INDERPRASTHA ENGINEERING COLLEGE GHAZIABAD

Department of Information Technology

Computer Networks Lab (RCS-651)

(2019-20)

Name	**:**	**Natasha Sharma**
Roll Number	**:**	**1703013043**
Course	**:**	**B.Tech. (I.T.)**
Year	**:**	**3**
Semester	**:**	**6**
Section	**:**	**A**

INDEX

PROGRAM:1

OBJECTIVE - Study of different types of Networks cables and practically implement the cross -wired cable and straight through cable using clamping tool.

THEORY –

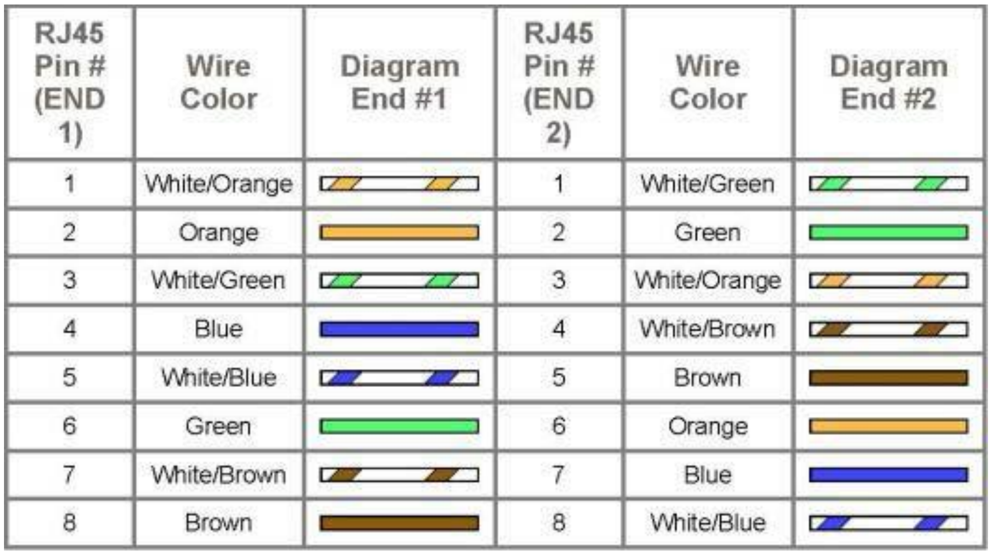
PROCEDURE: To do these practical following steps should be done:

1. Start by stripping off about 2 inches of the plastic jacket off the end of the cable. Be very careful at this point, as to not nick or cut into the wires, which are inside. Doing so could alter the characteristics of your cable, or even worse render is useless. Check the wires, one more time for nicks or cuts. If there are any, just whack the whole end off, and start over.

2. Spread the wires apart, but be sure to hold onto the base of the jacket with your other hand. You do not want the wires to become untwisted down inside the jet. Category 5 cable must only have 1/2 of an inch of 'untwisted' wire at the end; otherwise it will be 'out of spec'. At this point, you obviously have ALOT more than 1/2 of an inch of un-twisted wire.

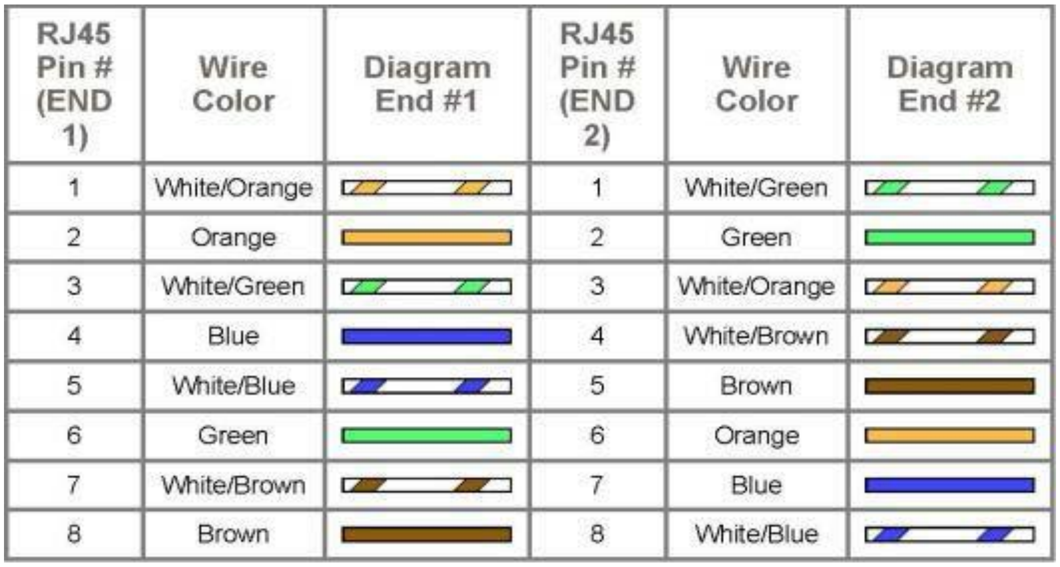
3. You have 2 end jacks, which must be installed on your cable. If you are using a pre-made cable, with one of the ends whacked off, you only have one end to install the crossed over e nd. Below are two diagrams, which show how you need to arrange the cables for each type of cable end. Decide at this point which end you are making and examine the associa ted picture below.

Diagram shows you how to prepare Cross wired connection



RJ45 Pin # (END 1)	Wire Color	Diagram End #1	RJ45 Pin # (END 2)	Wire Color	Diagram End #2
1	White/Orange	⬜/🟧	1	White/Green	⬜/🟩
2	Orange	🟧	2	Green	🟩
3	White/Green	⬜/🟩	3	White/Orange	⬜/🟧
4	Blue	🟦	4	White/Brown	⬜/🟫
5	White/Blue	⬜/🟦	5	Brown	🟫
6	Green	🟩	6	Orange	🟧
7	White/Brown	⬜/🟫	7	Blue	🟦
8	Brown	🟫	8	White/Blue	⬜/🟦

Diagram shows you how to prepare straight through wired connection



RJ45 Pin # (END 1)	Wire Color	Diagram End #1	RJ45 Pin # (END 2)	Wire Color	Diagram End #2
1	White/Orange	⬜/🟧	1	White/Green	⬜/🟩
2	Orange	🟧	2	Green	🟩
3	White/Green	⬜/🟩	3	White/Orange	⬜/🟧
4	Blue	🟦	4	White/Brown	⬜/🟫
5	White/Blue	⬜/🟦	5	Brown	🟫
6	Green	🟩	6	Orange	🟧
7	White/Brown	⬜/🟫	7	Blue	🟦
8	Brown	🟫	8	White/Blue	⬜/🟦

PROGRAM: 2

OBJECTIVE - Study of various types of network devices in detail such as Repeater, Hub, Switch, Bridge, Router and Gateway.

THEORY – Networking devices may include gateways, routers, network bridges, modems, wireless access points, networking cables, line drivers, switches, hubs, and repeaters; and may also include hybrid network devices such as multilayer switches, protocol converters, bridge routers, proxy servers, firewalls, network address translators, multiplexers, network interface controllers, wireless network interface controllers, ISDN terminal adapters and other related hardware.The most common kind of networking hardware today is a copper-based Ethernet adapter which is a standard inclusion on most modern computer systems. Wireless networking has become increasingly popular, especially for portable and handheld devices.Other networking hardware used in computers includes data center equipment (such as file servers, database servers and storage areas), network services (such as DNS, DHCP, email, etc.) as well as devices which assure content delivery.

PROCEDURE –Following should be done to understand this practical.

1. Repeater:Functioning at Physical Layer.A repeater is an electronic device that receives a signal and retransmits it at a higher level and/or higher power, or onto the other side of an obstruction, so that the signal can cover longer distances. Repeater have two ports, so cannotbe use to connect for more than two devices

2. Hub: An Ethernet hub, active hub, network hub, repeater hub, hub or concentrator
is a device for connecting multiple twisted pair or fiber optic Ethernet devices together and making them act as a single network segment. Hubs work at the physical layer (layer 1) of theOSI model. The device is a form of multiport repeater. Repeater hubs also participate incollision detection, forwarding a jam signal to all ports if it detects a collision.

3. Switch:A network switch or switching hub is a computer networking device that
Connects network segments.The term commonly refers to a network bridge that processes and routes data at the data link layer (layer 2) of the OSI model. Switches that additionally process data at the network layer (layer 3 and above) are often referred to as Layer 3 switches or multilayer switches.

4. Bridge: A network bridge connects multiple network segments at the data link layer (Layer 2) of the OSI model. In Ethernet networks, the term bridge formally means a devicethat behaves according to the IEEE 802.1 D standards. A bridge and switch are very much alike; a switch being a bridge with numerous ports. Switch or Layer 2 switch is often used interchangeably with bridge. Bridges can analyze incoming data packets to determine if the bridge is able to send the given packet to another segment of the network.

5. Router: A router is an electronic device that interconnects two or more computer Networks, and selectively interchanges packets of data between them. Each data packet contains address information that a router can use to determine if the source and destination are on the same network, or if the data packet must be transferred from one network to another. Where multiple routers are used in a large collection of interconnected networks, the routers exchange information about target system addresses, so that each router can build up a table showing the preferred paths between any two systems on the interconnected networks.

6. Gate Way: In a communications network, a network node equipped for interfacing with another network that uses different protocols.

- A gateway may contain devices such as protocol translators, impedance matching devices, rate converters, fault isolators, or signal translators as necessary to provide system interoperability. It also requires the establishment of mutually acceptable administrative procedures between both networks.
- A protocol translation/mapping gateway interconnects networks with different network protocol technologies by performing the required protocol conversions.

OUTPUT –Succefully studied about various devices and simulated it on Software Simulator (Cisco Packet Tracer).

PROGRAM: 3

OBJECTIVE - Study of basic network command and Network configuration commands like ping, traceroute, nslookup, ARP, Telnet, FTP, etc

THEORY – All commands related to Network configuration which includes how to switch to privilege mode and normal mode and how to configure router interface and how to save this configuration to flash memory or permanent memory.

This commands includes

- Configuring the Router commands
- General Commands to configure network
- Privileged Mode commands of a router
- Router Processes & Statistics
- IP Commands
- Other IP Commands e.g. show ip route etc.

PROCEDURE - To do this EXPERIMENT- follows these steps:

Students have to understand basic networking commands e.g ping, tracert etc.

ping:

ping(8) sends an ICMP ECHO_REQUEST packet to the specified host. If the host responds, you get an ICMP packet back. Sound strange? Well, you can "ping" an IP address to see if a machine is alive. If there is no response, you know something is wrong.

Traceroute:

Tracert is a command which can show you the path a packet of information takes from your

computer to one you specify. It will list all the routers it passes through until it reaches its destination, or fails to and is discarded. In addition to this, it will tell you how long each 'hop' from router to router takes.

nslookup:

Displays information from Domain Name System (DNS) name servers.

NOTE :If you write the command as above it shows as default your pc's server name firstly.

pathping:

A better version of tracert that gives you statics about packet lost and latency.In any command mode, you can get a list of available commands by entering a question mark (?).

Router>?

To obtain a list of commands that begin with a particular character sequence, type in those

haracters followed immediately by the question mark (?).

Router#co?

configure connect copy

To list keywords or arguments, enter a question mark in place of a keyword or argument. Include a space before the question mark.

Router#configure ?

memory Configure from NV memory

network Configure from a TFTP network host

terminal Configure from the terminal

You can also abbreviate commands and keywords by entering just enough characters to make the command unique from other commands. For example, you can abbreviate the show command to sh.

Configuration Files

Any time you make changes to the router configuration, you must save the changes to memory because if you do not they will be lost if there is a system reload or power outage. There are twotypes of configuration files: the running (current operating) configuration and the startupconfiguration.

Use the following privileged mode commands to work with configuration files.

- configure terminal – modify the running configuration manually from the terminal.
- show running-config – display the running configuration.
- show startup-config – display the startup configuration.
- copy running-configstartup-config – copy the running configuration to the startup configuration.
- copy startup-config running-config – copy the startup configuration to the running configuration.
- erase startup-config – erase the startup-configuration in NVRAM.
- copy tftp running-config – load a configuration file stored on a Trivial File Transfer Protocol (TFTP) server into the running configuration.
- copy running-configtftp – store the running configuration on a TFTP server.

IP Address Configuration

Take the following steps to configure the IP address of an interface.

Step 1: Enter privileged EXEC mode:

Router>enable password

Step 2: Enter the configure terminal command to enter global configuration mode.

Router#config terminal

Step 3: Enter the interface type slot/port (for Cisco 7000 series) or interface type port (for Ci

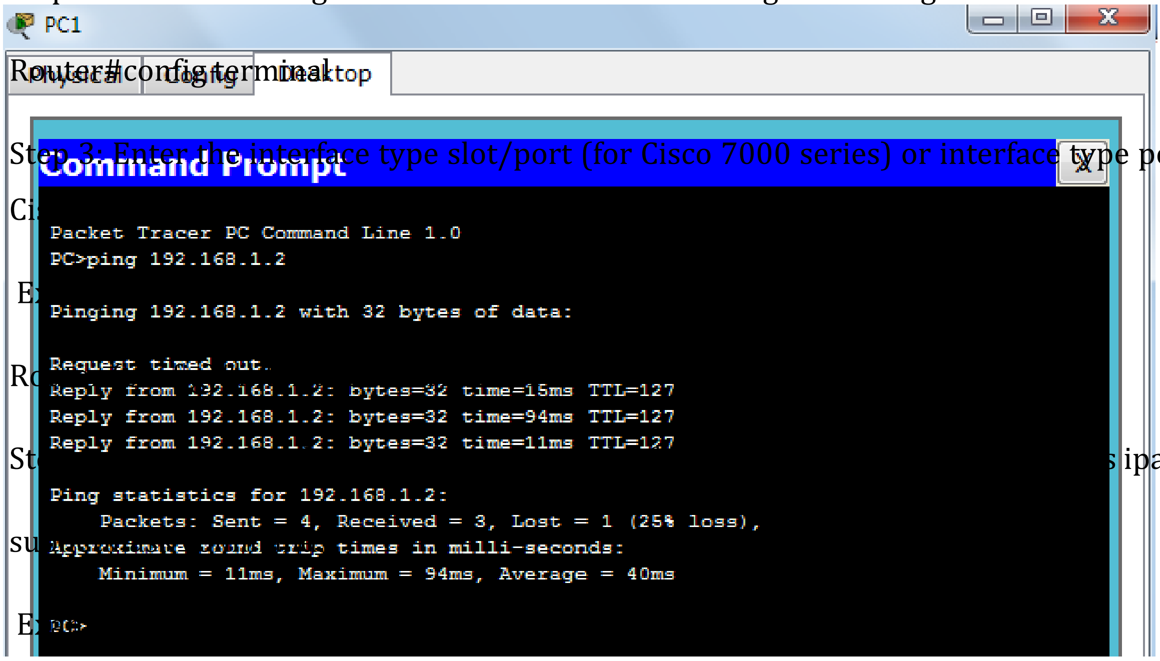
E

Ro

St ipaddress

su

E



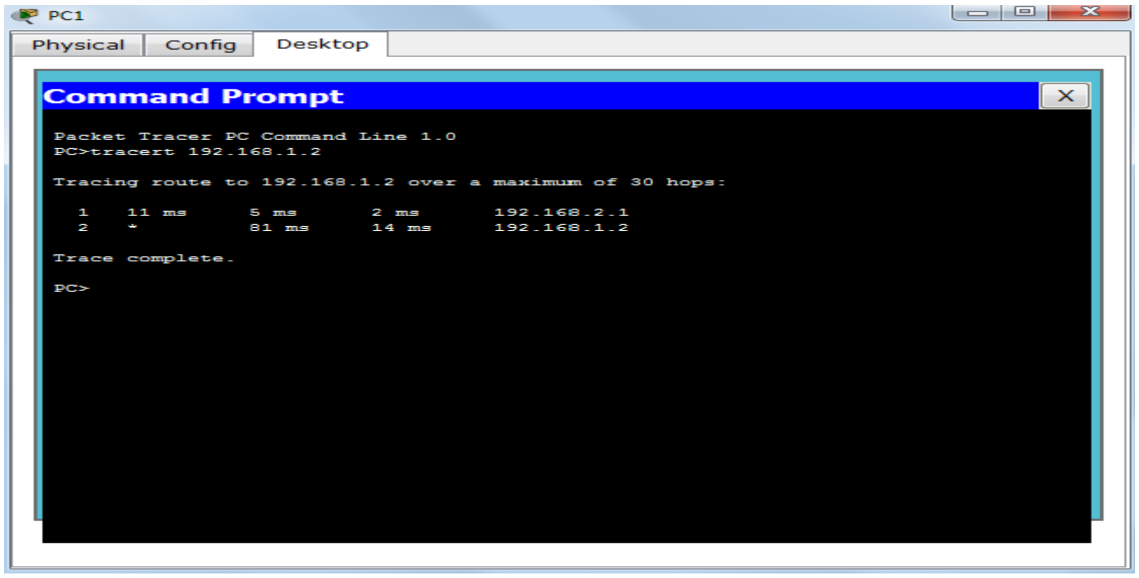
```
Packet Tracer PC Command Line 1.0
PC>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:

Request timed out.
Reply from 192.168.1.2: bytes=32 time=15ms TTL=127
Reply from 192.168.1.2: bytes=32 time=94ms TTL=127
Reply from 192.168.1.2: bytes=32 time=11ms TTL=127

Ping statistics for 192.168.1.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 11ms, Maximum = 94ms, Average = 40ms

PC>
```

Router (config-if)#ip address 192.168.10.1 255.255.255.0

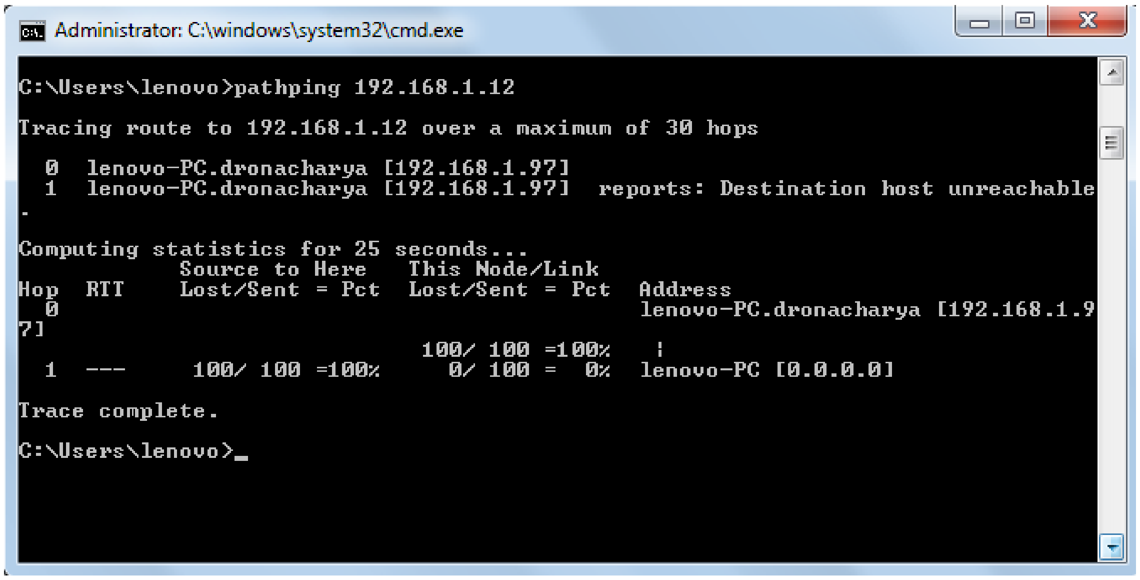
Step 5: Exit the configuration mode by pressing Ctrl-Z

Router(config-if)#[Ctrl-Z]

OUTPUT: The following oututs would be observed when running the above networking commands:



```
Packet Tracer PC Command Line 1.0
PC>tracert 192.168.1.2

Tracing route to 192.168.1.2 over a maximum of 30 hops:

  1   11 ms     5 ms      2 ms      192.168.2.1
  2   *         81 ms     14 ms     192.168.1.2

Trace complete.

PC>
```


Administrator: C:\windows\system32\cmd.exe

```
C:\Users\lenovo>pathping 192.168.1.12

Tracing route to 192.168.1.12 over a maximum of 30 hops

  0  lenovo-PC.dronacharya [192.168.1.97]
  1  lenovo-PC.dronacharya [192.168.1.97]  reports: Destination host unreachable
.

Computing statistics for 25 seconds...
            Source to Here   This Node/Link
Hop  RTT     Lost/Sent = Pct  Lost/Sent = Pct  Address
  0                                            lenovo-PC.dronacharya [192.168.1.9
7]
                              100/ 100 =100%    |
  1  ---      100/ 100 =100%     0/ 100 =  0%  lenovo-PC [0.0.0.0]

Trace complete.

C:\Users\lenovo>_
```

PROGRAM: 4

OBJECTIVE - Configure a Network topology using packet tracer software.

THEORY – Cisco Packet Tracer is a network simulation program that gives students the opportunity to experiment and learn the different behaviors of networks and asks “what if” questions. It is also a vital part of the Networking Academy learning experience. The Packet Tracer provides simulation, visualization, authoring, assessment, and enhances the teaching and learning of complex technology concepts.

PROCEDURE: To implement this practical following network topology following steps are to be followed.

Step 1: Open your Network Topology. Once you've opened your Network Topology on Cisco Packet Tracer, access your network and identify the components of your network, for example; Servers, Routers, End Devices, etc.

Step 2:Complete the cabling. Access the cables section and connect completely and correctly the cables between the network in order to ensure connectivity between the devices in the network using the connections table given.

Step 3:Configure the IP addresses on the end devices. Using the address table still, correctly and completely configure the IP addresses on all end devices. This can be done by accessing the desktop platform on each device and locating the IP configuration section. The reason for doing this is to enable the devices be on the right network.

Step 4:Configure the IP addresses on your routers and switches. After configuring the right IP addresses on the end devices, you will have to do the same on the routers and switches also, using the address table. But this time in a different way because there's no desktop platform on the routers and switches. You will have to access the configuration panel on both devices and this can be done in two ways:

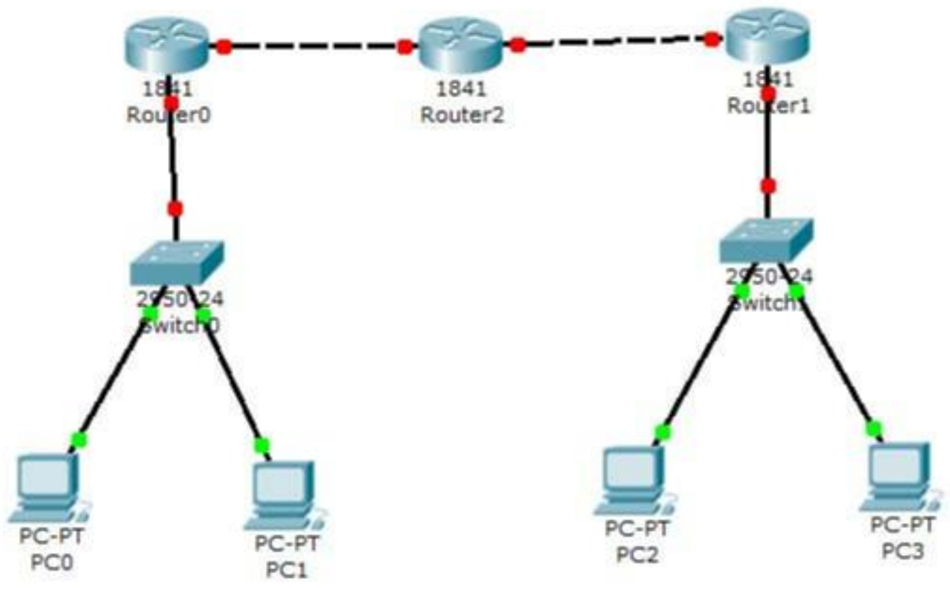
- Click on the device and open the Command Line Interface (CLI) and then type in the right commands to configure the right addresses for the router using the addressing table.

- Use a console cable from an end device and connect it to the device you wish to configure and access the terminal platform on the end device and it will take you to the device's Command Line Interface and then you type in the commands in other to configure the right addresses.

Step 5: Configure your default gateway. After configuring the IP addresses, you will need to configure the default gateway also. The reason for this is so the end devices would know what network they are operating on. You can find the default gateway either in the addressing table (if given) or in the network topology.

Step 6: Test connectivity. After configuring the addresses, you will have to test connectivity by opening a command prompt window on the end devices and try pinging the address which the network operates on. If it gives you a reply, it means your network was configured correctly.

OUTPUT – The following network is made using the Cisco packet tracer.



PROGRAM: 5

OBJECTIVE - Implementation of echo server and client using TCP sockets.

THEORY –TCP socket routines enable reliable IP communication using the transmission control protocol (TCP). This section describes the implementation of the Transmission Control Protocol (TCP) in the Network Component. TCP runs on top of the Internet Protocol (IP). TCP is a connection-oriented and reliable, full duplex protocol supporting a pair of byte streams, one for each direction. A TCP connection must be established before exchanging data. TCP retransmits data that do not reach the final destination due to errors or data corruption. Data is delivered in the sequence of its transmission. Applications that do not require the reliability of a TCP connection may instead use the connectionless User Datagram Protocol (UDP).

PROCEDURE –

Server:

1. Create a server socket.
2. Wait for client to be connected.
3. Read text from the client
4. Echo the text back to the client.
5. Repeat steps 4-5 until 'bye' or 'null' is read.
6. Close the I/O streams
7. Close the server socket
8. Stop

Client:

1. Create a socket and establish connection with the server
2. Get input from user.
3. If equal to bye or null, then go to step 7.
4. Send text to the server.
5. Display the text echoed by the server
6. Repeat steps 2-4
7. Close the I/O streams

8. Close the client socket

9. Stop

OUTPUT -

Server:

```
$ javac tcpechoserver.java
$ javatcpechoserver
Server Ready
Client Connected
Client [ hello ]
Client [ how are you ]
Client [ i am fine ]
Client [ ok ]
Client Disconnected
```

Client:

```
$ javac tcpechoclient.java
$ javatcpechoclient
Type "bye" to quit
Enter msg to server : hello
Server [ hello ]
Enter msg to server : how are you
Server [ how are you ]
Enter msg to server : i am fine
Server [ i am fine ]
Enter msg to server : ok
Server [ ok ]
Enter msg to server : bye
```

PROGRAM: 6

OBJECTIVE - Implementation of echo server and client using UDP sockets.

THEORY –UDP socket routines enable simple IP communication using the user datagram protocol (UDP). The User Datagram Protocol (UDP) runs on top of the Internet Protocol (IP) and was developed for applications that do not require reliability, acknowledgment, or flow control features at the transport layer. This simple protocol provides transport layer addressing in the form of UDP ports and an optional checksum capability. UDP is a very simple protocol. Messages, so called datagrams, are sent to other hosts on an IP network without the need to set up special transmission channels or data paths beforehand. The UDP socket only needs to be opened for communication. It listens for incoming messages and sends outgoing messages on request.

PROCEDURE / ALGORITHM –

Server:

1. Create an array of hosts and its ip address in another array
2. Create a datagram socket and bind it to a port
3. Create a datagram packet to receive client request
4. Read the domain name from client to be resolved
5. Lookup the host array for the domain name
6. If found then retrieve corresponding address
7. Create a datagram packet and send ip address to client
8. Repeat steps 3-7 to resolve further requests from clients
9. Close the server socket
10. Stop

Client:

1. Create a datagram socket
2. Get domain name from user
3. Create a datagram packet and send domain name to the server
4. Create a datagram packet to receive server message
5. Read server's response

6. If ipaddress then display it else display "Domain does not exist"
7. Close the client socket
8. Stop

OUTPUT –

Server:

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

Client:

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

PROGRAM: 7

OBJECTIVE - Send and receive message between client and server using TCP.

THEORY –ServerSocket:

This class implements server sockets. A server socket waits for requests to come in over the network. It performs some operation based on that request, and then possibly returns a result to the requester.

Methods:

ServerSocket()	Creates an unbound server socket
ServerSocket(int port)	Creates a server socket, bound to specified port
accept()	Listens for a connection to be made to this socket and accepts it.
getInetAddress()	Returns the local address of this server socket.
getLocalPart()	Returns the port on which this socket is listening.
close()	Closes this socket

PROCEDURE /ALGORITHM –

TCP Client

- In the TCP client a socket is created.
- Using the socket a connection is made to the server using the connect ( ) function.
- After a connection is established, we send messages input from the user and display the data received from the server using send( ) and read( ) functions.

Server-client communication using TCP/IP

The following tasks are done at client side:

· Create a socket for communication

· Configure TCP protocol with IP address of server and port number

· Connect with server through socket

- Wait for acknowledgement from server
- Send message to the server
- Receive message from server

The following tasks are done at server side:

- Create a socket for communication
- Bind the local port and connection address
- Configure TCP protocol with port number
- Listen for client connection
- Accept connection from client
- Send Acknowledgement
- Receive message from client
- Send message to the client

OUTPUT –

```
CLIENT:
D:\Program Files\Java\jdk1.6.0_20\bin>javacex14client.java
D:\Program Files\Java\jdk1.6.0_20\bin> java ex14client
Connection accepted.....
'echoing ..lines#0
'echoing ..lines#1
'echoing ..lines#2
'echoing ..lines#3
'echoing ..lines#4
closing
SERVER:
D:\Program Files\Java\jdk1.6.0_20\bin>javacex14server.java
```

```
D:\Program Files\Java\jdk1.6.0_20\bin> java ex14server
lines#0
lines#1
lines#2
lines#3
lines#4
closing
```

PROGRAM: 8

OBJECTIVE - Send and receive message between client and server using UDP.

THEORY –Datagram Socket: This class represents a socket for sending and receiving datagram packets. A datagram socket is the sending or receiving point for a packet delivery service. Each packet sent or received on a datagram socket is individually addressed and routed. Multiple packets sent from one machine to another may be routed differently, and may arrive in any order. User Datagram Protocol (UDP) broadcasts always enabled on a DatagramSocket. In order to receive broadcast packets a DatagramSocket should be bound to the wildcard address. In some implementations, broadcast packets may also be received when a DatagramSocket is bound to a more specific address.

DatagramPacket:

Datagram packets are used to implement a connectionless packet delivery service. Multiple packets sent from one machine to another might be routed differently, and might arrive in any order. Packet delivery is not guaranteed. Methods:

DatagramSocket(int port)	Constructs a datagram socket and binds it to any available port on the local host machine.
receive(DatagramPacket p)	Receives a datagram packet from this socket.
send(DatagramPacket p)	Sends a datagram packet from this socket.
DatagramPacket(byte[] buf, int length)	Constructs a DatagramPacket for receiving packets of length.
DatagramPacket(byte[] buf,int length,	Constructs a datagram packet for sending packets of length to the specified port

InetAddress address, int port	number on the specified host.
getData()	Returns the data buffer

PROCEDURE / ALGORITHM:

The following tasks are done at server side program:

- Define the necessary library file at server side

- Create a socket for end point communication

- Define the UDP protocol (i.e SOCK_DGRAM) and bind the communication port (in this case: port number is 5000).

- After successful binding, server waits to receive the data from client

- If data is received from client, the server displays it on the command terminal.

The following tasks are done at client side program:

- Define the necessary library file at client side

- Create a socket for end point communication

- Define the UDP protocol (i.e SOCK_DGRAM) and bind the communication port (in this case: port number is 5000).

- After successful binding, the client can communicate with the server.

- Now, the user can send the data from terminal.

OUTPUT (Sample)–

Server:

```
D:\Program Files\Java\jdk1.6.0_20\bin>javac ex15server.java
D:\Program Files\Java\jdk1.6.0_20\bin> java ex15server
```

```
Server started
HELLO SERVER-FROM CLIENT
End of Reception
End of Sending
```

Client:

```
D:\Program Files\Java\jdk1.6.0_20\bin>javac ex15client.java
D:\Program Files\Java\jdk1.6.0_20\bin> java ex15client
End of Sending
```

PROGRAM: 9

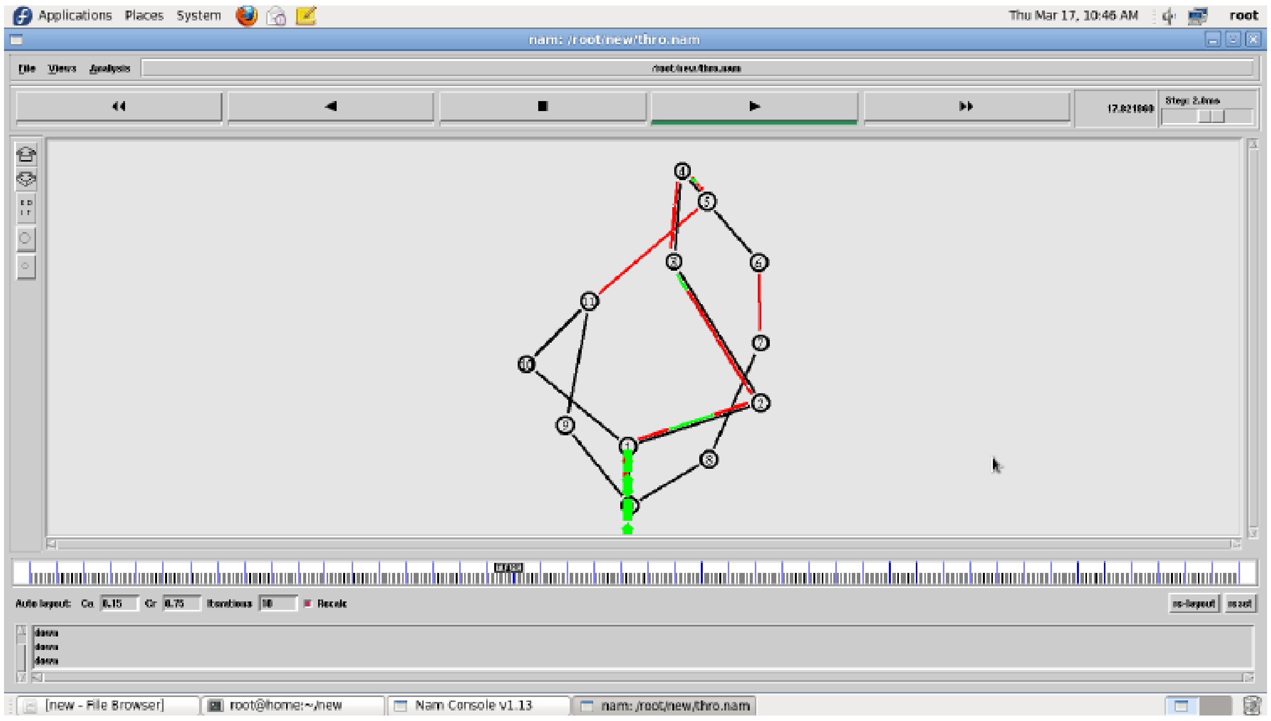
OBJECTIVE - Configure Network using Link State Vector Routing protocol.

THEORY - In link state routing, each router shares its knowledge of its neighborhood with every other router in theinternet work. (i) Knowledge about Neighborhood: Instead of sending its entire routing table a router sendsinfo about its neighborhood only. (ii) To all Routers: each router sends this information to every other routeron the internet work not just to its neighbor .It does so by a process called flooding. (iii)Information sharingwhen there is a change: Each router sends out information about the neighbors when there is change.

PROCEDURE -The Dijkstra algorithm follows four steps to discover what is called the shortest path tree(routing table) foreach router:The algorithm begins to build the tree by identifying its roots. The root router's trees the routeritself. The algorithm then attaches all nodes that can be reached from the root. The algorithm compares thetree's temporary arcs and identifies the arc with the lowest cumulative cost. This arc and the node to which itconnects are now a permanent part of the shortest path tree. The algorithm examines the database and identifiesevery node that can be reached from its chosen node. These nodes and their arcs are added temporarily to thetree.The last two steps are repeated until every node in the network has become a permanent part of the tree. The steps include :

1. Create a simulator object
2. Define different colors for different data flows
3. Open a nam trace file and define finish procedure then close the trace file, and execute nam on trace file.
4. Create n number of nodes using for loop
5. Create duplex links between the nodes
6. Setup UDP Connection between n(0) and n(5)
7. Setup another UDP connection between n(1) and n(5)
8. Apply CBR Traffic over both UDP connections
9. Choose Link state routing protocol to transmit data from sender to receiver.
10. Schedule events and run the program

OUTPUT –



```
Auto layout:  Ca 0.15   Cr 0.75   Iterations 10   Recalc
```

```
down
down
down
```

PROGRAM: 10

OBJECTIVE - Study of sliding window protocol.

THEORY - A sliding window protocol is a feature of packet-based data transmission protocols. Sliding window protocols are used where reliable in-order delivery of packets is required, such as in the Data Link Layer (OSI model) as well as in the Transmission Control Protocol (TCP). Conceptually, each portion of the transmission (packets in most data link layers, but bytes in TCP) is assigned a unique consecutive sequence number, and the receiver uses the numbers to place received packets in the correct order, discarding duplicate packets and identifying missing ones. The problem with this is that there is no limit on the size of the sequence number that can be required. By placing limits on the number of packets that can be transmitted or received at any given time, a sliding window protocol allows an unlimited number of packets to be communicated using fixed-size sequence numbers. The term "window" on the transmitter side represents the logical boundary of the total number of packets yet to be acknowledged by the receiver. The receiver informs the transmitter in each acknowledgment packet the current maximum receiver buffer size (window boundary). The TCP header uses a 16 bit field to report the receive window size to the sender. Therefore, the largest window that can be used is 216 = 64 kilobytes. In slow-start mode, the transmitter starts with low packet count and increases the number of packets in each transmission after receiving acknowledgment packets from receiver. For every ack packet received, the window slides by one packet (logically) to transmit one new packet. When the window threshold is reached, the transmitter sends one packet for one ack packet received. If the window limit is 10 packets then in slow start mode the transmitter may start transmitting one packet followed by two packets (before transmitting two packets, one packet ack has to be received), followed by three packets and so on until 10 packets. But after reaching 10 packets, further transmissions are restricted to one packet transmitted for one ack packet received. In a simulation this appears as if the window is moving by one packet distance for every ack packet received. On the receiver side also the window moves one packet for every packet received. The sliding window method ensures that traffic congestion on the network is avoided. The application layer will still be offering data for transmission to TCP without worrying about the network traffic congestion issues as the TCP on sender and receiver side implement sliding windows of packet buffer. The window size may vary dynamically depending on network traffic. For the highest possible throughput, it is important that the transmitter is not forced to stop sending by the sliding window protocol earlier than one round-trip delay time (RTT). The limit on the amount of data that it can send before stopping to wait for an acknowledgment should be larger than the bandwidth-delay product of the communications link. If it is not, the protocol will limit the effective bandwidth of the link.

PROCEDURE /ALGORITHM -

1. Create a simulator object

2. Define different colors for different data flows

3. Open a nam trace file and define finish procedure then close the trace file, and execute nam on trace file.

4. Create two nodes that forms a network numbered 0 and 1

5. Create duplex links between the nodes to form a STAR Topology

6. Setup TCP Connection between n(1) and n(3)

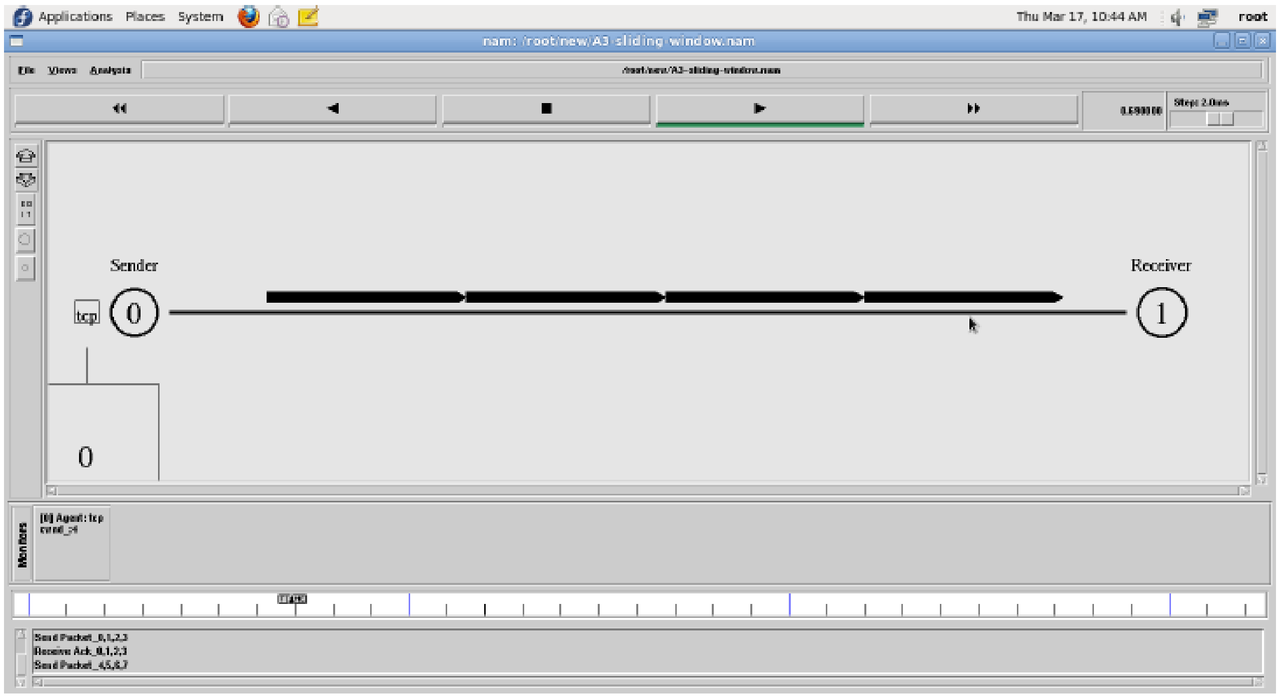
7. Apply CBR Traffic over TCP

8. Schedule events and run the program.

```
# sliding window mechanism with some features
# such as labeling, annotation, nam-graph, and window size monitoring
set ns [new Simulator]
set n0 [$ns node]
set n1 [$ns node]
$ns at 0.0 "$n0 label Sender"
$ns at 0.0 "$n1 label Receiver"
setnf [open sliding.nam w]
$ns namtrace-all $nf
set f [open sliding.tr w]
$ns trace-all $f
```

```
$ns duplex-link $n0 $n1 0.2Mb 200ms DropTail
$ns duplex-link-op $n0 $n1 orient right
$ns queue-limit $n0 $n1 10
Agent/TCP set nam_tracevar_ true
settcp [new Agent/TCP]
$tcp set windowInit_ 4
$tcp set maxcwnd_ 4
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n1 $sink
$ns connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns monitor-agent-trace $tcp
$tcptracevarcwnd_
$ns at 0.1 "$ftp start"
$ns at 3.0 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n1 $sink"
$ns at 3.5 "finish"
$ns at 0.0 "$ns trace-annotate \"Sliding Window with window size 4 (normal operation)\""
$ns at 0.05 "$ns trace-annotate \"FTP starts at 0.1\""
$ns at 0.11 "$ns trace-annotate \"Send Packet_0,1,2,3\""
$ns at 0.34 "$ns trace-annotate \"Receive Ack_0,1,2,3\""
$ns at 0.56 "$ns trace-annotate \"Send Packet_4,5,6,7\""
$ns at 0.79 "$ns trace-annotate \"Receive Ack_4,5,6,7\""
$ns at 0.99 "$ns trace-annotate \"Send Packet_8,9,10,11\""
$ns at 1.23 "$ns trace-annotate \"Receive Ack_8,9,10,11 \""
$ns at 1.43 "$ns trace-annotate \"Send Packet_12,13,14,15\""
$ns at 1.67 "$ns trace-annotate \"Receive Ack_12,13,14,15\""
$ns at 1.88 "$ns trace-annotate \"Send Packet_16,17,18,19\""
$ns at 2.11 "$ns trace-annotate \"Receive Ack_16,17,18,19\""
$ns at 2.32 "$ns trace-annotate \"Send Packet_20,21,22,23\""
$ns at 2.56 "$ns trace-annotate \"Receive Ack_24,25,26,27\""
$ns at 2.76 "$ns trace-annotate \"Send Packet_28,29,30,31\""
$ns at 3.00 "$ns trace-annotate \"Receive Ack_28\""
$ns at 3.1 "$ns trace-annotate \"FTP stops\""
proc finish {} {
global ns
$ns flush-trace
# close $nf
puts "running nam..."
execnamsliding.nam&
exit 0
}
$ns run
```

OUTPUT: The sample outputs for Go Back N and Selective repeat ARQ respectively are :



```
Send Packet_0,1,2,3
Receive Ack_0,1,2,3
Send Packet_4,5,6,7
```