1.NETWORKING COMMANDS

AIM:

To study the basic networking commands.

NETWORKING COMMANDS:

C:\>tcpdump: Tcpdump is a command line utility that allows you to capture and analyze network traffic going through your system. It is often used to help troubleshoot network issues, as well as a security tool.

C:\>netstat: Netstat displays a variety of statistics about a computers active TCP/IP connections. This tool is most useful when you're having trouble with TCP/IP applications such as HTTP, andFTP.

C:\>nbtstat –a: This command helps solve problems with NetBIOS name resolution. (Nbtstands for NetBIOS over TCP/IP)

C:\>ifconfig: The command ifconfig stands for interface configurator. This command enables us to initialize an interface, assign IP address, enable or disable an interface. It display route and network interface.

C:\>ipconfig: The ipconfig command displays information about the host (the computer yoursitting at)computer TCP/IP configuration.

C:\>nslookup: Nslookup is used for diagnosing DNS problems. If you can access aresource by specifying an IP address but not it's DNS you have a DNS problem.

` The route command displays the computers routing table. A typical computer, with a single network interface, connected to a LAN, with a router is fairly simple and generally doesn'tpose any network problems. But if you're having trouble accessing other computers on your network, you can use the route command to make sure the entries in the routing table are correct.

C:\>tracert: The tracert command displays a list of all the routers that a packet has to go through toget from the computer where tracert is run to any other computer on the internet.

C:\>pathping: Pathping is unique to Window's, and is basically a combination of the Ping and Tracert commands. Pathping traces the route to the destination address then launches a 25 second test of each router along the way, gathering statistics on the rate of data loss along each hop.

C:\>ping: Ping is the most basic TCP/IP command, and it's the same as placing a phone call to your best friend. You pick up your telephone and dial a number, expecting your best friend to replywith "Hello" on the other end. Computers make phone calls to each other over a network by usinga Ping command. The Ping commands main purpose is to place a phone call to another computer on the network, and request an answer. Ping has 2 options it can use to place a phone call to another computer on the network. It can use the computers name or IP address

RESULT:

Thus the above list of primitive has been studied.

2.WRITE A HTTP WEB CLIENT PROGRAM TO DOWNLOAD A WEB PAGE USING TCP SOCKETS

PROGRAM

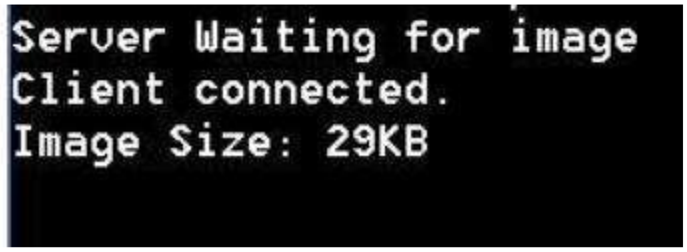
CLIENT

```
import javax.swing.*;
import java.net.*;
import java.awt.image.*;
import javax.imageio.*;
import java.io.*;
import java.awt.image.BufferedImage;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;
public class Client{
public static void main(String args[]) throws Exception{
Socket soc;
BufferedImage img = null;
soc=new Socket("localhost",4000);
System.out.println("Client is running. ");
try {
System.out.println("Reading image from disk. ");
img = ImageIO.read(new File("digital_image_processing.jpg"));
ByteArrayOutputStream baos = new ByteArrayOutputStream();
ImageIO.write(img, "jpg", baos);
baos.flush();
byte[] bytes = baos.toByteArray();
baos.close(); System.out.println("Sending image to server. ");
OutputStream out = soc.getOutputStream();
DataOutputStream dos = new DataOutputStream(out);
dos.writeInt(bytes.length);
dos.write(bytes, 0, bytes.length);
System.out.println("Image sent to server. ");
dos.close();
out.close();
}catch (Exception e) {
System.out.println("Exception: " + e.getMessage());
soc.close();
}
soc.close();
}
}
```

SERVER

```
import java.net.*;
import java.io.*;
import java.awt.image.*;
import javax.imageio.*;
import javax.swing.*;
class Server {
public static void main(String args[]) throws Exception{
ServerSocket server=null;
Socket socket;
server=new ServerSocket(4000);
System.out.println("Server Waiting for image");
socket=server.accept();
System.out.println("Client connected.");
InputStream in = socket.getInputStream();
DataInputStream dis = new DataInputStream(in);
int len = dis.readInt();
System.out.println("Image Size: " + len/1024 + "KB");
byte[] data = new byte[len];
dis.readFully(data);
dis.close();
in.close();
InputStream ian = new ByteArrayInputStream(data);
BufferedImage bImage = ImageIO.read(ian); JFrame f =
new JFrame("Server"); ImageIcon icon = new
ImageIcon(bImage);
JLabel l = new JLabel();
l.setIcon(icon);
f.add(l);
f.pack();
f.setVisible(true);

}

}
```

OUTPUT:



```
Server Waiting for image
Client connected.
Image Size: 29KB
```

3. A. SOCKET PROGRAM FOR ECHO

PROGRAM:

ECHO CLIENT

```
import java.io.*;
import java.net.*;
public class eclient
{
public static void main(String args[])
{
Socket c=null;
String line;
DataInputStream is,is1;
PrintStream os;
try
{
c=new Socket("localhost",8080);
}
catch(IOException e)
{
System.out.println(e);
}
try
{
os=new PrintStream(c.getOutputStream());
is=new DataInputStream(System.in);
is1=new DataInputStream(c.getInputStream());
do
{
System.out.println("client");
line=is.readLine();
os.println(line);
if(!line.equals("exit"))
System.out.println("server:"+is1.readLine());
}while(!line.equals("exit"));
}
catch(IOException e)
{
System.out.println("socket closed");
}
}
}
```

ECHO SERVER:

```
import java.io.*;
import java.net.*;
import java.lang.*;
public class eserver
{
public static void main(String args[])throws IOException
{
ServerSocket s=null;
String line;
DataInputStream is;
PrintStream ps;
Socket c=null;
try
{
s=new ServerSocket(8080);
}
catch(IOException e)
{
System.out.println(e);
}
try
{
c=s.accept();
is=new DataInputStream(c.getInputStream());
ps=new PrintStream(c.getOutputStream());
while(true)
{
line=is.readLine();
System.out.println("msg received and sent back to client");
ps.println(line);
}
}
catch(IOException e)
{
System.out.println(e);
}
}
}
```

OUTPUT:

CLIENT

```
Enter the IP address 127.0.0.1
CONNECTION ESTABLISHED
Enter the data CSE
Client received CSE
```

SERVER

```
CONNECTION ACCEPTED
Server received CSE
```

3.B. CLIENT- SERVER APPLICATION FOR CHAT

PROGRAM:

TCPSERVER1

```
import java.net.*;
import java.io.*;
public class TCPserver1
{
public static void main(String arg[])
{
ServerSocket s=null;
String line;
DataInputStream is=null,is1=null;
PrintStream os=null;
Socket c=null;
try
{
s=new ServerSocket(9999);
}
catch(IOException e)
{
System.out.println(e);
}
try
{
c=s.accept();
is=new DataInputStream(c.getInputStream());
is1=new DataInputStream(System.in);
os=new PrintStream(c.getOutputStream());
do
{
line=is.readLine();
System.out.println("Client:"+line);
System.out.println("Server:");
line=is1.readLine();
os.println(line);
}
while(line.equalsIgnoreCase("quit")==false);
is.close();
os.close();
}
catch(IOException e)
{
System.out.println(e);
}
}
}
```

TCPCLIENT1.JAVA

```
import java.net.*;
import java.io.*;
public class TCPclient1
{
public static void main(String arg[])
{
Socket c=null;
String line;
DataInputStream is,is1;
PrintStream os;
try
{
c=new Socket("10.0.200.36",9999);
}
catch(IOException e)
{
System.out.println(e);
}
try
{
os=new PrintStream(c.getOutputStream());
is=new DataInputStream(System.in);
is1=new DataInputStream(c.getInputStream());
do
{
System.out.println("Client:");
line=is.readLine();
os.println(line);
System.out.println("Server:" + is1.readLine());
}
while(line.equalsIgnoreCase("quit")==false);
is1.close();
os.close();
}
catch(IOException e)
{
System.out.println("Socket Closed!Message Passing is over");
}
}
```

OUTPUT:

SERVER

```
C:\Program Files\Java\jdk1.5.0\bin>javac TCPserver1.java Note:
TCPserver1.java uses or overrides a deprecated API. Note:
Recompile with -deprecation for details. C:\Program
Files\Java\jdk1.5.0\bin>java TCPserver1
Client: Hai Server
Server:Hai Client
Client: How are you
Server:Fine
Client: quit
Server:quit
```

CLIENT

```
C:\Program Files\Java\jdk1.5.0\bin>javac TCPclient1.java
Note: TCPclient1.java uses or overrides a deprecated API.
Note: Recompile with -deprecation for details. C:\Program
Files\Java\jdk1.5.0\bin>java TCPclient1
Client: Hai Server
Server: Hai Client
Client: How are you
Server: Fine
Client: quit
Server: quit
```

3.C. FILE TRANSFER IN CLIENT & SERVER

PROGRAM:

CLIENT SIDE

```
import java.net.*;
import java.io.*;
public class FileClient{
public static void main (String [] args ) throws IOException
{
Int filesize=6022386;
System.currentTimeMillis();
int bytesRead;
int current = 0;
Socket sock = new Socket("127.0.0.1",13267);
System.out.println("Connecting...");
byte [] mybytearray = new byte [filesize];
InputStream is = sock.getInputStream();
FileOutputStream fos = new FileOutputStream("source-copy.pdf");
BufferedOutputStream bos = new BufferedOutputStream(fos);
bytesRead = is.read(mybytearray,0,mybytearray.length);
current = bytesRead;
do {
bytesRead =is.read(mybytearray, current, (mybytearray.length-current));
if(bytesRead >= 0) current += bytesRead;
} while(bytesRead > -1);
bos.write(mybytearray, 0 , current);
bos.flush();
long end = System.currentTimeMillis();
System.out.println(end-start);
bos.close();
sock.close();
}
}
```

SERVER SIDE

```
import java.net.*;
import java.io.*;
public class FileServer
{
public static void main (String [] args ) throws IOException {
ServerSocket servsock = new ServerSocket(13267);
while (true)
{
System.out.println("Waiting...");
Socket sock =servsock.accept();
System.out.println("Accepted connection : " + sock);
File myFile = new File("source.pdf");
byte [] mybytearray = new byte [(int)myFile.length()];
FileInputStream fis = new FileInputStream(myFile);
BufferedInputStream bis = new BufferedInputStream(fis);
bis.read(mybytearray,0,mybytearray.length);
OutputStream os = sock.getOutputStream();
System.out.println("Sending...");
os.write(mybytearray,0,mybytearray.length);
os.flush();
sock.close();
}
}
}
```

OUTPUT:

SERVER OUTPUT

```
C:\Program Files\Java\jdk1.6.0\bin>javac FServer.java
C:\Program Files\Java\jdk1.6.0\bin>java FServer
Waiting for clients...
Connection Established
Client wants file:network.txt
```

CLIENT OUTPUT

```
C:\Program Files\Java\jdk1.6.0\bin>javac FClient.java
C:\Program Files\Java\jdk1.6.0\bin>java FClient
Connection request.....Connected
Enter the filename: network.txt
Computer networks: A computer network, often simply referred to as a network, is acollection
of computers and devices connected by communications channels thatfacilitates
communications among users and allows users to shareresources with other user
```

4. SIMULATION OF DNS USING UDP SOCKETS.

PROGRAM:

UDP DNS SERVER/UDPDNSSERVER

```
import java.io.*;
import java.net.*;
public class udpdnsserver
{
private static int indexOf(String[] array, String str)
{
str = str.trim();
for (int i=0; i < array.length; i++)
{
if (array[i].equals(str))
return i;
}
return -1;
}
public static void main(String arg[])throws IOException
{
String[] hosts = {"yahoo.com", "gmail.com","cricinfo.com", "facebook.com"};
String[] ip = {"68.180.206.184", "209.85.148.19","80.168.92.140", "69.63.189.16"};
System.out.println("Press Ctrl + C to Quit");
while (true){
DatagramSocket serversocket=new DatagramSocket(1362);
byte[] senddata = new byte[1021];
byte[] receivedata = new byte[1021];
DatagramPacket recvpack = new DatagramPacket(receivedata, receivedata.length);
serversocket.receive(recvpack);
String sen = new String(recvpack.getData());
InetAddress ipaddress =recvpack.getAddress();
int port = recvpack.getPort();
String capsent;
System.out.println("Request for host " + sen);
if(indexOf (hosts, sen) != -1)
capsent = ip[indexOf(hosts, sen)];
else
capsent = "Host Not Found";
senddata = capsent.getBytes();
DatagramPacket pack = new DatagramPacket (senddata, senddata.length,ipaddress,port);
serversocket.send(pack);
serversocket.close();
}
}
}
```

UDP DNS CLIENT –UDPDNSCLIENT

```
import java.io.*;
import java.net.*;
public class udpdnsclient
{
public static void main(String args[])throws IOException
{
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
DatagramSocket clientsocket = new DatagramSocket();
InetAddress ipaddress;
if (args.length == 0)
ipaddress = InetAddress.getLocalHost();
else
ipaddress = InetAddress.getByName(args[0]);
byte[] senddata =new byte[1024];
byte[] receivedata = new byte[1024];
int portaddr = 1362;
System.out.print("Enter the hostname : ");
String sentence = br.readLine();
Senddata = sentence.getBytes();
DatagramPacket pack = new DatagramPacket(senddata,senddata.length, ipaddress,portaddr);
clientsocket.send(pack);
DatagramPacket recvpack =new DatagramPacket(receivedata,receivedata.length);
clientsocket.receive(recvpack);
String modified = new String(recvpack.getData());
System.out.println("IP Address: " + modified);
clientsocket.close();
}
}
```

OUTPUT:

SERVER

```
javac udpdnsserver.java
java udpdnsserver
Press Ctrl + C to Quit Request for host yahoo.com
Request for host cricinfo.com
Request for host youtube.com
```

CLIENT

```
javac udpdnsclient.java
java udpdnsclient
Enter the hostname : yahoo.com
IP Address: 68.180.206.184
java udpdnsclient
Enter the hostname : cricinfo.com
IP Address: 80.168.92.140
java udpdnsclient
Enter the hostname : youtube.com
IP Address: Host Not Found
```

5.A. WRITE A CODE SIMULATING ARP PROTOCOLS.

PROGRAM

ARP CLIENT

```
import java.io.*;
import java.net.*;
class ArpClient
{
public static void main(String args[])throws IOException
{
try
{
Socket ss=new Socket(InetAddress.getLocalHost(),1100);
PrintStream ps=new PrintStream(ss.getOutputStream());
BufferedReader br=new BufferedReader(newInputStreamReader(System.in));
String ip;
System.out.println("Enter the IPADDRESS:");
ip=br.readLine();
ps.println(ip);
String str,data;
BufferedReader br2=new BufferedReader(newInputStreamReader(ss.getInputStream()));
System.out.println("ARP From Server::");
do
{
str=br2.readLine();
System.out.println(str);
}
while(!(str.equalsIgnoreCase("end")));
}
catch(IOException e)
{
System.out.println("Error"+e);
}
}
}
```

ARP SERVER

```
import java.io.*;
import java.net.*;
class ArpServer
{
public static void main(String args[])throws IOException
{
try
{
ServerSocket ss=new ServerSocket(1100);
Socket s=ss.accept();
PrintStream ps=new PrintStream(s.getOutputStream());
BufferedReader br1=new BufferedReader(newInputStreamReader(s.getInputStream()));
String ip;
ip=br1.readLine();
Runtime r=Runtime.getRuntime();
Process p=r.exec("arp -a "+ip);
BufferedReader br2=new BufferedReader(newInputStreamReader(p.getInputStream()));
String str;
while((str=br2.readLine())!=null)
{
ps.println(str);
}
}
catch(IOException e)
{
System.out.println("Error"+e);
}
}
}
```

OUTPUT

```
C:\Networking Programs>java ArpServer
C:\Networking Programs>java ArpClient
Enter the IPADDRESS:
192.168.11.58
ARP From Server::
Interface: 192.168.11.57 on Interface 0x1000003
Internet Address      Physical Address     Type
192.168.11.58         00-14-85-67-11-84    dynamic
```

5.B. WRITE A CODE SIMULATING RARP PROTOCOLS.

PROGRAM:

CLIENT:

```
import java.io.*;
import java.net.*;
import java.util.*;
class Clientrarp12
{
public static void main(String args[])
{
try
{
DatagramSocket client=new DatagramSocket();
InetAddress addr=InetAddress.getByName("127.0.0.1");
byte[] sendbyte=new byte[1024];
byte[] receivebyte=new byte[1024];
BufferedReader in=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Enter the Physical address (MAC):");
String str=in.readLine();
sendbyte=str.getBytes();
DatagramPacket sender=newDatagramPacket(sendbyte,sendbyte.length,addr,1309);
client.send(sender);
DatagramPacket receiver=new DatagramPacket(receivebyte,receivebyte.length);
client.receive(receiver);
String s=new String(receiver.getData()); System.out.println("The Logical Address
is(IP):"+s.trim()); client.close();
}
catch(Exception e)
{
System.out.println(e);
}
}
}
```

SERVER:

```
import java.io.*;
import java.net.*;
import java.util.*;
class Serverrarp12
{
public static void main(String args[])
{
try
{
DatagramSocket server=new DatagramSocket(1309);
while(true)
{
byte[] sendbyte=new byte[1024];
byte[] receivebyte=new byte[1024];
DatagramPacket receiver=new DatagramPacket(receivebyte,receivebyte.length);
server.receive(receiver);
String str=new String(receiver.getData()); String
s=str.trim();
InetAddress addr=receiver.getAddress();
int port=receiver.getPort();
String ip[]={"165.165.80.80","165.165.79.1"};
String mac[]={"6A:08:AA:C2","8A:BC:E3:FA"};
for(int i=0;i<ip.length;i++)
{
if(s.equals(mac[i]))
{
sendbyte=ip[i].getBytes();
DatagramPacket sender=newDatagramPacket(sendbyte,sendbyte.length,addr,port);
server.send(sender);
break;
}
}
break;
}
}
catch(Exception e)
{
System.out.println(e);
}
}
}
```

OUTPUT:

```
I:\ex>java Serverrarp12
I:\ex>java Clientrarp12
Enter the Physical address (MAC):
6A:08:AA:C2
The Logical Address is(IP): 165.165.80.80
```

6. SIMULATION OF ERROR CORRECTION CODE (LIKE CRC)

PROGRAM:

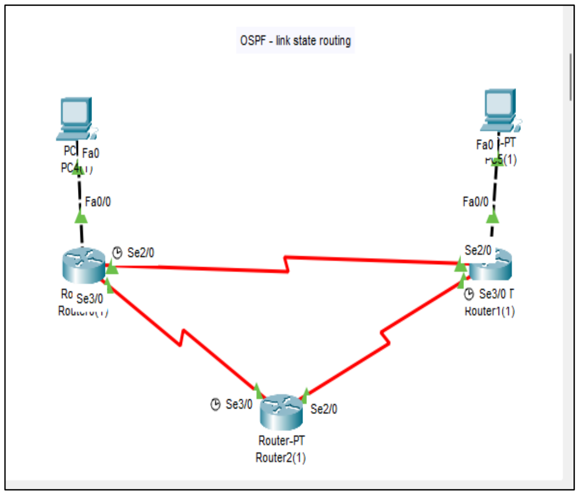
```
import java.io.*;
class CRC
{
public static void main(String args[]) throws IOException
{
 BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
 System.out.println("Enter Generator:");
 String gen = br.readLine();
 System.out.println("Enter Data:");
 String data = br.readLine();
 String code = data;
 while(code.length() < (data.length() + gen.length() - 1))
 code = code + "0";
 code = data + div(code,gen);
 System.out.println("The transmitted Code Word is: " + code);
 System.out.println("Please enter the received Code Word: ");
 String rec = br.readLine();
 if(Integer.parseInt(div(rec,gen)) == 0)
 System.out.println("The received code word contains no errors.");
 else
 System.out.println("The received code word contains errors.");
}
static String div(String num1,String num2)
{
 int pointer = num2.length();
 String result = num1.substring(0, pointer);
 String remainder = "";
 for(int i = 0; i < num2.length(); i++)
 {
 if(result.charAt(i) == num2.charAt(i))
 remainder += "0";
 else
 remainder += "1";
 }
 while(pointer < num1.length())
 {
 if(remainder.charAt(0) == '0')
 {
 remainder = remainder.substring(1, remainder.length());
 remainder = remainder + String.valueOf(num1.charAt(pointer));
 pointer++;
 }
 result = remainder;
 remainder = "";
 for(int i = 0; i < num2.length(); i++)
 {
 if(result.charAt(i) == num2.charAt(i))
 remainder += "0";
 else
 remainder += "1";
 }
 }
 return remainder.substring(1,remainder.length());
}
}
```

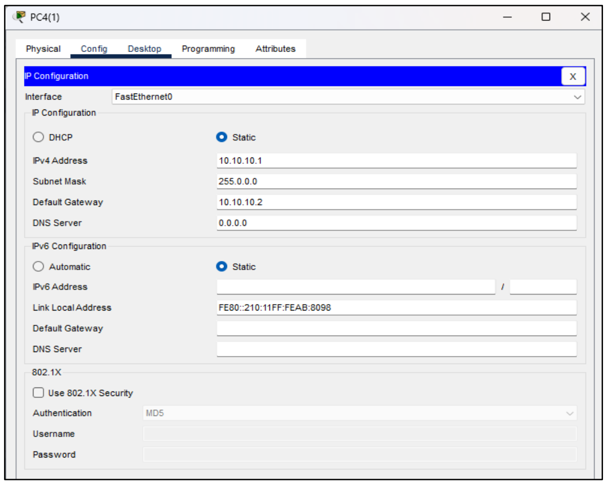
OUTPUT:

```
Enter data as binary bit stream (7 bits):

1110110

Code word is 11101100110

Enter the received hamming code 10101100110

There is an error in bit position 2 of received code word corrected code word is 11101100110

Enter data as binary bit stream(7 bits) 11101110

Code word is 11101100110

Enter the received hamming code 00101100110

There are 2 or more error in received code… Sorry…!
```

7. SIMULATION OF LINK STATE ROUTING ALGORITHM

OUTPUT:





PC4(1)

Physical | Config | Desktop | Programming | Attributes

IP Configuration

Interface: FastEthernet0

IP Configuration

DHCP [✓ Static]

IPv4 Address: 10.10.10.1

Subnet Mask: 255.0.0.0

Default Gateway: 10.10.10.2

DNS Server: 0.0.0.0

IPv6 Configuration

Automatic [✓ Static]

IPv6 Address: /

Link Local Address: FE80::210:11FF:FEAB:8098

Default Gateway:

DNS Server:

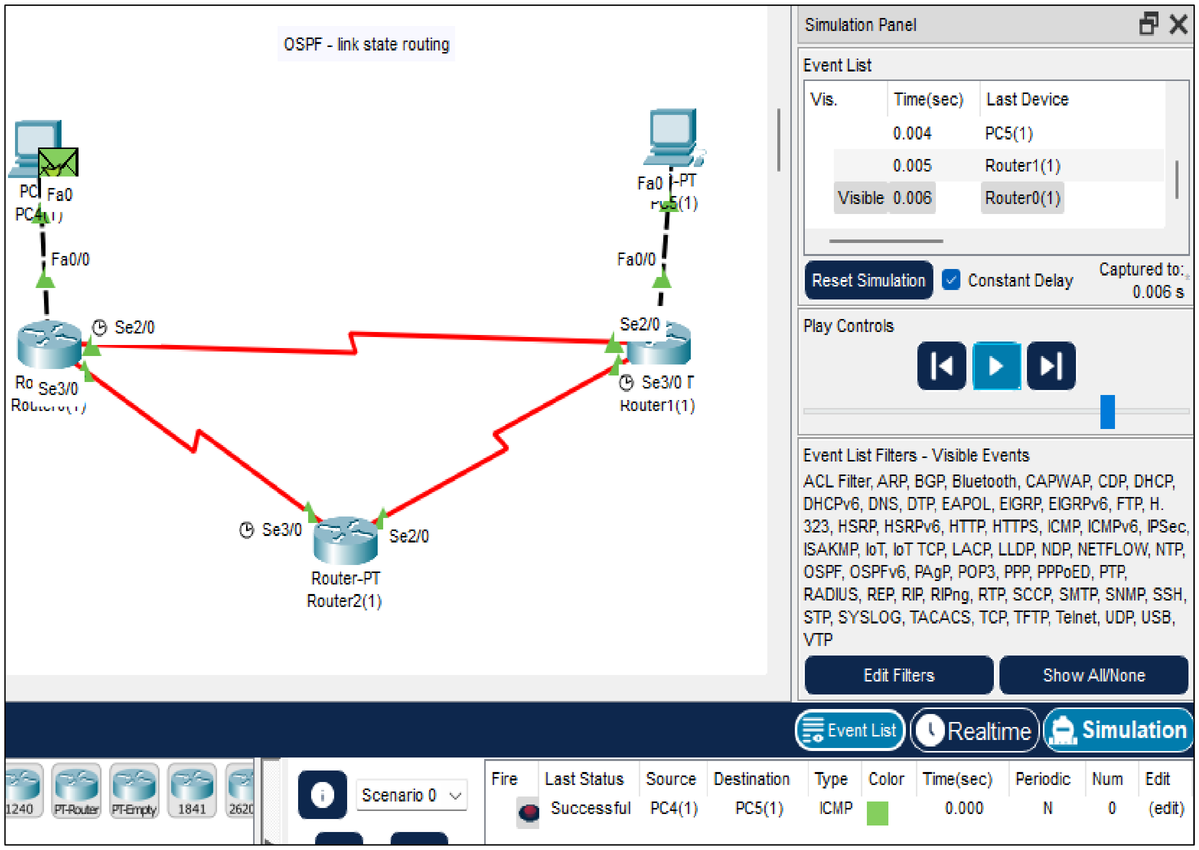
802.1X

☐ Use 802.1X Security

Authentication: MD5

Username:

Password:



Event List

Vis.	Time(sec)	Last Device
	0.004	PC5(1)
	0.005	Router1(1)
Visible	0.006	Router0(1)

Reset Simulation ✓ Constant Delay Captured to: 0.006 s

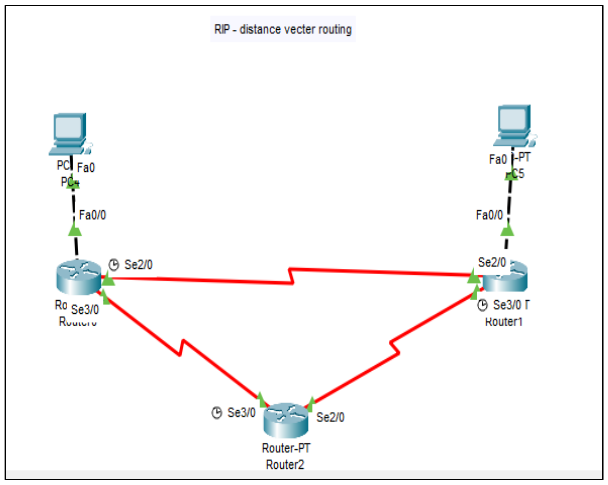
Event List Filters - Visible Events

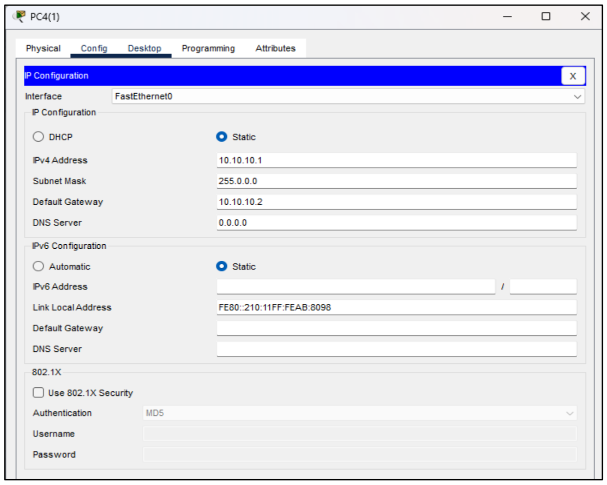
ACL Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, IoT, IoT TCP, LACP, LLDP, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoED, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, USB, VTP

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit
●	Successful	PC4(1)	PC5(1)	ICMP	🟩	0.000	N	0	(edit)

8. SIMULATION OF DISTANCE VECTOR ROUTING ALGORITHM

OUTPUT:





PC4(1)

Physical | Config | Desktop | Programming | Attributes

IP Configuration

Interface	FastEthernet0

IP Configuration

DHCP [✓ Static]

IPv4 Address	10.10.10.1
Subnet Mask	255.0.0.0
Default Gateway	10.10.10.2
DNS Server	0.0.0.0

IPv6 Configuration

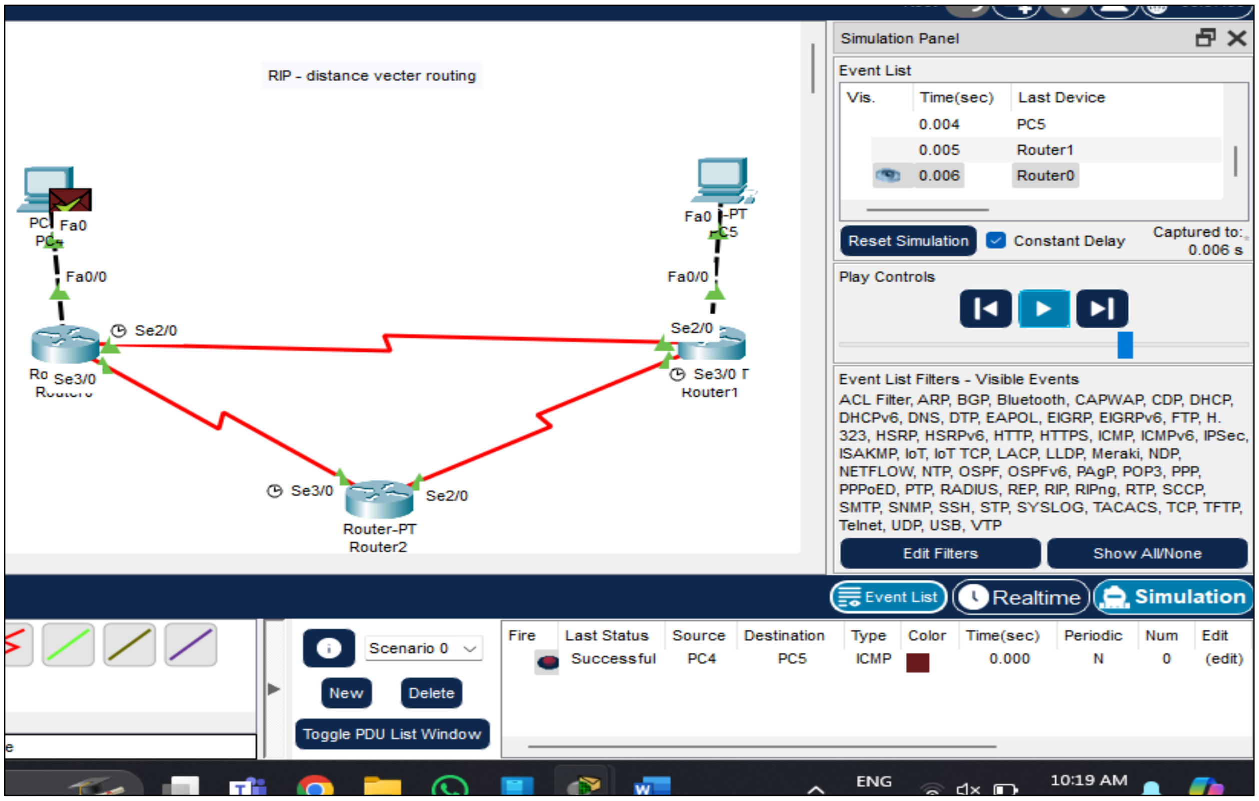
Automatic [✓ Static]

IPv6 Address	/
Link Local Address	FE80::210:11FF:FEAB:8098
Default Gateway	
DNS Server	

802.1X

☐ Use 802.1X Security

Authentication	MD5
Username	
Password	



Simulation Panel

Event List

Vis.	Time(sec)	Last Device
	0.004	PC5
	0.005	Router1
	0.006	Router0

Constant Delay ✓

Captured to: 0.006 s

Event List Filters - Visible Events

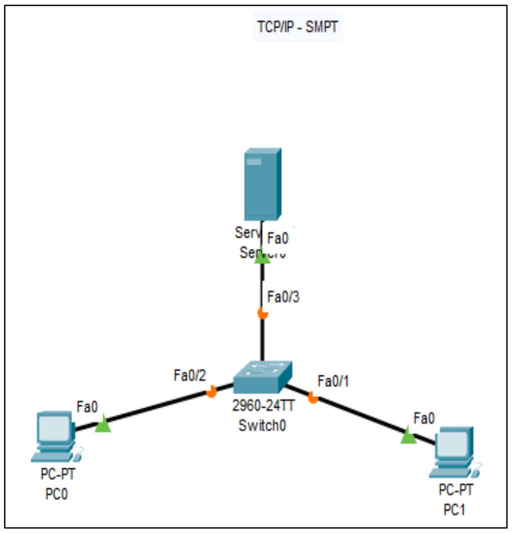
ACL Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, IoT, IoT TCP, LACP, LLDP, Meraki, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoED, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, USB, VTP

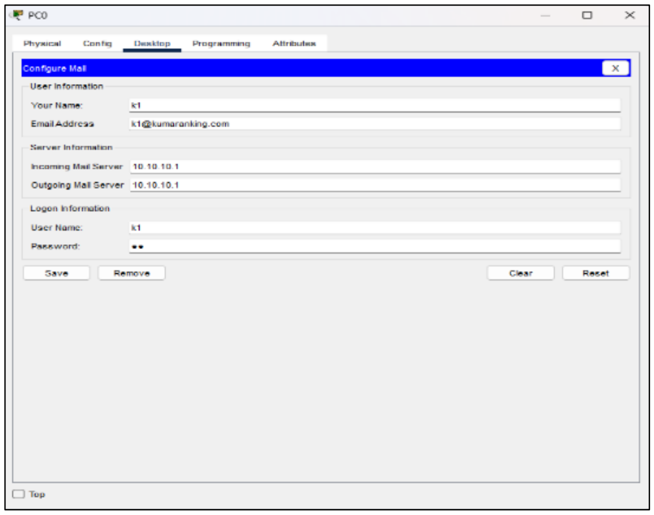
Scenario 0

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit
●	Successful	PC4	PC5	ICMP	🟥	0.000	N	0	(edit)

9. CONSTRUCT AND SIMULATE SMTP USING NETWORK SIMULATOR

OUTPUT:





Configure Mail

User Information

Your Name: k1

Email Address: k1@kumaranking.com

Server Information

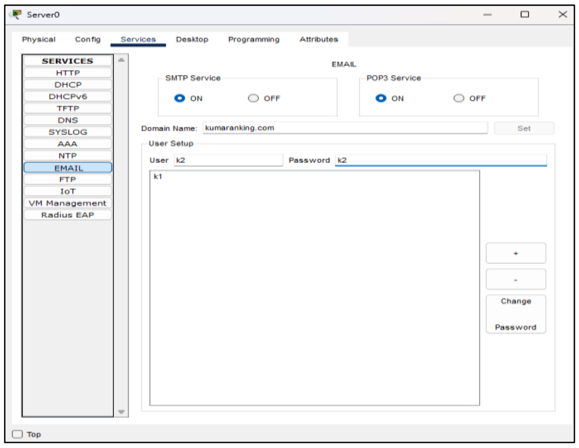
Incoming Mail Server: 10.10.10.1

Outgoing Mail Server: 10.10.10.1

Logon Information

User Name: k1

Password: ••



EMAIL

SMTP Service: [✓ ON] OFF

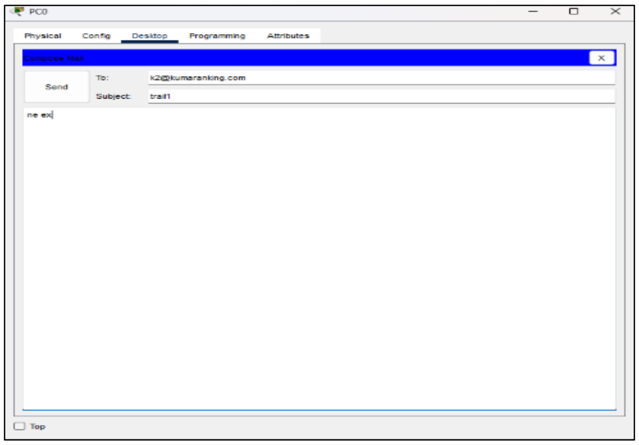
POP3 Service: [✓ ON] OFF

Domain Name: kumaranking.com

User Setup

User k2 Password k2

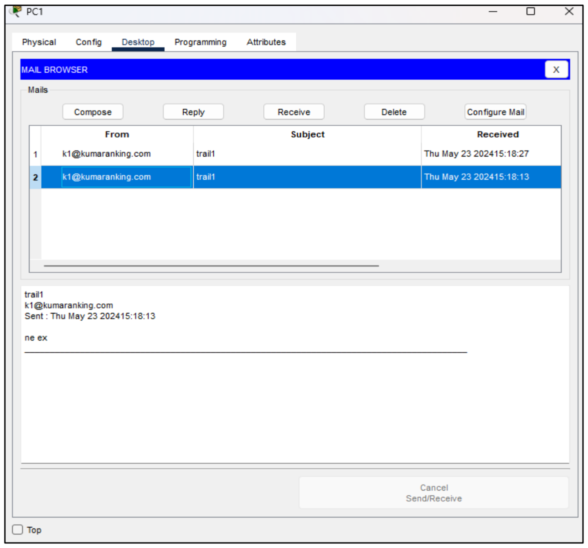
k1



To: k2@kumaranking.com

Subject: trail1

ne ex



PC1

Physical | Config | Desktop | Programming | Attributes

MAIL BROWSER

Mails

Compose | Reply | Receive | Delete | Configure Mail

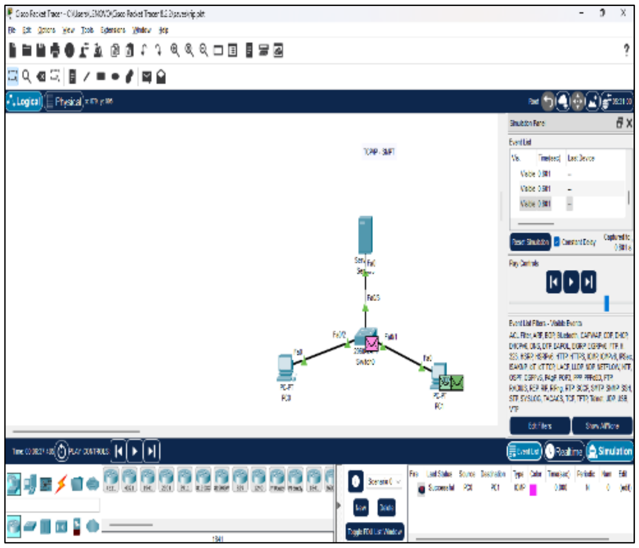
	From	Subject	Received
1	k1@kumaranking.com	trail1	Thu May 23 202415:18:27
2	k1@kumaranking.com	trail1	Thu May 23 202415:18:13

trail1
k1@kumaranking.com
Sent : Thu May 23 202415:18:13

ne ex

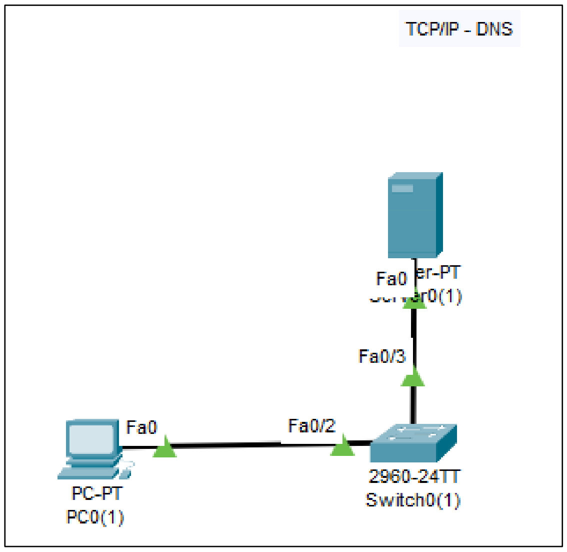
Cancel
Send/Receive

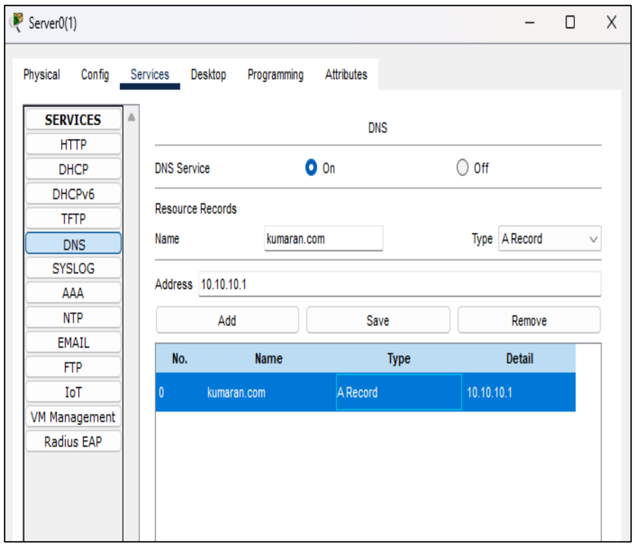
Top



10. CONSTRUCT AND SIMULATE DNS USING NETWORK SIMULATOR

OUTPUT:





DNS

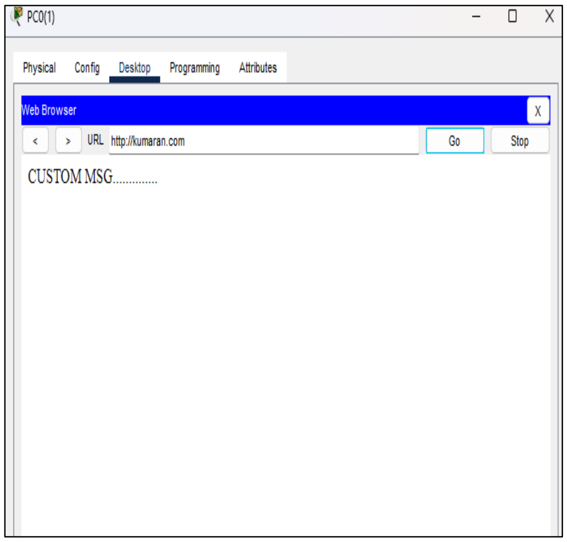
DNS Service [✓ On] Off

Resource Records

Name kumaran.com Type A Record

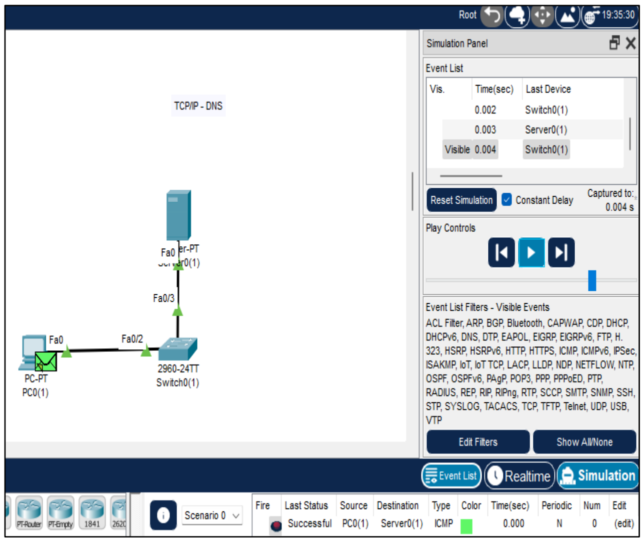
Address 10.10.10.1

No.	Name	Type	Detail
0	kumaran.com	A Record	10.10.10.1



URL http://kumaran.com

CUSTOM MSG.............



Event List

Vis.	Time(sec)	Last Device
	0.002	Switch0(1)
	0.003	Server0(1)
Visible	0.004	Switch0(1)

✓ Constant Delay

Captured to: 0.004 s

Event List Filters - Visible Events

ACL Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H. 323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, IoT, IoT TCP, LACP, LLDP, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoED, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, USB, VTP

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit
●	Successful	PC0(1)	Server0(1)	ICMP	🟩	0.000	N	0	(edit)