses such as 2400:cb00:2048

Scanned with CamScanner

* DNS and Active Directory :- Windows Server 2003 Active Directory services uses DNS as it domain controller location mechanism when any of the principal Active Directory operation is performed, such as authentication updating or searching window server 2003 computer use DNS to locate Active Directory domain controllers and these DNS to locate each other.

* DNS Architecture :- DNS Architecture is a hierarchical distributed database and an associated set of protocols that define:

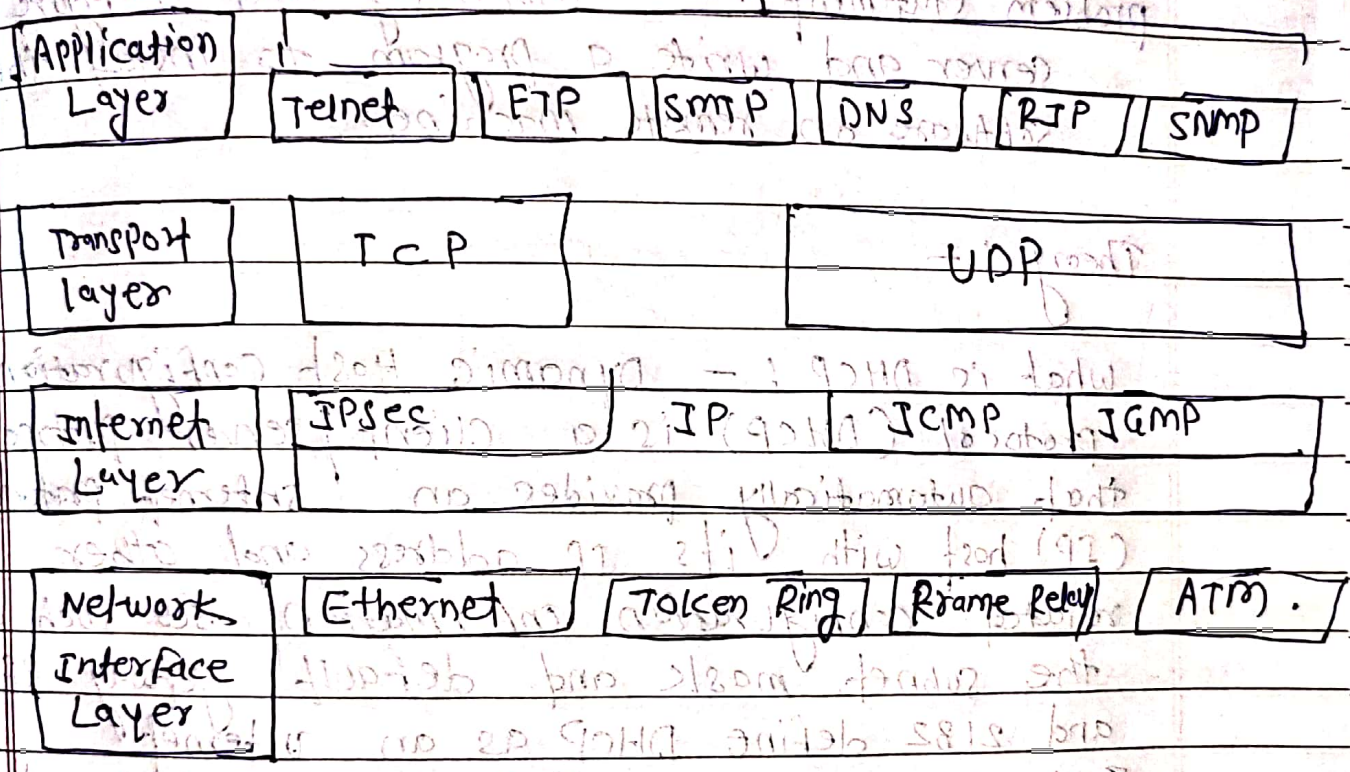
- A mechanism for querying and updating the database
- A mechanism for replicating the information in the database among servers.
- A schema of the database.

DNS in TCP/IP

— x —

TCP/IP model

TCP/IP Protocol Suite



Conclusion :- Hence we studied DNS in detail.

Experiment NO 12

Title :- study of DHCP

objective :- ① student should to be able to understand Dynamic Host configuration protocol

problem statement :- Installing and configure DHCP server and write a program to install the software on remote machine.

Theory :-

What is DHCP :- Dynamic Host configuration protocol (DHCP) is a client/server protocol that automatically provides an Internet protocol (IP) host with its IP address and other related configuration information such as the subnet mask and default gateway RFCs 2131 and 2132 define DHCP as an Internet Engineering Task force (IETF) standard based on Bootstrap protocol (BOOTP), a protocol with which DHCP shares many implementation details DHCP allows hosts to obtain necessary TCP/IP configuration information from a DHCP server.

The microsoft windows server 2003 operating system includes a DHCP server service which is an optional networking components All windows-based client include the DHCP client as part of TCP/IP.

Scanned with CamScanner

In windows Server 2003 the DHCP server service provide the following benefits.

- Reliable IP address configuration :- DHCP minimizes configuration error caused by manual IP address configuration such as typographical error, or address conflicts caused by the assignment of an IP address to more than one computer at the same time.

- Reduced network administration :- DHCP includes the following features to reduce network administration.

  - centralized and automated TCP/IP configuration
  - The ability to define TCP/IP configuration from a central local
  - The ability to assign a full range of additional TCP/IP configuration values by means of DHCP options.
  - The efficient handling of IP address changes for clients that must be updated frequently such as those for portable computers that move to different location on a wireless network.

- The Forwarding of initial DHCP message by using a DHCP relay agent thus eliminating the need to have a DHCP server on every subnet.

Scanned with CamScanner

Why use DHCP :- Every device on a TCP/IP -base network must have a unique unicast IP address to access the network and its resources. Without DHCP IP address must be configured manually for new computers or computers that are moved from one subnet to another and manually reclaimed for computer that are removed from the network.

- Valid TCP/IP configuration parameters for all clients on the network.
- Valid IP addresses, maintained in a pool for assignment to client as well as excluded address.

Conclusion :- Hence we studied DHCP in details.

Scanned with CamScanner

Experiment No 13

Title :- Study of TCP socket programming for wired network.

Objective :-

1) Getting familiar with the client-server communication model.
2) Learning the most important Library function (the UNIX and internet sockets) used for the design of the client-server application.
3) Designing simple client or server application for stream.

Problem Statement :-

write a ~~programming~~ program using TCP sockets for wired network to implement.

a) peer to peer chat
b) multiuser chat

Demonstrate the packets captured traces using wireshark packet Analyzer tool for peer to peer mode

Theory :- server-client communication uses TCP/IP Three main things are needed to establish connection between server-client models.

1) Transport protocol (TCP or UDP)
2) Socket
3) IP address and port.

Scanned with CamScanner

* Transport protocol (TCP) :— TCP is a connection oriented protocol which stand for transmission control protocol. It is a reliable and secure protocol. In TCP the receiver can generate the anlcnowledgement of received packet so sender (client) does need to wait for the ankowledge and if the back response doesn't come or any packet error is generated. it will resend to the client.

* Sockets programming concepts :— There are diffront type of sokets. what make the differen between them is the topology of the communication and the type of address used There are three types of address the first one is represented by the internet address (the corresponding sockets are called internet socket)

Socket :— It represent an end of the communication defining a form of the inter process communication for processes found on the same station or on diffrent station of the same network. A pair of connected sockets represents a communication interface between two processes, an interface similar to the pipe interface in unix.

Association :— it defines in a unique way the connection established between two ends of communication as being a set of parameters. like (local-protocol, local-address, local-port, remote-address, remote-port).

Scanned with CamScanner

The source-port and destination-port notions allow a unique definition of the communication between pair of sockets and make a unique identification of a SAP (service access point) of the transport level.

* Socket descriptor :- it is a file descriptor the argument used in the library function.

* Binding :- it is the operation that associates a socket can be accessed.

* port :- It is an identifier used for making the difference between sockets found at the same address. There can be used address 1024 - 49151 (registered ports) and address between 49152 - 65535 named also dynamic or ephemeral ports the address between 1 - 1023 are reserved for the system.

* The family of address :- it represent the format of the address used in sockets. (usually the Internet and Unix address)

Conclusion :- Hence we studied and implement program to demonstrate UCP socket programming for wired network for peer to peer chat and multi user chat.

Scanned with CamScanner

```
/***** B9 Arithmatic client *****/


// Client
import java.io.*;

import java.net.*;

public class arithtcpclient

{

public static void main(String[] args) throws IOException

{

System.out.println();

System.out.println("ARITHMETIC CLIENT");

System.out.println("*****************");

try

{

Socket clientsoc = new Socket("localhost", 6666);

System.out.println("Enter the inputs");

PrintWriter out = new PrintWriter(clientsoc.getOutputStream(), true);

BufferedReader in = new BufferedReader(new
InputStreamReader(clientsoc.getInputStream()));

BufferedReader stdin = new BufferedReader(new
InputStreamReader(System.in));

String userinput;


while (true)

{

do
```

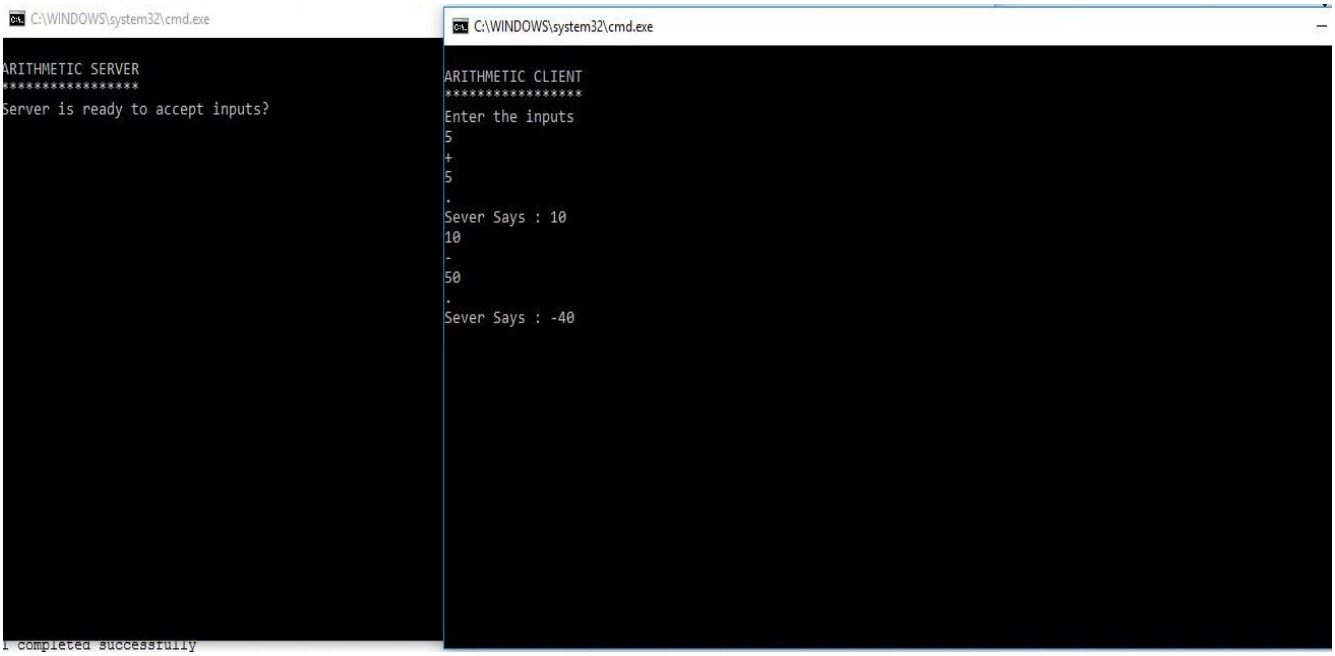
```
{
userinput = stdin.readLine();
out.println(userinput);
}while(!userinput.equals("."));
System.out.println("Sever Says : " + in.readLine());
}
}
catch(Exception e)
{
System.exit(0);
}
}
}
```

```
/**** B9 Arithmatic Server ****/

//arithtcpserver

import java.io.*;

import java.net.*;

public class arithtcpserver

{

public static void main(String arg[]) throws Exception

{

System.out.println();

System.out.println("ARITHMETIC SERVER");

System.out.println("*****************");

System.out.println("Server is ready to accept inputs…");

ServerSocket serversoc=new ServerSocket(6666);

Socket clientsoc = serversoc.accept();

PrintWriter out = new PrintWriter(clientsoc.getOutputStream(), true);

BufferedReader in = new BufferedReader(new
InputStreamReader(clientsoc.getInputStream()));

String inputline;

BufferedReader stdin = new BufferedReader(new
InputStreamReader(System.in));

try

{

while (true)

{

String s,op="",st;
```

```
int i=0,c=0;
int[] a=new int[2];
while(true)
{
s=in.readLine();
if(s.equals("+") || s.equals("-") || s.equals("*") || s.equals("/"))
op=s;
else if(s.equals("."))
break;
else
{
a[i]=Integer.parseInt(s);
i++;
}
}
if(op.equals("+"))
c=a[0]+a[1];
else if(op.equals("-"))
c=a[0]-a[1];
else if(op.equals("*"))
c=a[0]*a[1];
else if(op.equals("/"))
c=a[0]/a[1];
s=Integer.toString(c);
out.println(s);
}
```

```
}
catch(Exception e)
{
System.exit(0);
}
}
}
```

*****OUTPUT*****



/*** B9. File transfer client ***/

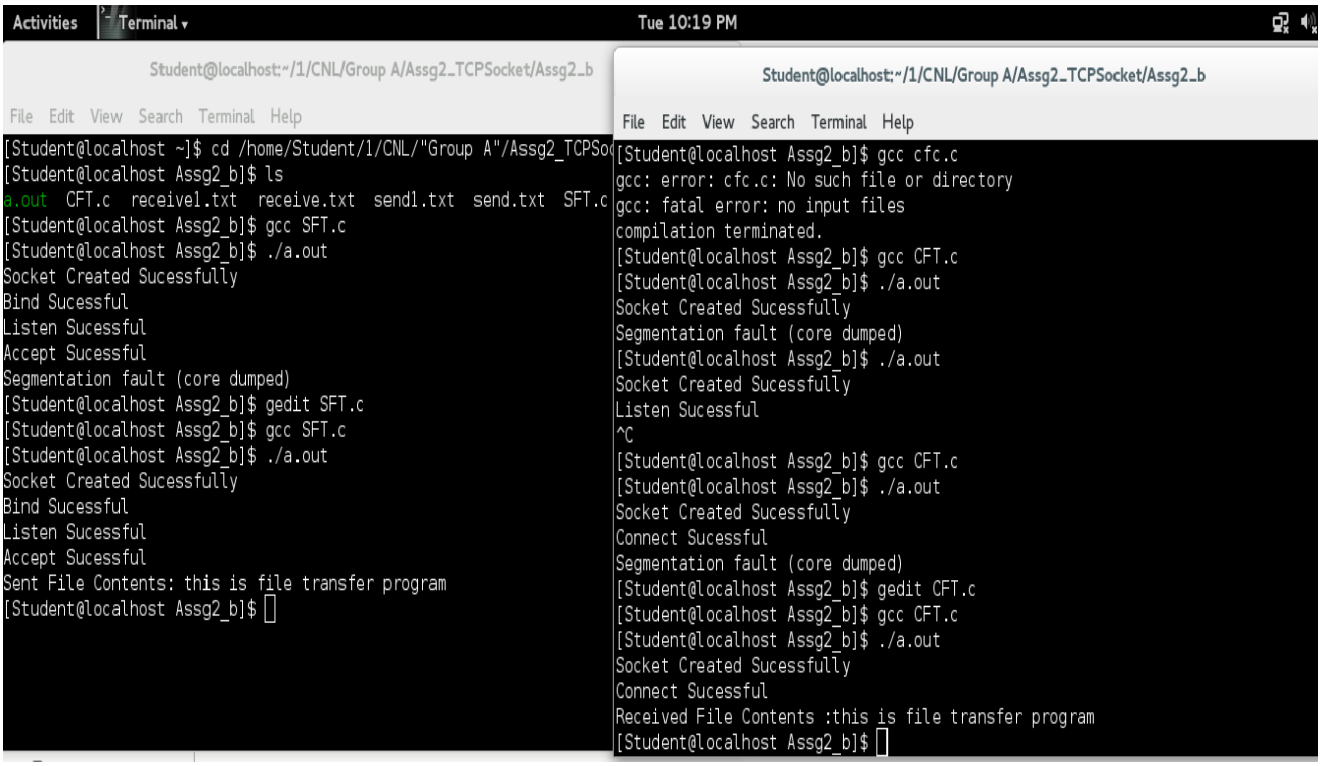
```
#include<sys/types.h>
#include<sys/socket.h>
#include<stdio.h>
#include<netinet/in.h>
#include <unistd.h>
#include<string.h>
#include<strings.h>
#include <arpa/inet.h>
#include <stdlib.h>
#define MAXBUFLEN 1000000
//#define buffsize  150
void main()
{
int b,sockfd,sin_size,con,n,len;
//char buff[256];

if((sockfd=socket(AF_INET,SOCK_STREAM,0))>=0)
printf("Socket Created Sucessfully \n");
struct sockaddr_in servaddr;

servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
servaddr.sin_port=6007;

sin_size = sizeof(struct sockaddr_in);
if((connect(sockfd,(struct sockaddr *) &servaddr, sin_size))==0)
//initiate a connection on a socket

printf("Connect Sucessful \n");

char buffer[10000];
char c[10000];
FILE *fp;
//bzero(buffer,10000);
//bzero(c,10000);
read(sockfd, buffer, 10000);

fp= fopen("/home/Student/1/CNL/Group
A/Assg2_TCPSocket/Assg2_b/receive.txt", "w+");


   /* Read and display data */
fwrite(buffer,  1,strlen(buffer) + 1, fp);
//fseek(fp, 0, SEEK_SET);
//fread(c, strlen(buffer)+1, 1, fp);
printf("Received File Contents :%s \n", buffer);
fclose(fp);
close(sockfd);
}
```

/*** B9. File transfer server ***/

```
#include<sys/types.h>
#include<sys/socket.h>
#include<stdio.h>
#include<netinet/in.h>
#include <unistd.h>
#include<string.h>
#include <arpa/inet.h>
#include <stdlib.h>
#define MAXBUFLEN 1000000
void main()
{
int b,sockfd,connfd,sin_size,l,n,len;

if((sockfd=socket(AF_INET,SOCK_STREAM,0))>=0)  //socket creation
printf("Socket Created Sucessfully \n");                 //on success
0 otherwise -1

struct sockaddr_in servaddr;
struct sockaddr_in clientaddr;

servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=inet_addr("127.0.0.1");
servaddr.sin_port=6007;

if((bind(sockfd, (struct sockaddr *)&servaddr,sizeof(servaddr)))==0)
//bind() assigns the
     //  address  specified  by  addr  to  the  socket  referred  to
by the file
      // descriptor sockfd.  addrlen  specifies  the  size,  in
bytes,  of  the
     //  address structure pointed to by addr.  Traditionally, this
operation is
      // called â€œassigning a name to a socketâ€□.

printf("Bind Sucessful \n");

if((listen(sockfd,5))==0)  //listen for connections on a socket
```

```
printf("Listen Sucessful \n");

sin_size = sizeof(struct sockaddr_in);
if((connfd=accept(sockfd,(struct sockaddr *)&clientaddr,&sin_size))>0)
printf("Accept Sucessful \n");
char buffer[100];
char c[10000] = "this is file transfer program";
//char source[MAXBUFLEN + 1];
//bzero(buffer,10000);

FILE *fp;
fp= fopen("/home/Student/1/CNL/Group
A/Assg2_TCPSocket/Assg2_b/send.txt", "w+");
//fp = fopen("file.txt", "w+");

   /* Write data to the file */
fwrite(c, 1, strlen(c) + 1, fp);

   /* Seek to the beginning of the file */
fseek(fp, 0, SEEK_SET);

   /* Read and display data */
fread(buffer, 1,strlen(c)+1, fp);

  // fclose(fp);

write(connfd, buffer, strlen(buffer));
printf("Sent File Contents: %s\n", buffer);
fclose(fp);
close(sockfd);
}
```

OUTPUT


Activities
Terminal
Tue 10:19 PM
Student@localhost:~/1/CNL/Group A/Assg2_TCPSocket/Assg2_b
File Edit View Search Terminal Help
[Student@localhost ~]$ cd /home/Student/1/CNL/"Group A"/Assg2_TCPSo
[Student@localhost Assg2_b]$ ls
a.out CFT.c receive1.txt receive.txt send1.txt send.txt SFT.c
[Student@localhost Assg2_b]$ gcc SFT.c
[Student@localhost Assg2_b]$ ./a.out
Socket Created Sucessfully
Bind Sucessful
Listen Sucessful
Accept Sucessful
Segmentation fault (core dumped)
[Student@localhost Assg2_b]$ gedit SFT.c
[Student@localhost Assg2_b]$ gcc SFT.c
[Student@localhost Assg2_b]$ ./a.out
Socket Created Sucessfully
Bind Sucessful
Listen Sucessful
Accept Sucessful
Sent File Contents: this is file transfer program
[Student@localhost Assg2_b]$
Student@localhost:~/1/CNL/Group A/Assg2_TCPSocket/Assg2_b
File Edit View Search Terminal Help
[Student@localhost Assg2_b]$ gcc cfc.c
gcc: error: cfc.c: No such file or directory
gcc: fatal error: no input files
compilation terminated.
[Student@localhost Assg2_b]$ gcc CFT.c
[Student@localhost Assg2_b]$ ./a.out
Socket Created Sucessfully
Segmentation fault (core dumped)
[Student@localhost Assg2_b]$ ./a.out
Socket Created Sucessfully
Listen Sucessful
^C
[Student@localhost Assg2_b]$ gcc CFT.c
[Student@localhost Assg2_b]$ ./a.out
Socket Created Sucessfully
Connect Sucessful
Segmentation fault (core dumped)
[Student@localhost Assg2_b]$ gedit CFT.c
[Student@localhost Assg2_b]$ gcc CFT.c
[Student@localhost Assg2_b]$ ./a.out
Socket Created Sucessfully
Connect Sucessful
Received File Contents :this is file transfer program
[Student@localhost Assg2_b]$