Computer Networks Lab

Week1: Study of different types of Network cables and Implement the cross-wired cable and straight through cable and configure the Network Topology using Packet Tracer

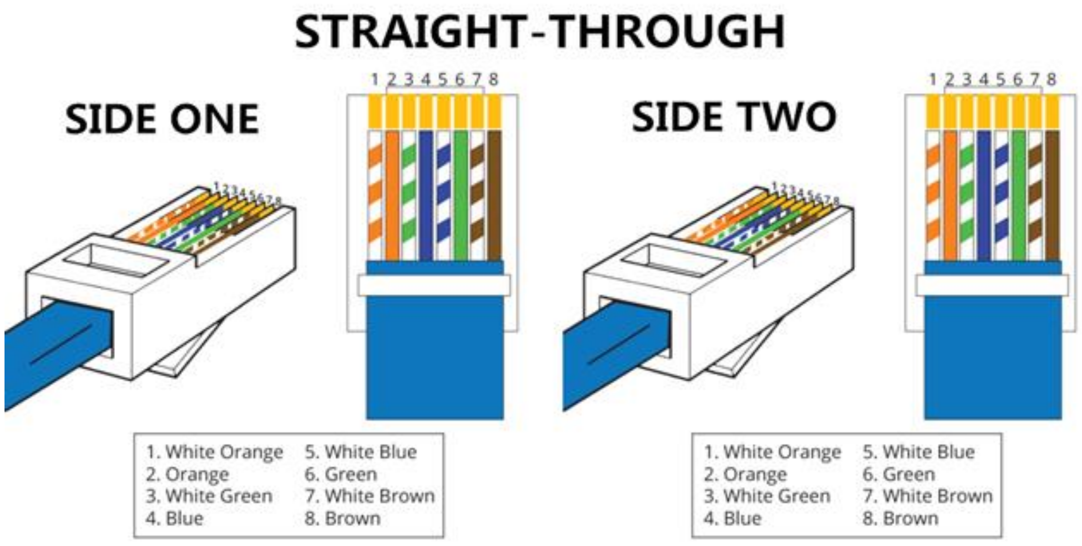
Ethernet cables can be wired as straight through or crossover.

The **straight through** Ethernet cable is the most common type and is used to connect computers to hubs or switches.

Crossover Ethernet cable is more commonly used to connect a computer to a computer

What Is Straight Through Cable?

A straight through cable is a type of twisted pair cable that is used in local area networks to connect a computer to a network hub such as a router. This type of cable is also sometimes called a patch cable and is an alternative to wireless connections where one or more computers access a router through a wireless signal. On a straight through cable, the wired pins match. Straight through cable use one wiring standard: both ends use T568A wiring standard or both ends use T568B wiring standard. The following figure shows a straight through cable of which both ends are wired as the T568B standard

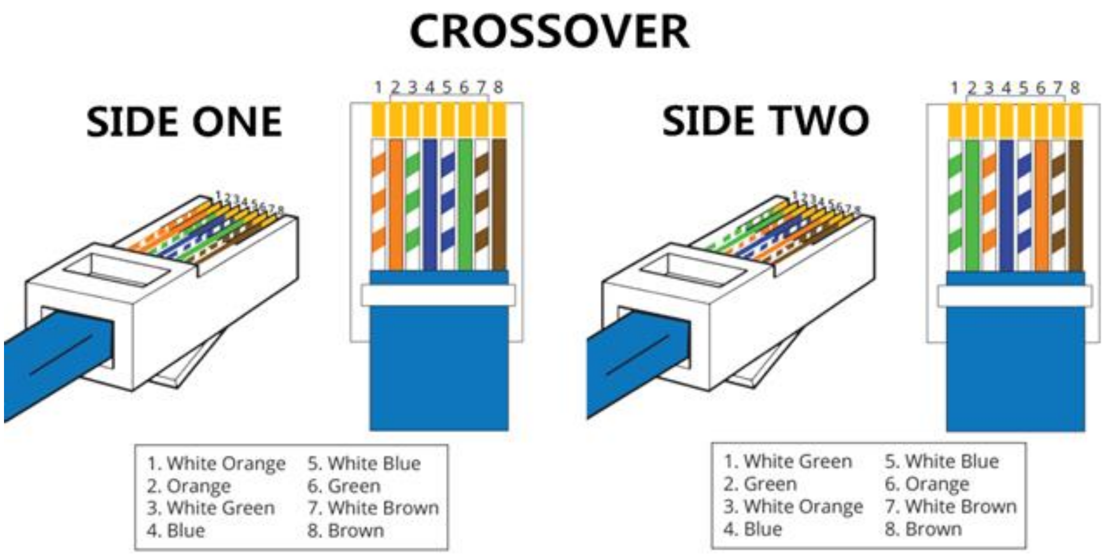


Use straight through Ethernet cable for the following cabling:

- Switch to router
- Switch to PC or server
- Hub to PC or server

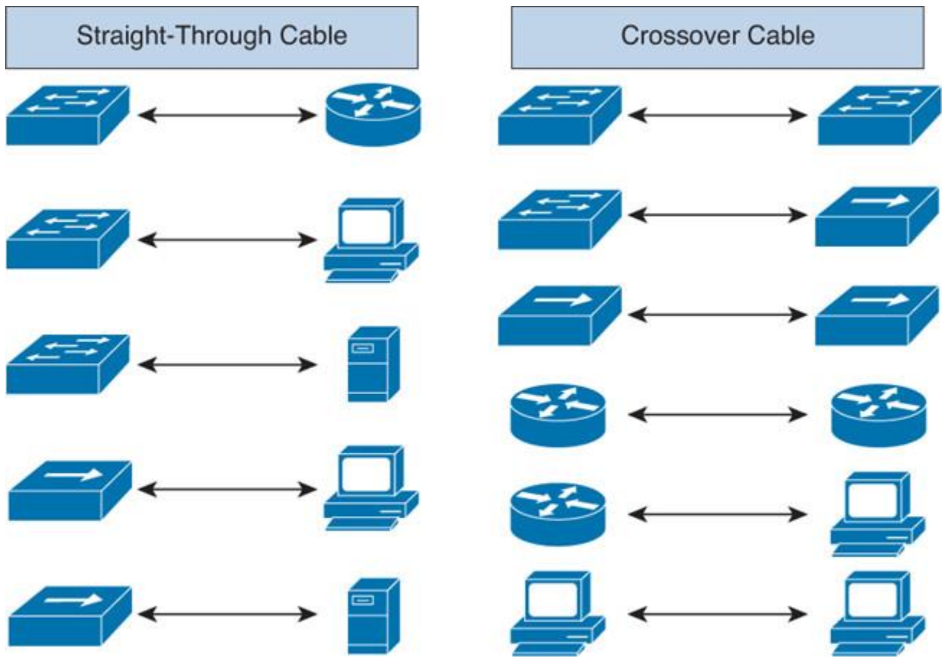
What Is Crossover Cable?

A crossover Ethernet cable is a type of Ethernet cable used to connect computing devices together directly. Unlike straight through cable, the RJ45 crossover cable uses two different wiring standards: one end uses the T568A wiring standard, and the other end uses the T568B wiring standard. The internal wiring of Ethernet crossover cables reverses the transmit and receive signals. It is most often used to connect two devices of the same type: e.g. two computers (via network interface controller) or two switches to each other.



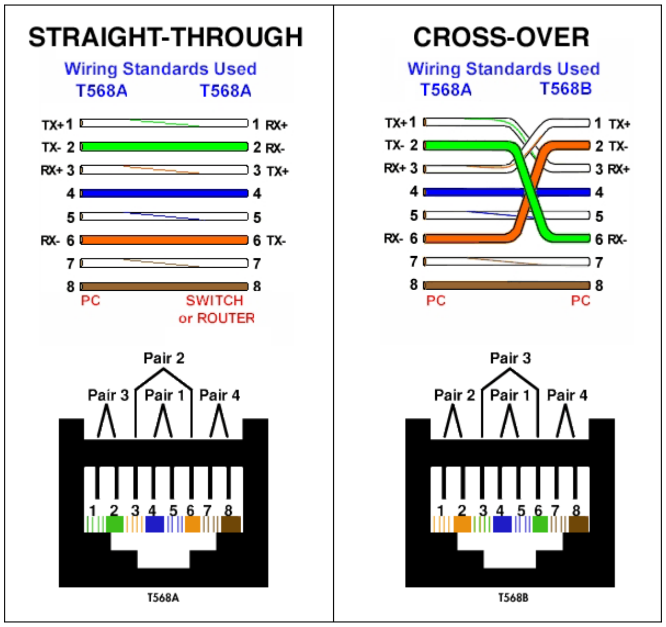
Use crossover cables for the following cabling:

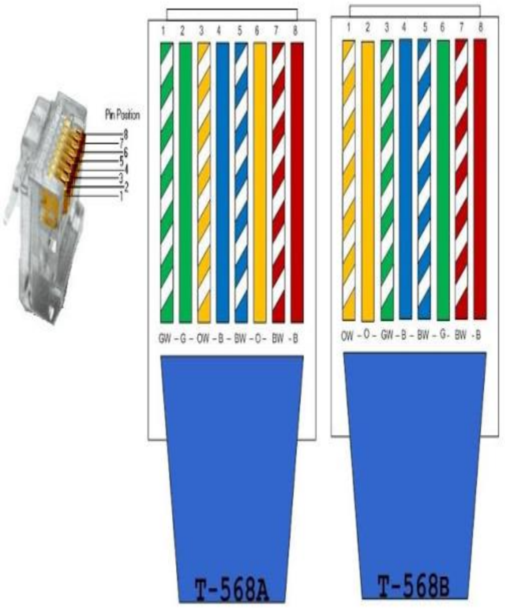
- Switch to switch
- Switch to hub
- Hub to hub
- Router to router
- Router Ethernet port to PC NIC
- PC to PC



Conclusion:

At present, the straight through cable is much more popular than crossover cable and is widely used by people. FS.COM provides a full range straight through Cat5e, Cat6, Cat6a and Cat7 Ethernet cables with many lengths and colors options.




Pin Position
1 2 3 4 5 6 7 8
GW - G - OW - B - BW - O - BW - B
OW - O - GW - B - BW - G - BW - B
T-568A
T-568B

Rollover Wired Cables

A rollover cable is a network cable that connects a computer terminal to a network router's console port.

Ex. Cisco console cable

Rollover wired cables, most commonly called rollover cables, have opposite Pin assignments on each end of the cable or, in other words,

Router Pin, Router Pin Name, Direction, Workstation Pin, Workstation Pin Name

1 – white-Orange, bi-directional, 8, brown

2 – orange, bi-directional, 7, white-brown

3 – white-green, bi-directional, 6, green

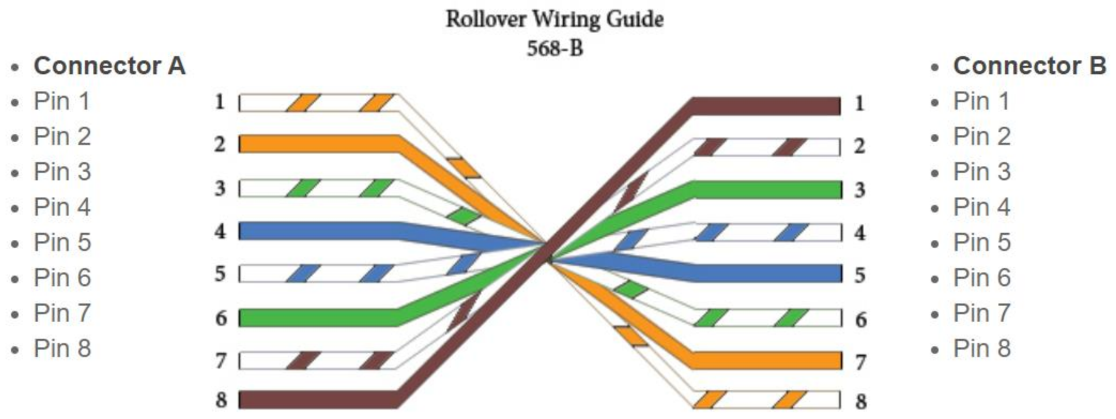
4 – blue, bi-directional, 5, white-blue

5 – white-blue, bi-directional, 4, blue

6 – green, bi-directional, 3, white-green

7 – white-brown, bi-directional, 2, orange

8 – brown, bi-directional, 1, white-orange



Reference :https://www.comparitech.com/net-admin/difference-between-straight-through-crossover-rollover-cables/

Configure the Network Topology using Packet Tracer:

Packet Tracer:

Packet Tracer is a cross-platform visual simulation tool designed by Cisco Systems that allows users to create network topologies and imitate modern computer networks.

Packet Tracer makes use of a drag and drop user interface, allowing users to add and remove simulated network devices as they see fit.

Packet Tracer can be run on Linux, Microsoft Windows, and macOS. Packet Tracer allows users to create simulated network topologies by dragging and dropping routers, switches and various other types of network devices. Packet Tracer supports an array of simulated Application Layer protocols, as well as basic routing with RIP, OSPF, EIGRP, BGP.

Packet Tracer supports a multi-user system that enables multiple users to connect multiple topologies together over a computer network.

Network Topology:

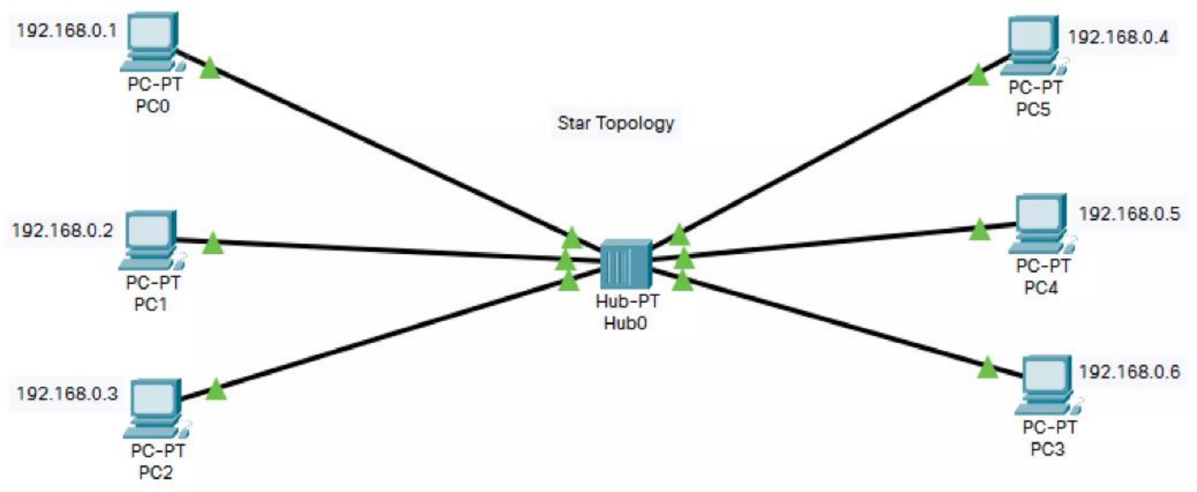
Topology defines the structure of the network of how all the components are interconnected to each other.

A network topology is the physical and logical arrangement of nodes and connections in a network. Nodes usually include devices such as switches, routers and software with switch and router.

Various Network Topologies are:

- Mesh Topology
- Star Topology
- Bus Topology
- Ring Topology
- Tree Topology
- Hybrid Topology

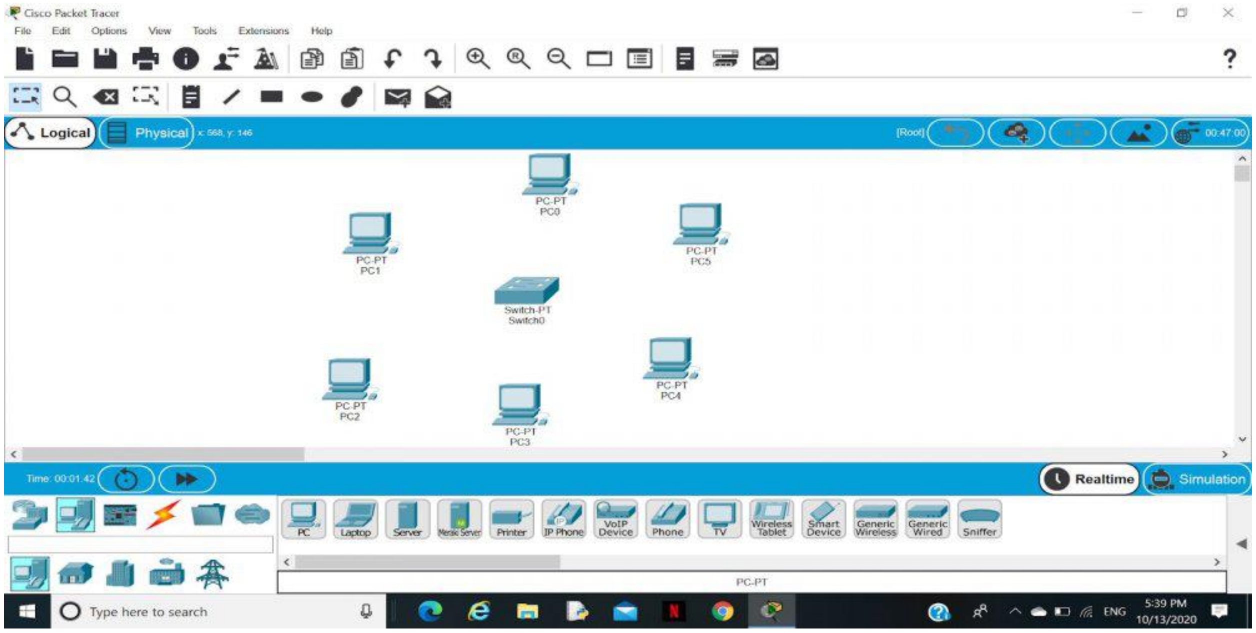
Star Topology: Star topology is an arrangement of the network in which every node is connected to the central hub, switch or a central computer.



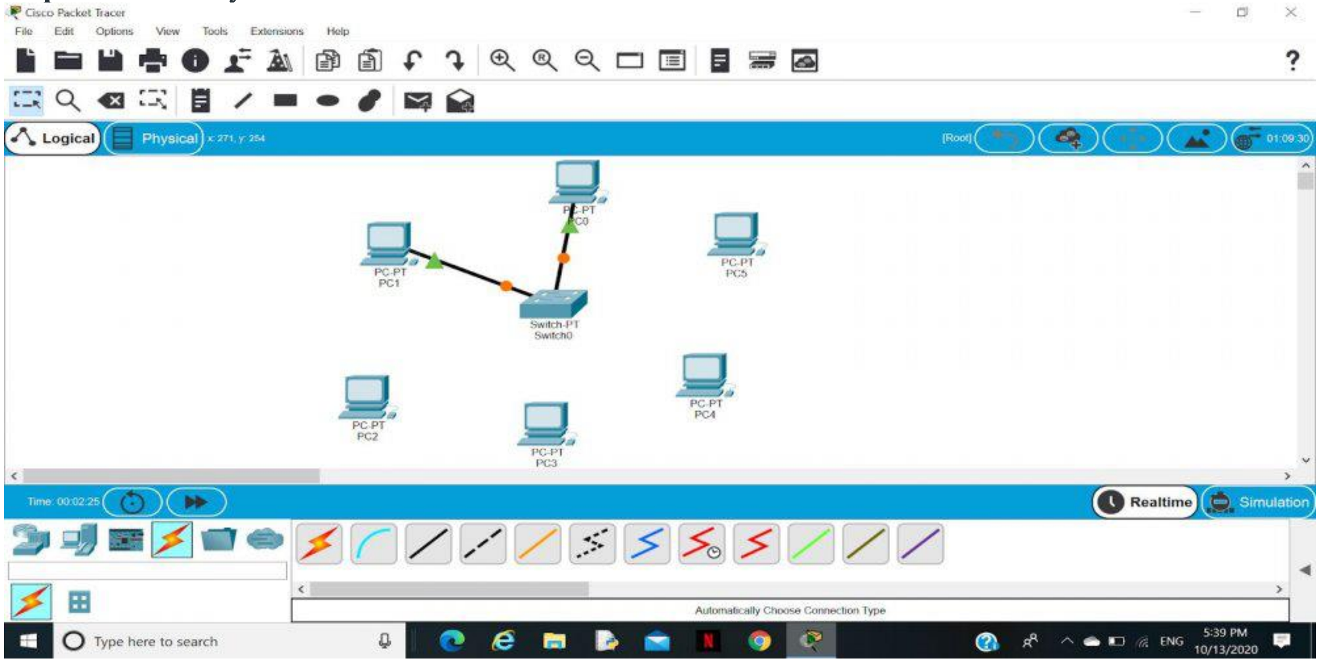
Objectives:

- Star topology network setup with “Cisco Packet Tracer”.
- Set of computer devices are connected individually to central Hub.
- Connection between Hub to Computer.
- Data transfer from any device to the others.

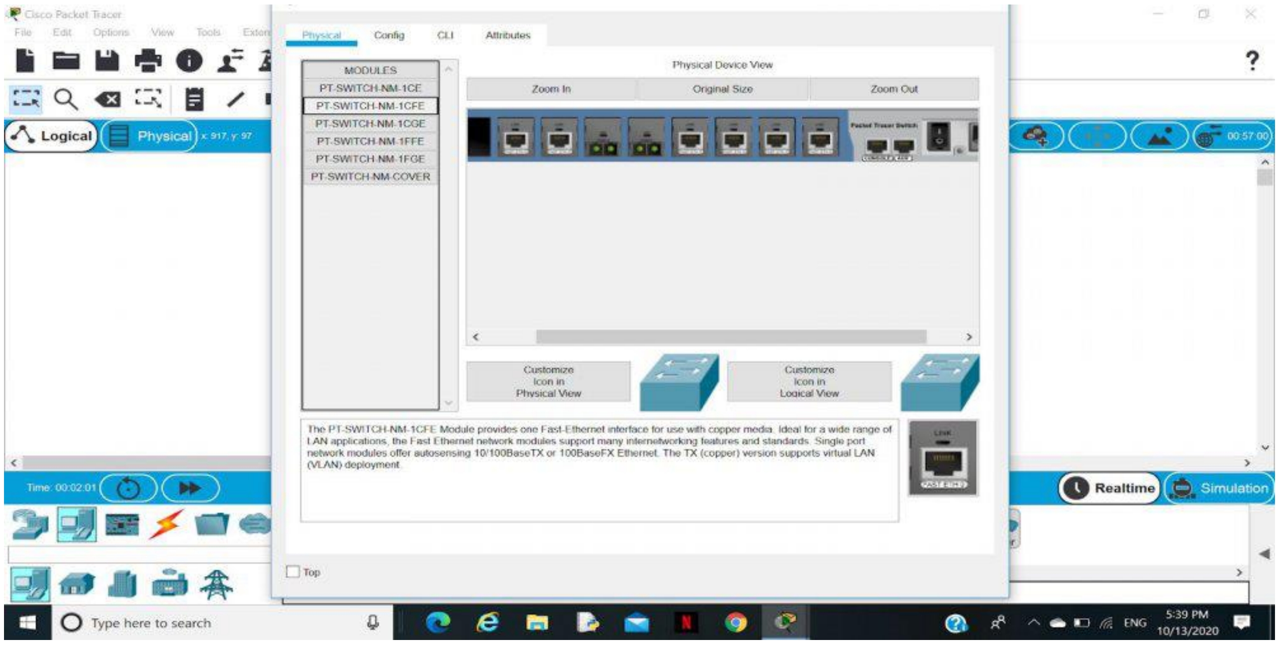
Step 1: We have taken a switch and linked it to six end devices.



Step 2: Link every device with the switch.



Step 3: Provide the IP address to each device.



The PT-SWITCH-NM-1CFE Module provides one Fast-Ethernet interface for use with copper media. Ideal for a wide range of LAN applications, the Fast Ethernet network modules support many internetworking features and standards. Single port network modules offer autosensing 10/100BaseTX or 100BaseFX Ethernet. The TX (copper) version supports virtual LAN (VLAN) deployment.

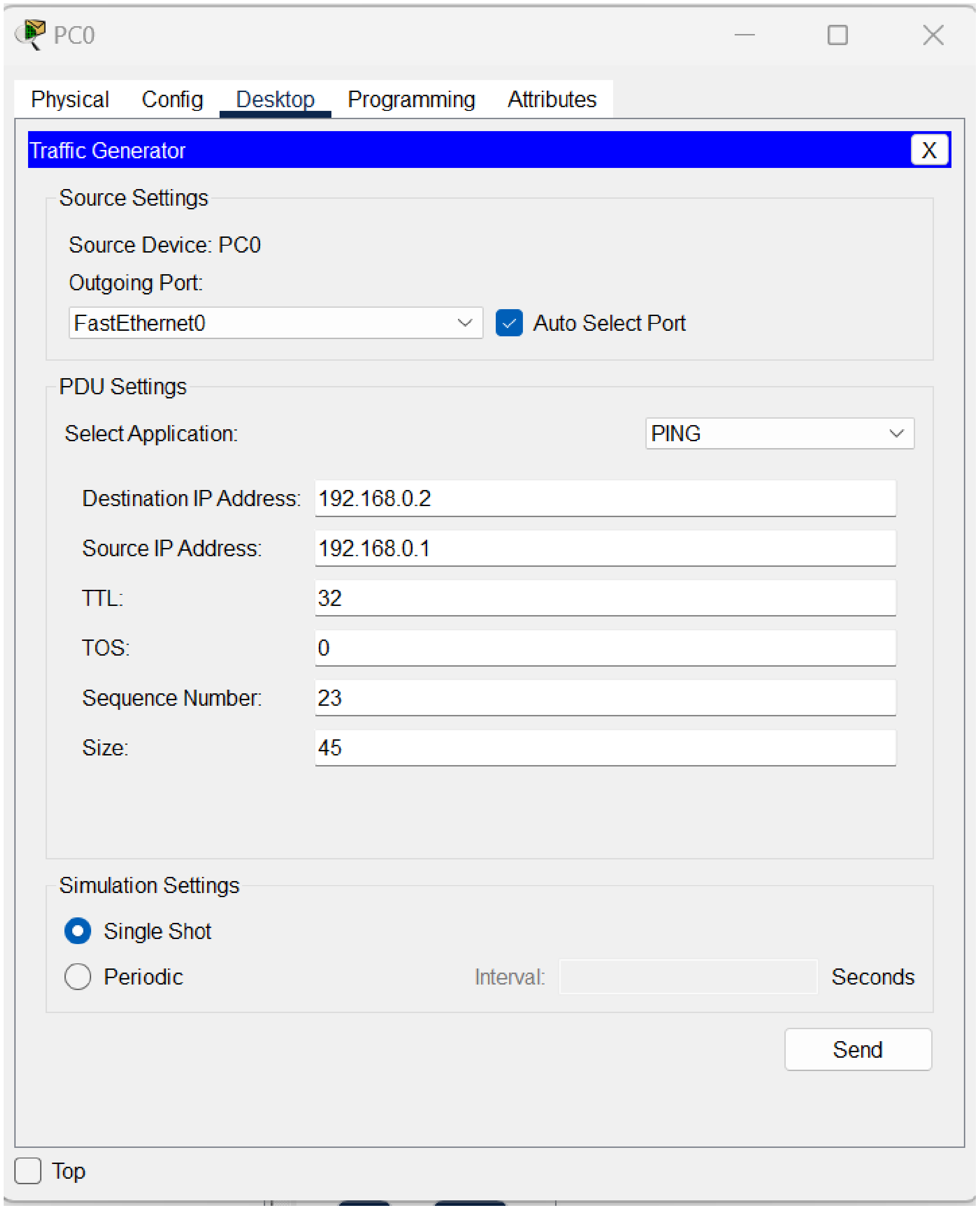
Step 4: Transfer message from one device to another and check the Table for Validation.

Now to check whether the connections are correct or not try to ping any device

To do ping one terminal of one device and run the following command:

Command:
"ping ip_address_of _any_device"

Example: ping 192.168.1.4



Source Settings

Source Device: PC0

Outgoing Port: FastEthernet0 [✓ Auto Select Port]

PDU Settings

Select Application: PING

Destination IP Address: 192.168.0.2

Source IP Address: 192.168.0.1

TTL: 32

TOS: 0

Sequence Number: 23

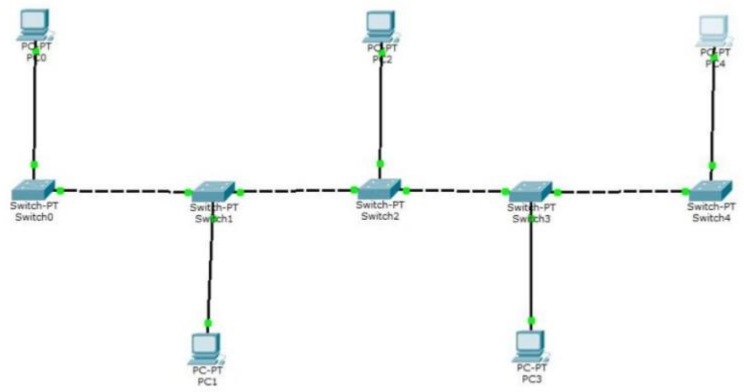
Size: 45

Simulation Settings

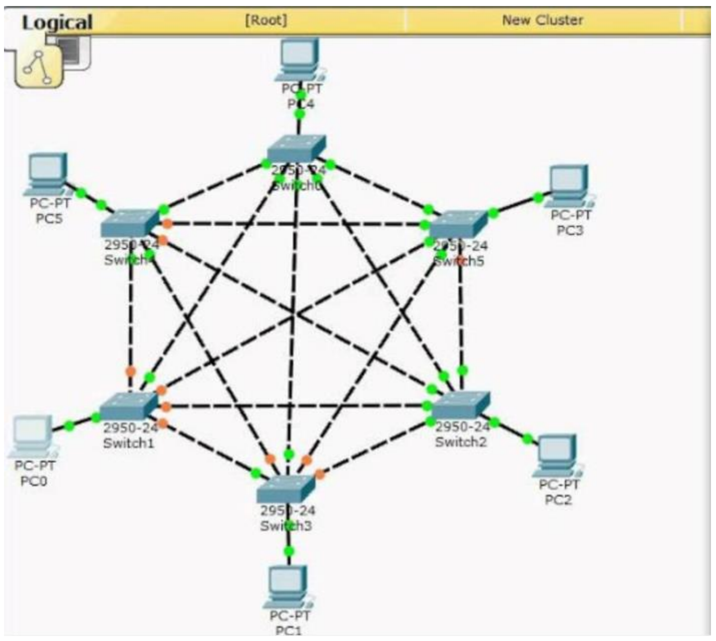
[✓ Single Shot] Periodic Interval: Seconds

Click on send..then click on simulation observe message is transferred from one device to another.

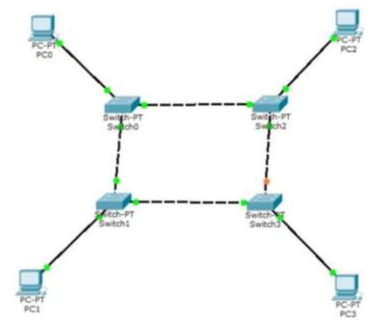
BUS TOPOLOGY



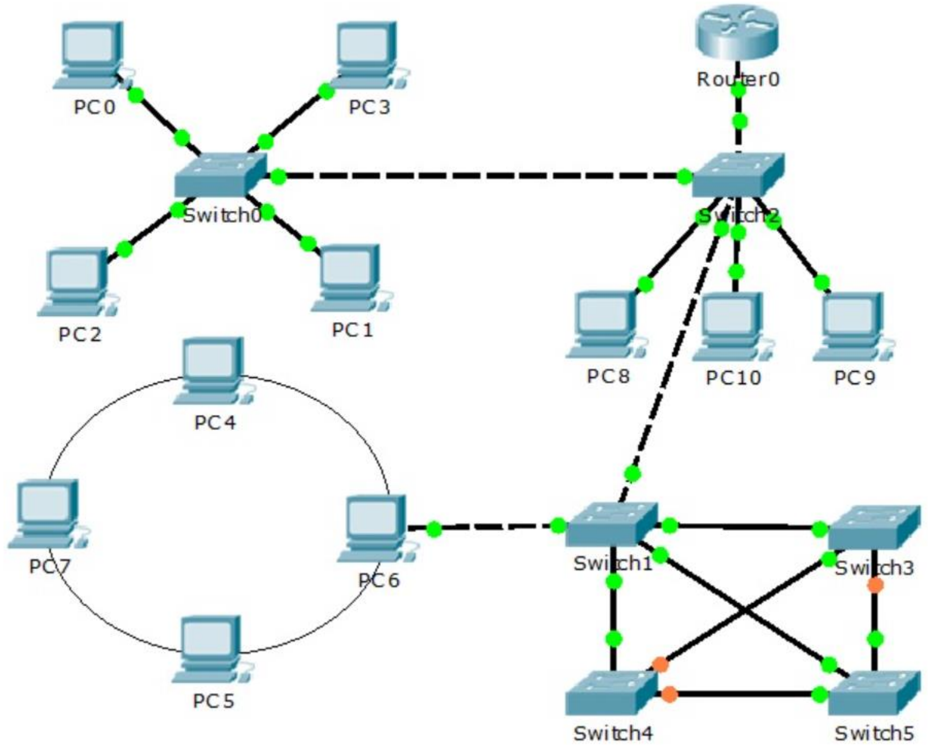
Mesh Topology



RING TOPOLOGY



Hybrid Topology



Week 2: Implement the data link layer framing methods such as character stuffing and bit stuffing.

// implement bit stuffing

```
#include<stdio.h>

#include<stdlib.h>

#define MAXSIZE 100

void main( )

{

char in[MAXSIZE];

char stuff[MAXSIZE];

char unstuff[MAXSIZE];

int count=0,j=0,i=0;

printf("enter the input character string (0ës & 1ës only):\n");

scanf("%s",in);

while(in[i]!='\0')

  {

if(in[i]=='0')

    {

stuff[j]=in[i];

i++;

j++;

    }

else

    {

while(in[i]=='1' && count!=5)
```

```
      {
            count++;
            stuff[j]=in[i];
            i++;
            j++;
      }
if(count==5)
      {
stuff[j]='0';
            j++;
      }
count=0;
    }
  }
stuff[j]='\0';
printf("\nthe stuffed character string is");
printf("01111110");
printf("\n%s",stuff);
printf("01111110");
i=0;
j=0;
while(stuff[i]!='\0')
  {
if(stuff[i]=='0')
    {
```

```
unstuff[j]=stuff[i];


i++;

j++;

   }

else

   {

while(stuff[i]=='1' && count!=5)

     {

          count++;

          unstuff[j]=stuff[i];

          i++;

          j++;

     }

if(count==5)

     {

          i++;

     }

count=0;

   }

  }

unstuff[j]='\0';

printf("\nthe unstuffed character string is");

printf("\n%s\n",unstuff);

}
```

INPUT/OUTPUT:

```
enter the input character string:
10111111011
The stuffed character string is:
01111110 101111101011 01111110
The unstuffed character string is:
10111111011
```

```
// Charater Stuffing
#include<stdio.h>
//#include<conio.h>
#include<string.h>
//#include<process.h>
void main()
{
int i=0,j=0,n,pos;
char a[20],b[50],ch;
//clrscr();
printf("enter string\n");
scanf("%s",a);
n=strlen(a);
printf("enter position\n");
scanf("%d",&pos);
if(pos>n)
{
```

```
printf("invalid position, Enter again :");

scanf("%d",&pos);

}

printf("enter the character\n");

ch=getche();

b[0]='d';

b[1]='l';

b[2]='e';

b[3]='s';

b[4]='t';

b[5]='x';

j=6;

while(i<n)

{

if(i==pos-1)

{

b[j]='d';

b[j+1]='l';

b[j+2]='e';

b[j+3]=ch;

b[j+4]='d';

b[j+5]='l';

b[j+6]='e';

j=j+7;

}
```

```
if(a[i]=='d' && a[i+1]=='l' && a[i+2]=='e')
{
b[j]='d';
b[j+1]='l';
b[j+2]='e';
j=j+3;
}
b[j]=a[i];
i++;
j++;
}
b[j]='d';
b[j+1]='l';
b[j+2]='e';
b[j+3]='e';
b[j+4]='t';
b[j+5]='x';
b[j+6]='\0';
printf("\nframe after stuffing:\n");
printf("%s",b);
//getch();
}
```

Output:

```
enter string
doodle
```

```
enter position

0

enter the character


frame after stuffing:

dlestx doodledle dleetx
```

week3:

//Implementation of Hamming code algorithm

```
#include <stdio.h>
#include <math.h>
int input[32];
int code[32];
int ham_calc(int,int);
void main()
{
	int n,i,p_n = 0,c_l,j,k;
	printf("Please enter the length of the Data Word: ");
	scanf("%d",&n);
	printf("Please enter the Data Word:\n");
	for(i=0;i<n;i++)
	{
		scanf("%d",&input[i]);
	}

	i=0;
	while(n>(int)pow(2,i)-(i+1))
	{
		p_n++;
		i++;
	}

	c_l = p_n + n;

	j=k=0;
	for(i=0;i<c_l;i++)
	{

		if(i==((int)pow(2,k)-1))
		{
			code[i]=0;
			k++;
		}
		else
		{
			code[i]=input[j];
			j++;
		}
	}
	for(i=0;i<p_n;i++)
```

```
        {
                int position = (int)pow(2,i);
                int value = ham_calc(position,c_l);
                code[position-1]=value;
        }
        printf("\nThe calculated Code Word is: ");
        for(i=0;i<c_l;i++)
                printf("%d",code[i]);
        printf("\n");
        printf("Please enter the received Code Word:\n");
        for(i=0;i<c_l;i++)
                scanf("%d",&code[i]);

        int error_pos = 0;
        for(i=0;i<p_n;i++)
        {
                int position = (int)pow(2,i);
                int value = ham_calc(position,c_l);
                if(value != 0)
                        error_pos+=position;
        }
        if(error_pos == 0)
                printf("The received Code Word is correct.\n");
        else
                printf("Error at bit position: %d\n",error_pos);
}
int ham_calc(int position,int c_l)
{
        int count=0,i,j;
        i=position-1;
        while(i<c_l)
        {
                for(j=i;j<i+position;j++)
                {
                        if(code[j] == 1)
                                count++;
                }
                i=i+2*position;
        }
        if(count%2 == 0)
                return 0;
        else
                return 1;
}
```

```
Please enter the length of the Data Word: 7
Please enter the Data Word:
```

```
1
0
1
1
0
1
1

The calculated Code Word is: 11100110011
Please enter the received Code Word:
1
1
1
1
0
1
1
0
0
1
1
Error at bit position: 4
```

Week 4: Implement on a data set of characters the three CRC polynomials – CRC 12, CRC and CCIP.

```
//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:

```
Enter Data: 1010000
Enter Key: 1001

Quotient is 1011011
Remainder is 011
Final data is: 1010000011
```

Week 5:

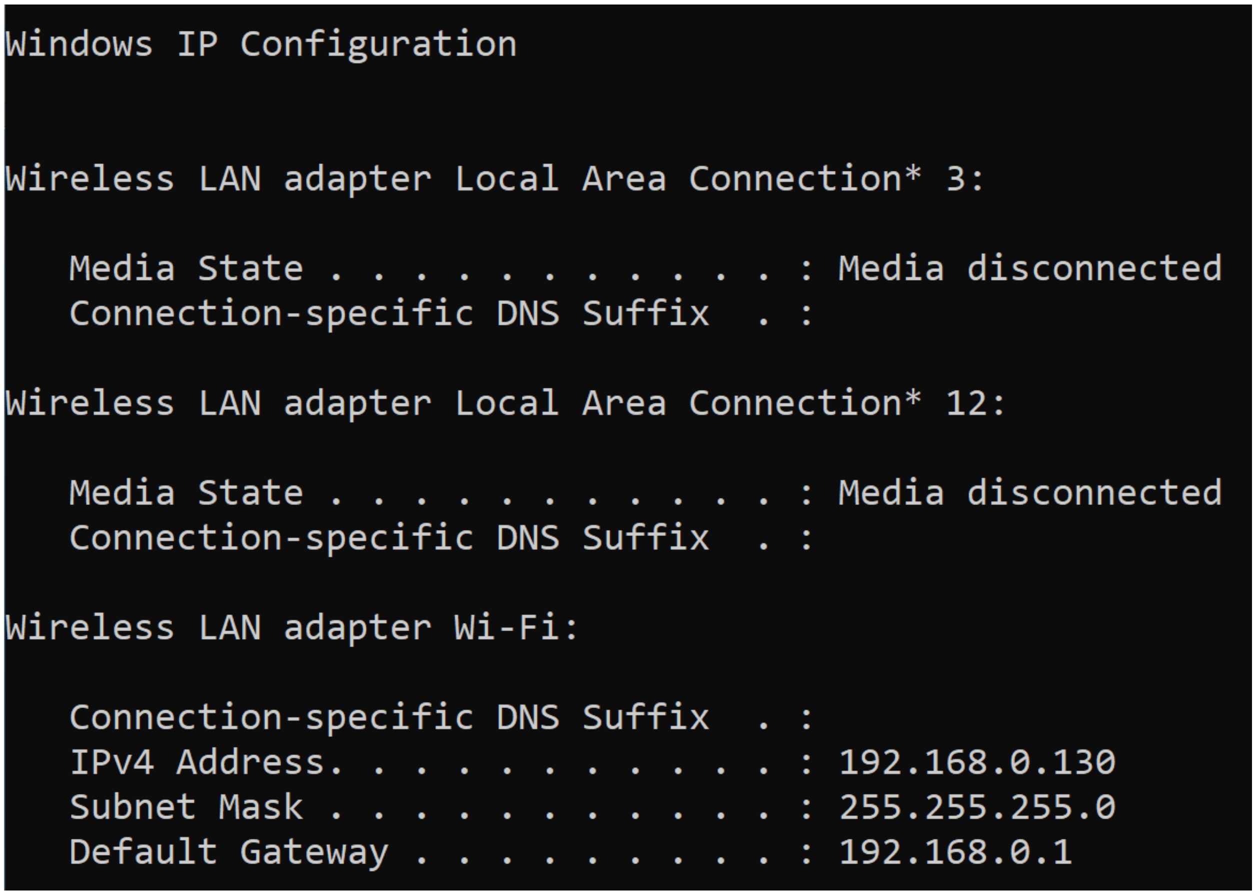
Study of Basic Commands and Network Configuration Commands and

a) Classification of IP address
b) Sub netting
c) Super netting

Basic commands and Network Configuration Commands

The below mentioned commands are some of the most useful commands required to troubleshoot network problems and configure network settings.

1. IPCONFIG



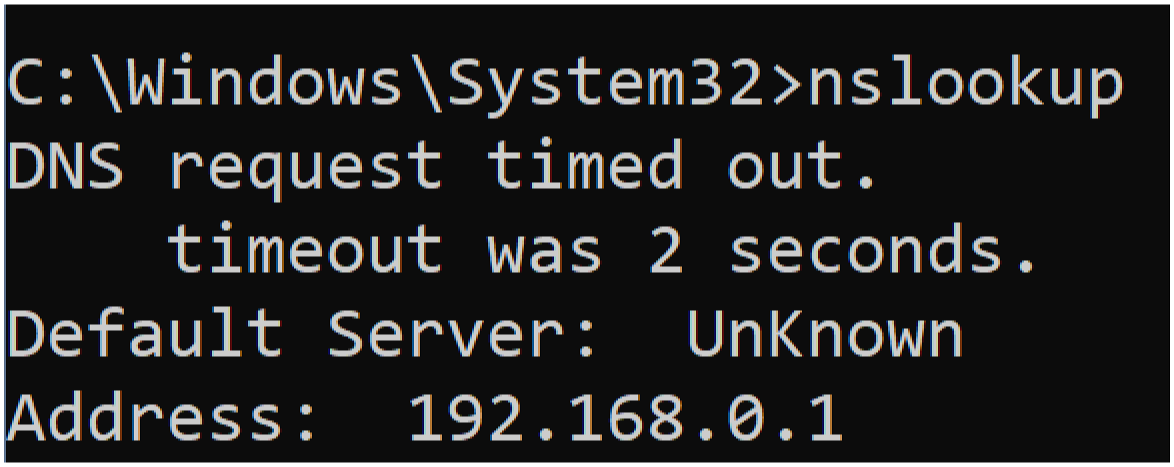
```
Windows IP Configuration


Wireless LAN adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 12:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.0.130
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.0.1
```

The IPConfig command also provides us with some variation in the primary command that targets specific system settings or data, which are:

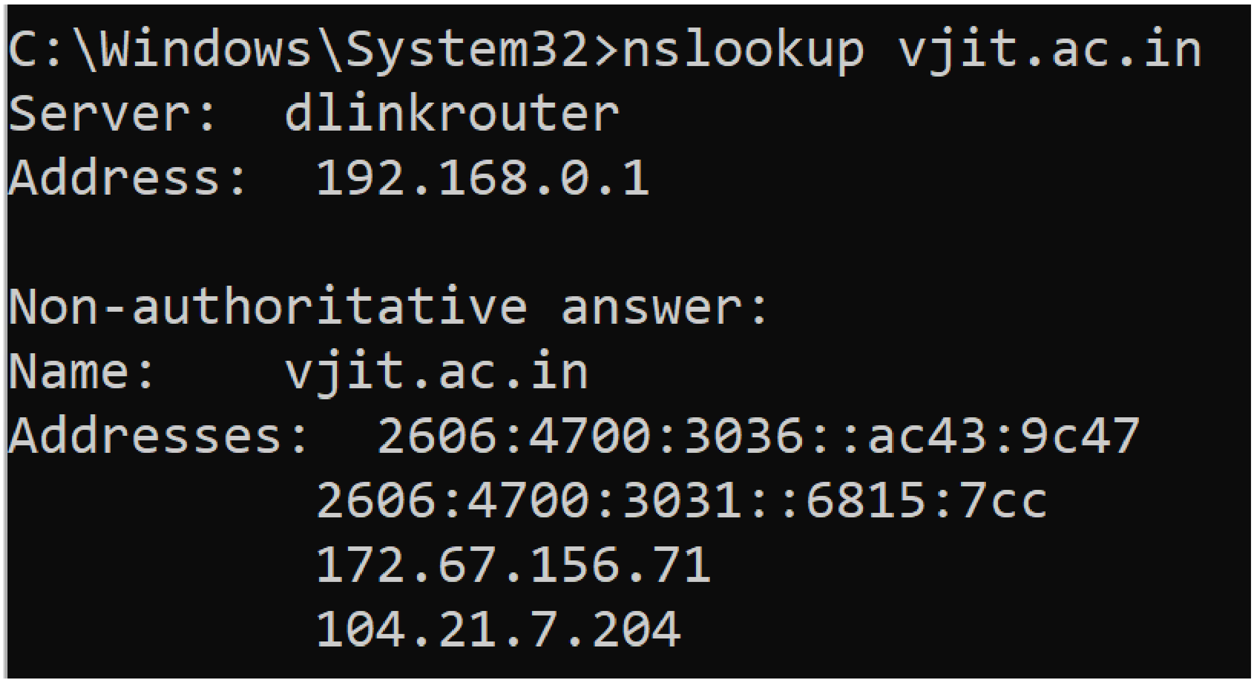
- IPConfig/all - Provides primary output with additional information about network adapters.
- IPConfig/renew - Used to renew the system’s IP address.
- IPConfig/release - Removes the system’s current IP address.

2. NSLOOKUP

The NSLOOKUP command is used to troubleshoot network connectivity issues in the system. Using the nslookup command, we can access the information related to our system's DNS server, i.e., domain name and IP address.



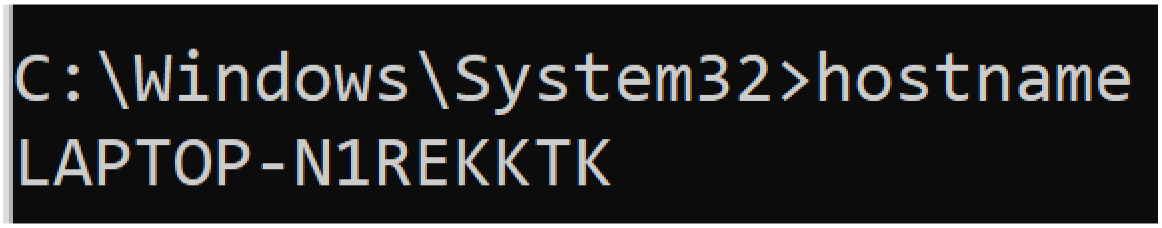
```
C:\Windows\System32>nslookup
DNS request timed out.
    timeout was 2 seconds.
Default Server:  UnKnown
Address:  192.168.0.1
```



```
C:\Windows\System32>nslookup vjit.ac.in
Server:  dlinkrouter
Address:  192.168.0.1

Non-authoritative answer:
Name:    vjit.ac.in
Addresses:  2606:4700:3036::ac43:9c47
          2606:4700:3031::6815:7cc
          172.67.156.71
          104.21.7.204
```

3. **HOSTNAME**

The **HOSTNAME** command displays the hostname of the system. The hostname command is much easier to use than going into the system settings to search for it.

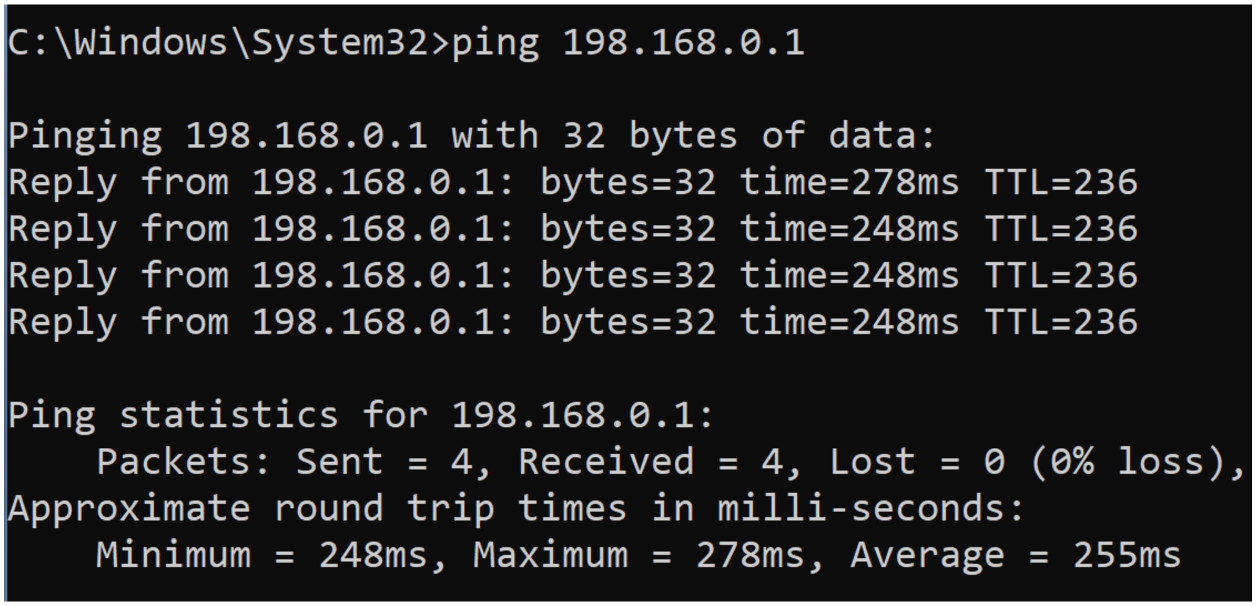


```
C:\Windows\System32>hostname
LAPTOP-N1REKKTK
```

4. PING

The Ping command is one of the most widely used commands in the prompt tool, as it allows the user to check the connectivity of our system to another host.

This command sends four experimental packets to the destination host to check whether it receives them successfully, if so, then, we can communicate with the destination host. But in case the packets have not been received, that means, no communication can be established with the destination host.



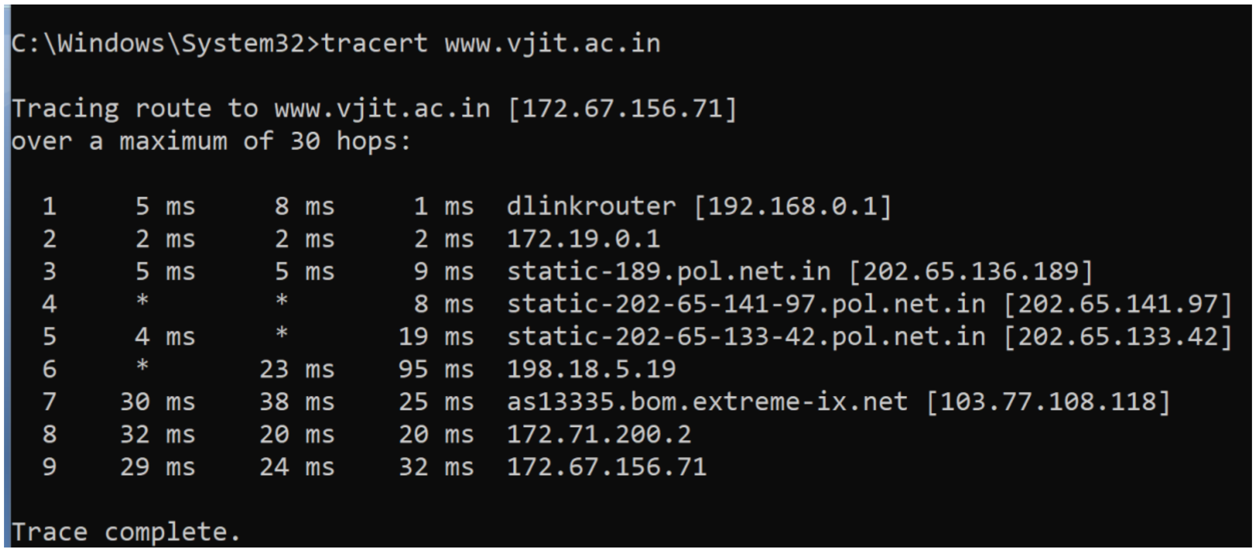
```
C:\Windows\System32>ping 198.168.0.1

Pinging 198.168.0.1 with 32 bytes of data:
Reply from 198.168.0.1: bytes=32 time=278ms TTL=236
Reply from 198.168.0.1: bytes=32 time=248ms TTL=236
Reply from 198.168.0.1: bytes=32 time=248ms TTL=236
Reply from 198.168.0.1: bytes=32 time=248ms TTL=236

Ping statistics for 198.168.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 248ms, Maximum = 278ms, Average = 255ms
```

5.TRACERT

The TRACERT command is used to trace the route during the transmission of the data packet over to the destination host and also provides us with the “hop” count during transmission.

Using the number of hops and the hop IP address, we can troubleshoot network issues and identify the point of the problem during the transmission of the data packet.

Syntax: tracert IP-address OR tracert www.destination_host_name.com



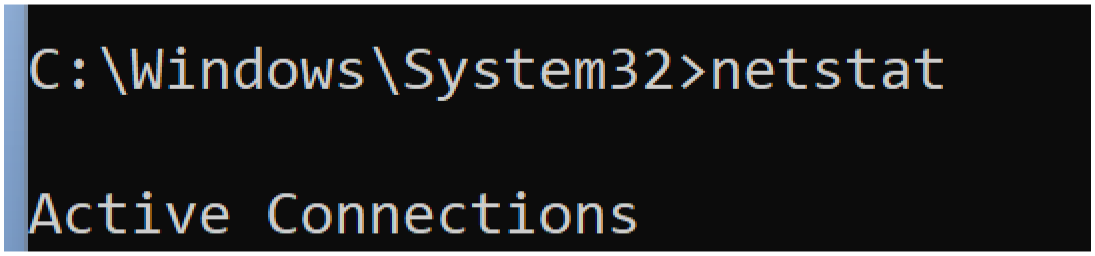
```
C:\Windows\System32>tracert www.vjit.ac.in

Tracing route to www.vjit.ac.in [172.67.156.71]
over a maximum of 30 hops:

  1     5 ms     8 ms     1 ms  dlinkrouter [192.168.0.1]
  2     2 ms     2 ms     2 ms  172.19.0.1
  3     5 ms     5 ms     9 ms  static-189.pol.net.in [202.65.136.189]
  4     *        *        8 ms  static-202-65-141-97.pol.net.in [202.65.141.97]
  5     4 ms     *       19 ms  static-202-65-133-42.pol.net.in [202.65.133.42]
  6     *       23 ms    95 ms  198.18.5.19
  7    30 ms    38 ms    25 ms  as13335.bom.extreme-ix.net [103.77.108.118]
  8    32 ms    20 ms    20 ms  172.71.200.2
  9    29 ms    24 ms    32 ms  172.67.156.71

Trace complete.
```

6. NETSTAT

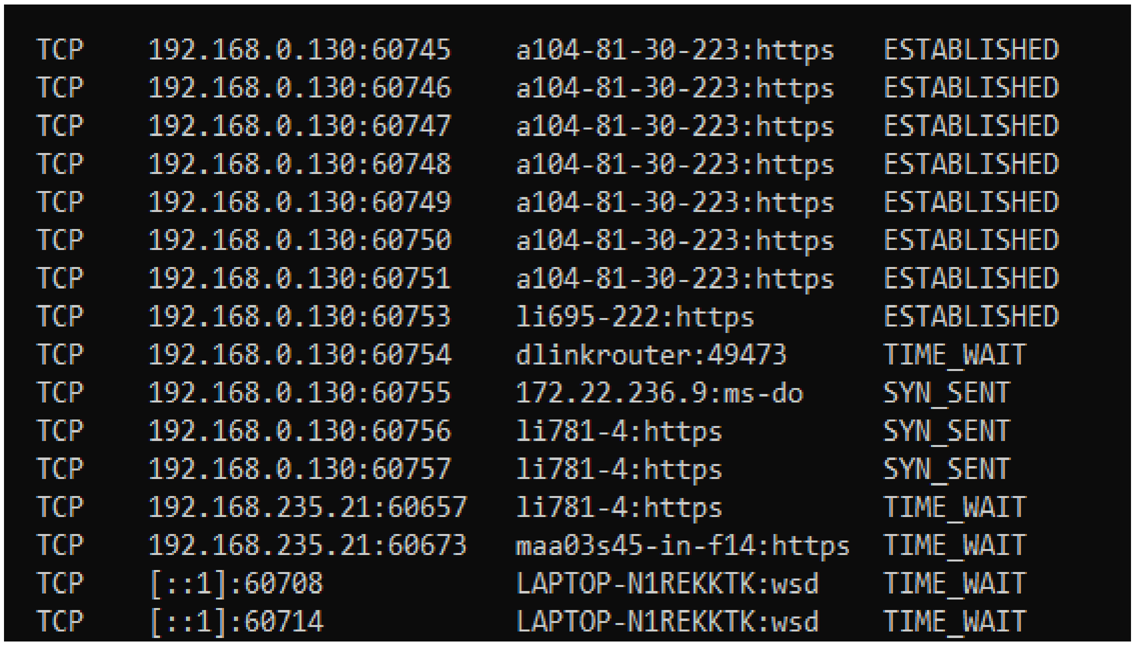
The Netstat command as the name suggests displays an overview of all the network connections in the device. The table shows detail about the connection protocol, address, and the current state of the network.

It is used for network statics, diagnostics, and analysis. If we are managing a huge college campus network, then this tool is useful because it provides an advanced aspect of the network.

Command to enter in Prompt - netstat



```
C:\Windows\System32>netstat

Active Connections
```

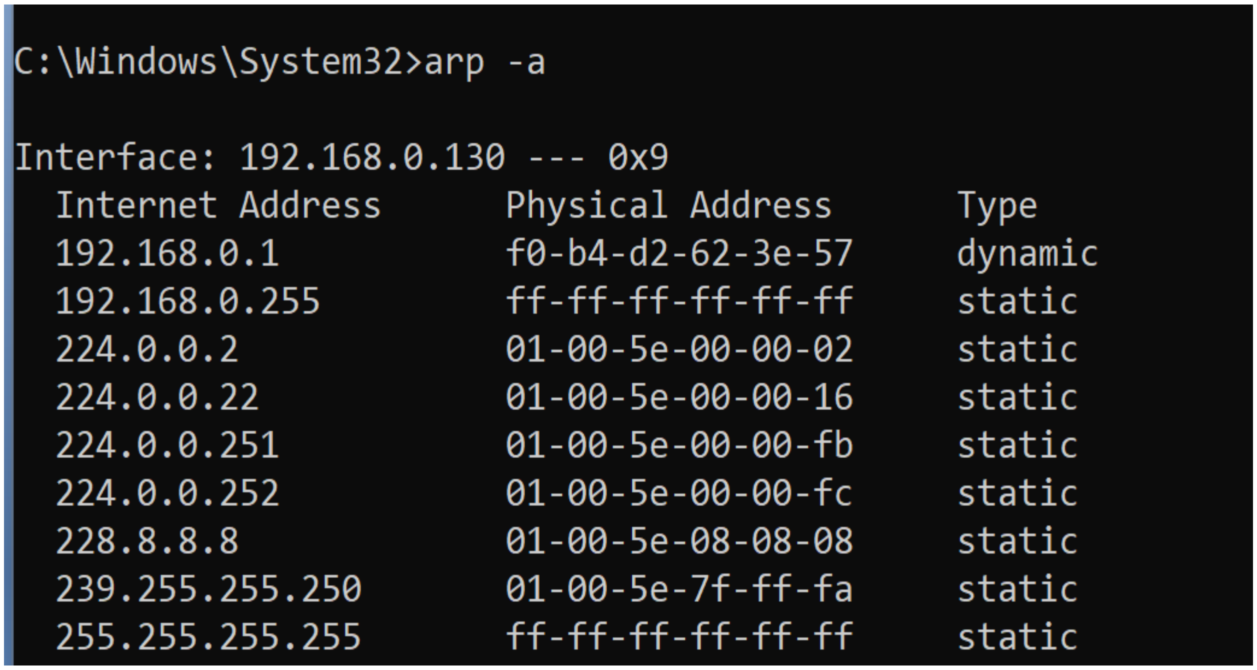


```
TCP    192.168.0.130:60745    a104-81-30-223:https   ESTABLISHED
TCP    192.168.0.130:60746    a104-81-30-223:https   ESTABLISHED
TCP    192.168.0.130:60747    a104-81-30-223:https   ESTABLISHED
TCP    192.168.0.130:60748    a104-81-30-223:https   ESTABLISHED
TCP    192.168.0.130:60749    a104-81-30-223:https   ESTABLISHED
TCP    192.168.0.130:60750    a104-81-30-223:https   ESTABLISHED
TCP    192.168.0.130:60751    a104-81-30-223:https   ESTABLISHED
TCP    192.168.0.130:60753    li695-222:https        ESTABLISHED
TCP    192.168.0.130:60754    dlinkrouter:49473      TIME_WAIT
TCP    192.168.0.130:60755    172.22.236.9:ms-do     SYN_SENT
TCP    192.168.0.130:60756    li781-4:https          SYN_SENT
TCP    192.168.0.130:60757    li781-4:https          SYN_SENT
TCP    192.168.235.21:60657   li781-4:https          TIME_WAIT
TCP    192.168.235.21:60673   maa03s45-in-f14:https  TIME_WAIT
TCP    [::1]:60708            LAPTOP-N1REKKTK:wsd    TIME_WAIT
TCP    [::1]:60714            LAPTOP-N1REKKTK:wsd    TIME_WAIT
```

7.arp (Address Resolution Protocol)

The ARP command is used to access the mapping structure of IP addresses to the MAC address. This provides us with a better understanding of the transmission of packets in the network channel.

Command to enter in Prompt – arp



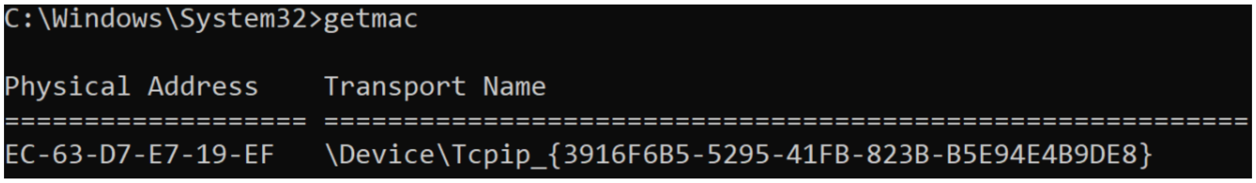
```
C:\Windows\System32>arp -a

Interface: 192.168.0.130 --- 0x9
  Internet Address      Physical Address      Type
  192.168.0.1           f0-b4-d2-62-3e-57     dynamic
  192.168.0.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  228.8.8.8             01-00-5e-08-08-08     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

8.Systeminfo

Using the SYSTEMINFO command, we can access the system's hardware and software details, such as processor data, booting data, Windows version, etc.

9.GETMAC

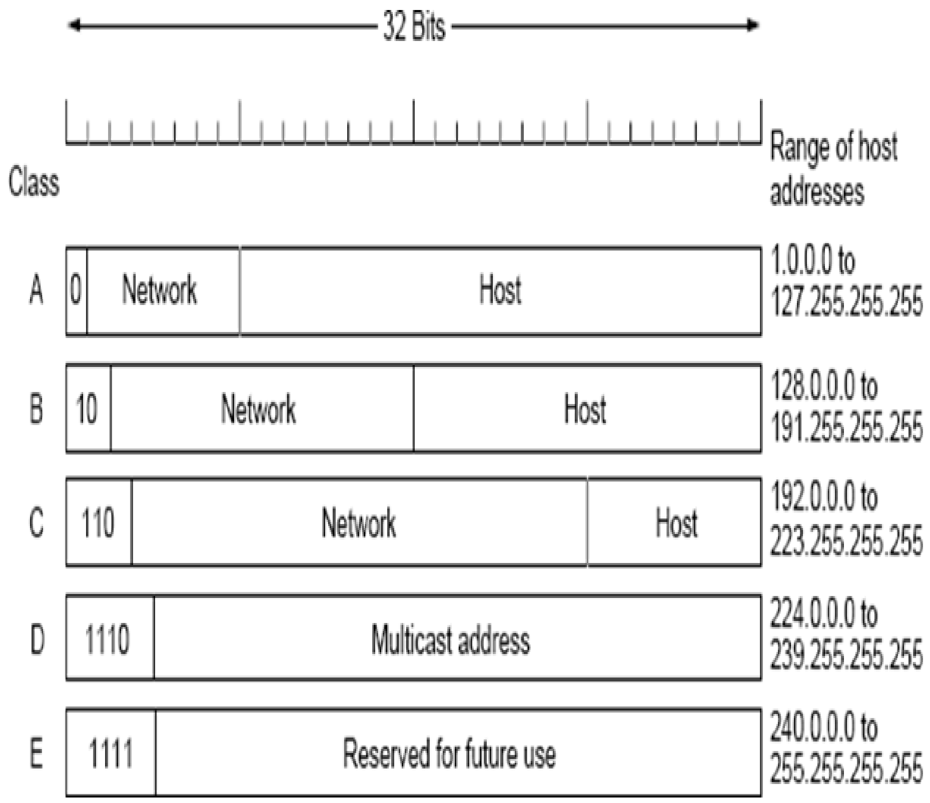
getmac (get mac address). This command returns the MAC address from all the network cards on a system. When troubleshooting a client connection



```
C:\Windows\System32>getmac

Physical Address    Transport Name
=================== ==========================================================
EC-63-D7-E7-19-EF   \Device\Tcpip_{3916F6B5-5295-41FB-823B-B5E94E4B9DE8}
```

a) Classification of IP address
   IPv4 address is 32 bits
   Divide it into a "network part" and "host part"
   - "network part" of the address identifies which network in the internetwork (e.g. the Internet)

   "host part" identifies host on that network


32 Bits
Class
Range of host addresses
A
0
Network
Host
1.0.0.0 to 127.255.255.255
B
10
Network
Host
128.0.0.0 to 191.255.255.255
C
110
Network
Host
192.0.0.0 to 223.255.255.255
D
1110
Multicast address
224.0.0.0 to 239.255.255.255
E
1111
Reserved for future use
240.0.0.0 to 255.255.255.255

Configure IP address to Router and PC in Packet Tracer

- As a Network Engineer, it is compulsory to know how to configure an IP Address on networking devices like Router, Switch, PC, and Server.
- Assigning an IP address to a device is a foundational requirement for all Cisco networking devices.
- Networking devices communicate with each other with the help of IP address that configures on an individual device.

Refer the link https://www.tutorialandexample.com/configure-ip-address-to-router-and-pc-in-packet-tracer

i) Configure an IP Address to PC

There are following steps involved to configure an IP Address to PC:

Step1: Open the Cisco Packet Tracer.

Step2: Drag and drop PC from the bottom of the interface into the middle of the working area.
Step3: Click on PC ->Config Gateway like 10.0.0.1

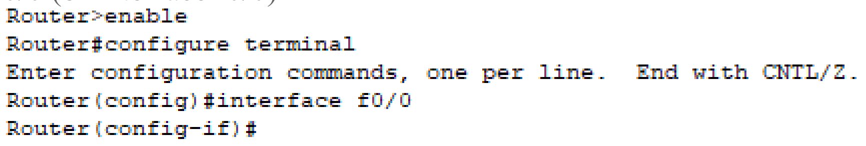
ii) Configure an IP Address on Router's Interface

There are two ways to configure an IP Address on the Router's Interface-

1. Configure an IP Address to Fast Ethernet Interface
2. Configure an IP Address to Serial Interface

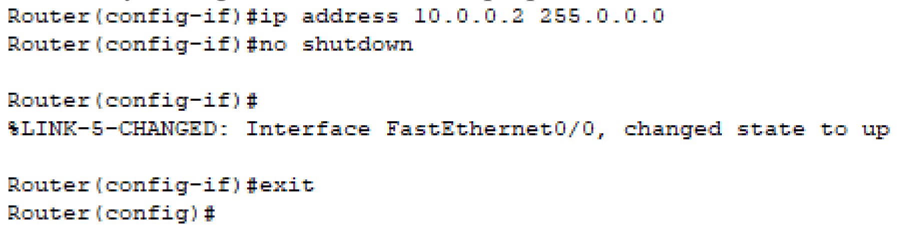
1. Configure an IP Address to Fast Ethernet Interface

There are following steps involved to configure an IP Address to Fast Ethernet Interface –
Step1: Open the Cisco Packet Tracer.
Step2: Drag and drop any series of the router from the bottom of the interface into the middle of the working area.
Step 3: Select cable from the bottom of the interface to connect the routers.
Step 4: Click on Router R0, then on CLI.
Step5: Go to the global configuration mode, and type **slot/ port** or interface Fast Ethernet 0/0 (or interface f0/0)



```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface f0/0
Router(config-if)#
```

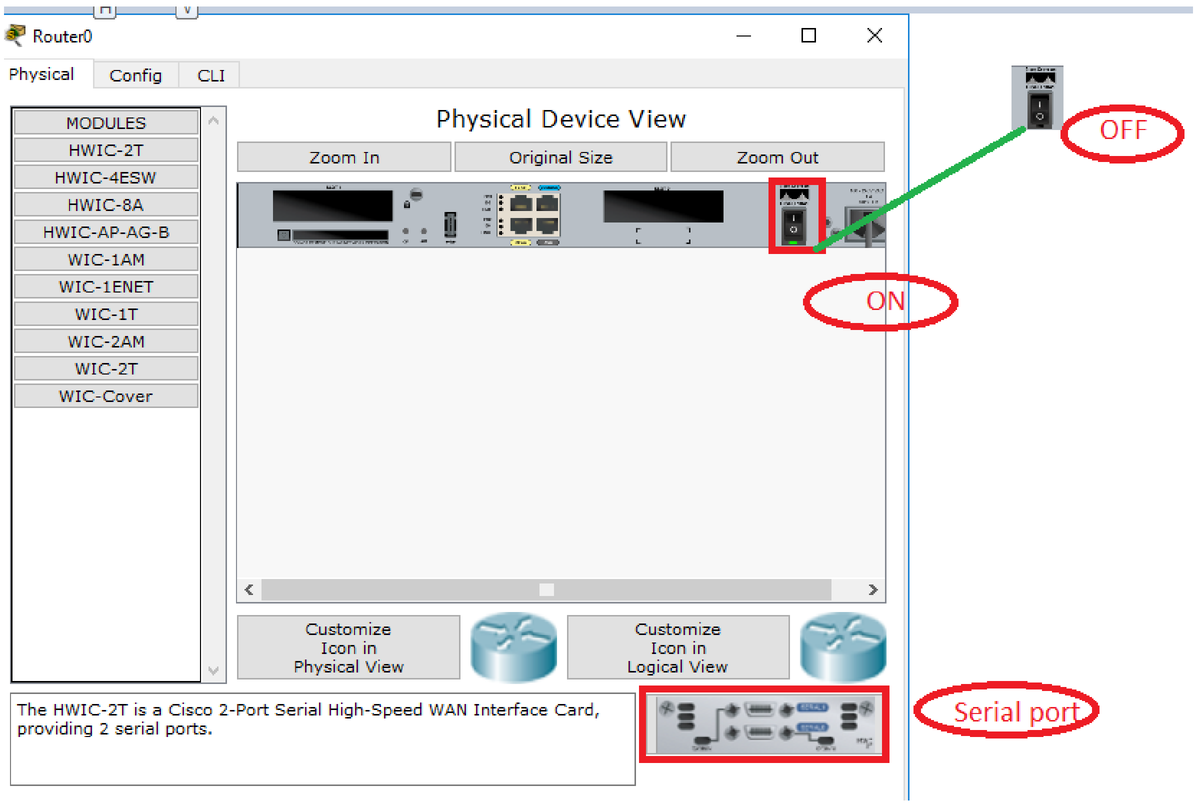
Step 6: Now configure an IP address and subnet mask then give "**no shutdown"** command. (In our case, IP address is 10.0.0.2 and subnet mask is 255.0.0.0)
Note: Carefully configure IP address with proper interfaces.



```
Router(config-if)#ip address 10.0.0.2 255.0.0.0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

Router(config-if)#exit
Router(config)#
```

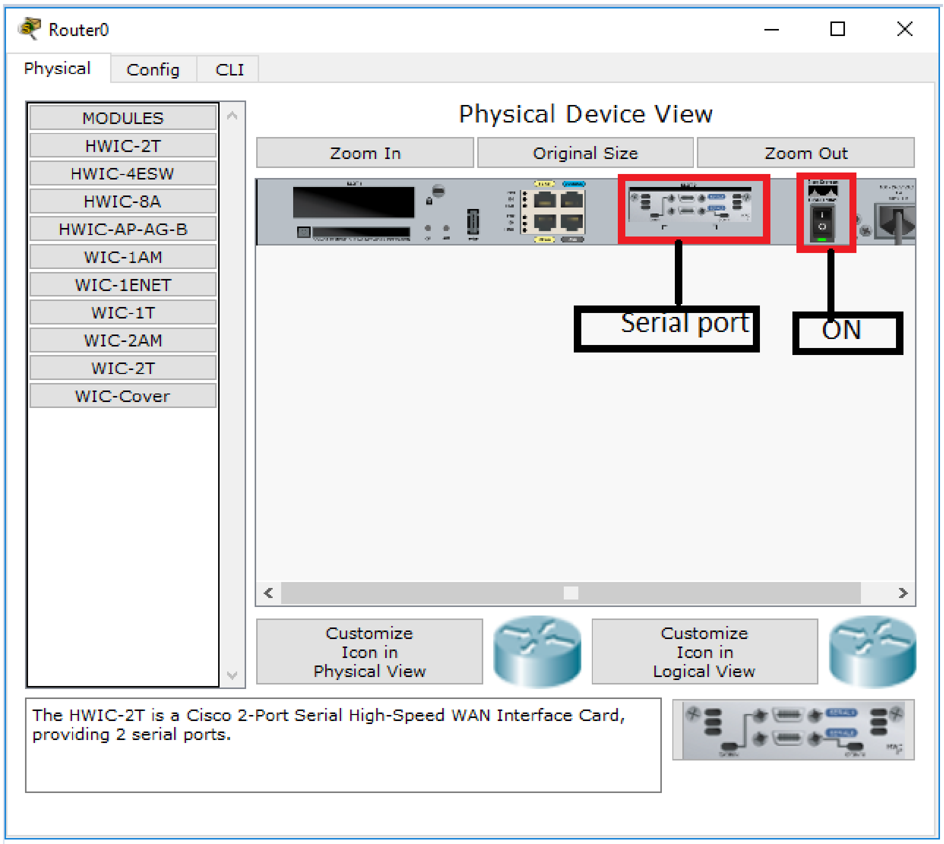
2) Configure an IP Address to Serial Interface

There are following steps involved to configure an IP Address to Serial Interface –
Step1: Open the Cisco packet tracer.
Step2: Drag and drop any series of the router from the bottom of the interface into the middle of the working area.
Step 3: Click the Router ->select then Physical->select then **WIC-IT->**switch off the router.



The HWIC-2T is a Cisco 2-Port Serial High-Speed WAN Interface Card, providing 2 serial ports.

- Add serial port -> Switch ON the router.



The HWIC-2T is a Cisco 2-Port Serial High-Speed WAN Interface Card, providing 2 serial ports.

- Now Go to CLI, then the global configuration mode, and type interface serial 0 0/0 (or interface s0/0/0).

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface s0/0/0
Router(config-if)#
```

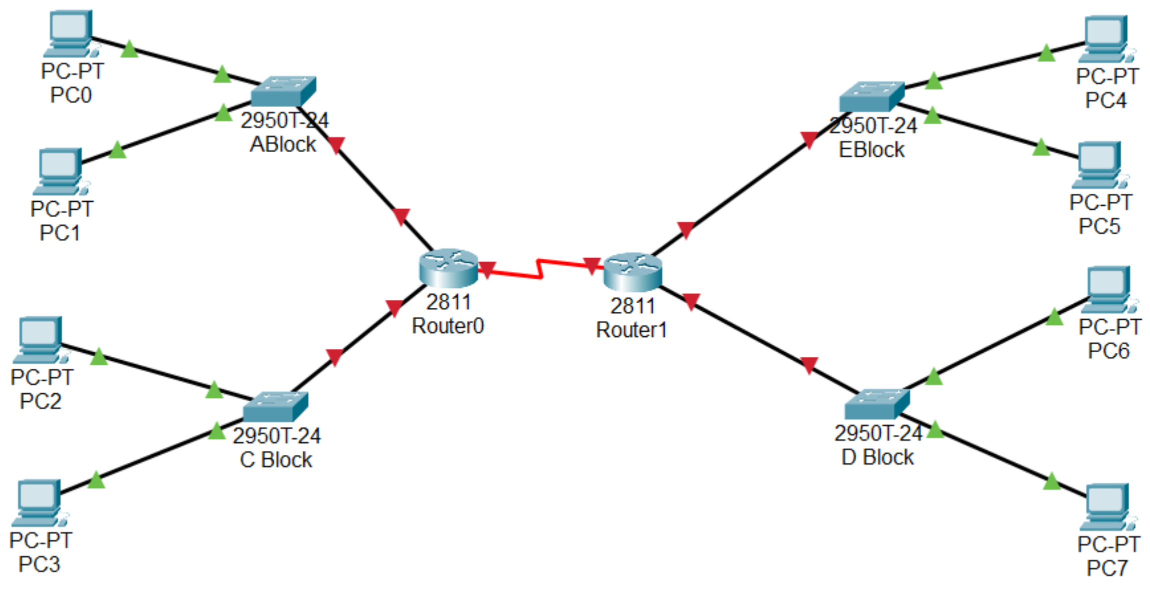
- Now, configure an IP address and subnet mask, and then give **no shutdown** command. (In our case IP address is 10.0.0.2 and subnet mask is 255.0.0.0)

```
Router(config-if)#ip address 10.0.0.2 255.0.0.0
Router(config-if)#clock rate 64000
Router(config-if)#no sh
Router(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial0/0/0, changed state to down
Router(config-if)#
Router(config-if)#
Router(config-if)#exit
Router(config)#
```

Note: Carefully configure IP address with proper interfaces.

b) Subnetting:

Subnetting is the procedure to divide the network into sub-networks or small networks, these smaller networks are known as subnets
Click the video and follow the steps https://www.youtube.com/watch?v=7RG52YSFBrM



c) Supernetting:
Supernetting is the procedure to combine small networks into larger spaces. In subnetting,
Or It is **the process of combining multiple subnetworks into one network**. It's also known as aggregation or route summarization
https://www.youtube.com/watch?v=PYmnFxErtm8

Week 6: Connect the computers in Local Area Network and Observing Static and Dynamic Routing using Packet Tracer

Static routing is a routing protocol that helps to keep your network organized and to optimize routing performance. It enables the router to assign a specific path to each network segment and to keep track of network changes. This helps to improve network stability and continuity. This adds security because a single administrator can only authorize routing to particular networks.

Steps to Configure and Verify Two Router Connections in Cisco Packet Tracer :

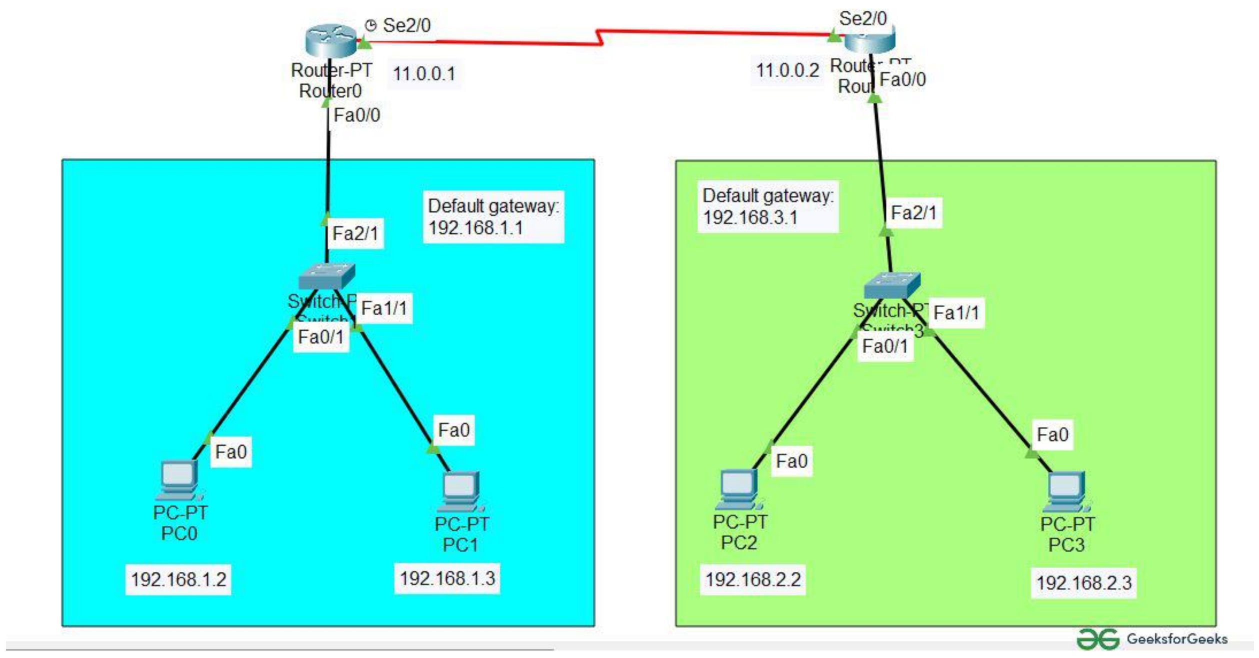
Step 1: First, open the cisco packet tracer desktop and select the devices given below:

S.NO	Device	Model Name	Qty.
1.	PC	PC	4
2.	Switch	PT-Switch	2
3.	Router	PT-Router	2

IP Addressing Table For PCs:

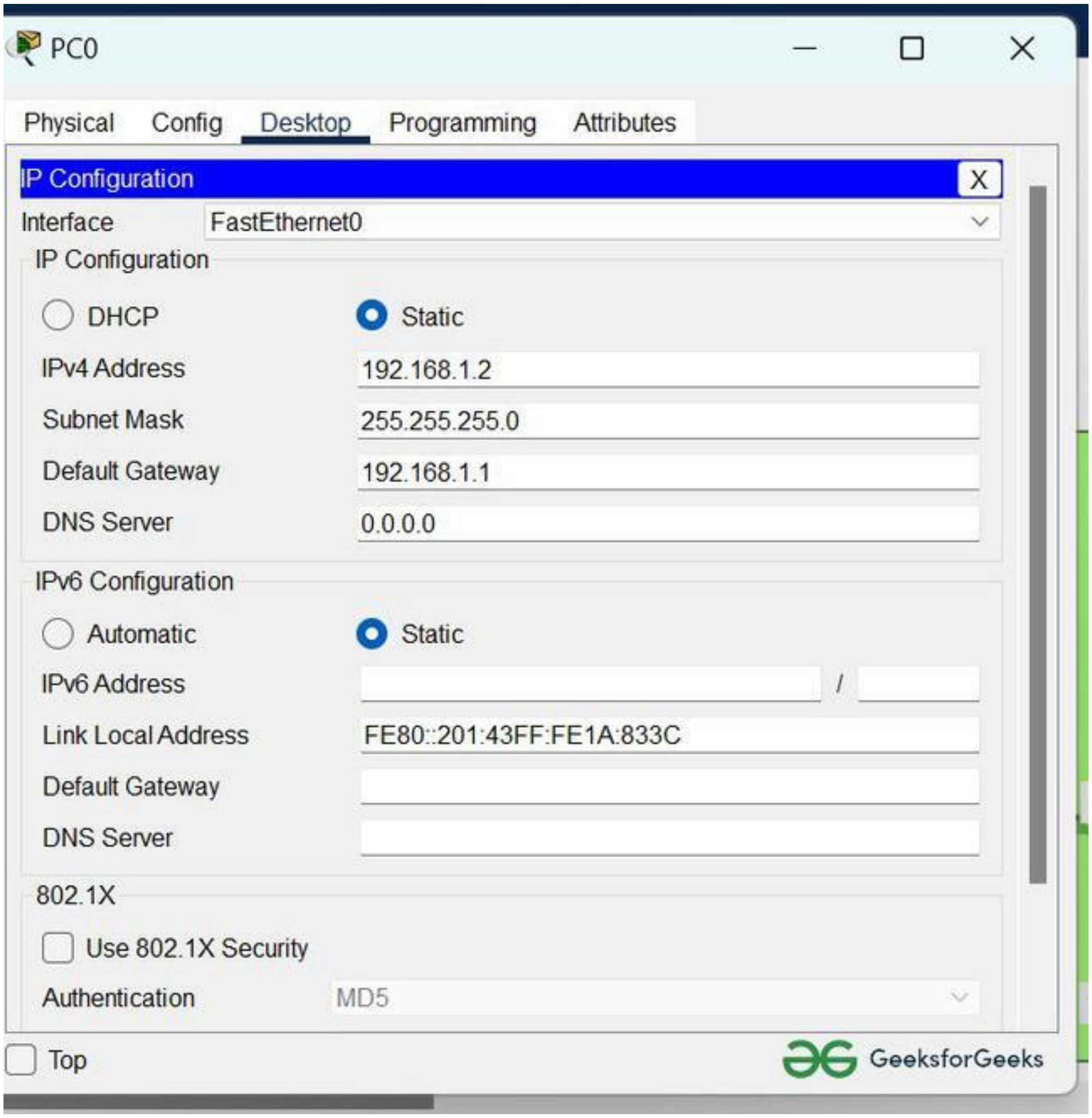
NO	Device	IPv4 Address	Subnet Mask	Default Gateway
1.	pc0	192.168.1.2	255.255.255.0	192.168.1.1
2.	pc1	192.168.1.3	255.255.255.0	192.168.1.1
3.	pc2	192.168.2.2	255.255.255.0	192.168.2.1
4.	pc3	192.168.2.3	255.255.255.0	192.168.2.1

- hen, create a network topology as shown below the image.
- Use an Automatic connecting cable to connect the devices with others.



Step 2: Configure the PCs (hosts) with IPv4 address and Subnet Mask according to the IP addressing table given above.

- To assign an IP address in PC0, click on PC0.
- Then, go to desktop and then IP configuration and there you will IPv4 configuration.
- Fill IPv4 address and subnet mask.



PC0

Physical Config Desktop Programming Attributes

IP Configuration

Interface: FastEthernet0

IP Configuration

DHCP [✓ Static]

IPv4 Address: 192.168.1.2

Subnet Mask: 255.255.255.0

Default Gateway: 192.168.1.1

DNS Server: 0.0.0.0

IPv6 Configuration

Automatic [✓ Static]

IPv6 Address: /

Link Local Address: FE80::201:43FF:FE1A:833C

Default Gateway:

DNS Server:

802.1X

Use 802.1X Security

Authentication: MD5

Top

GeeksforGeeks

Step 3: Assigning IP address using the ipconfig command.

- We can also assign an IP address with the help of a command.
- Go to the command terminal of the PC.
- Then, type ipconfig <IPv4 address><subnet mask><default gateway>(if needed)

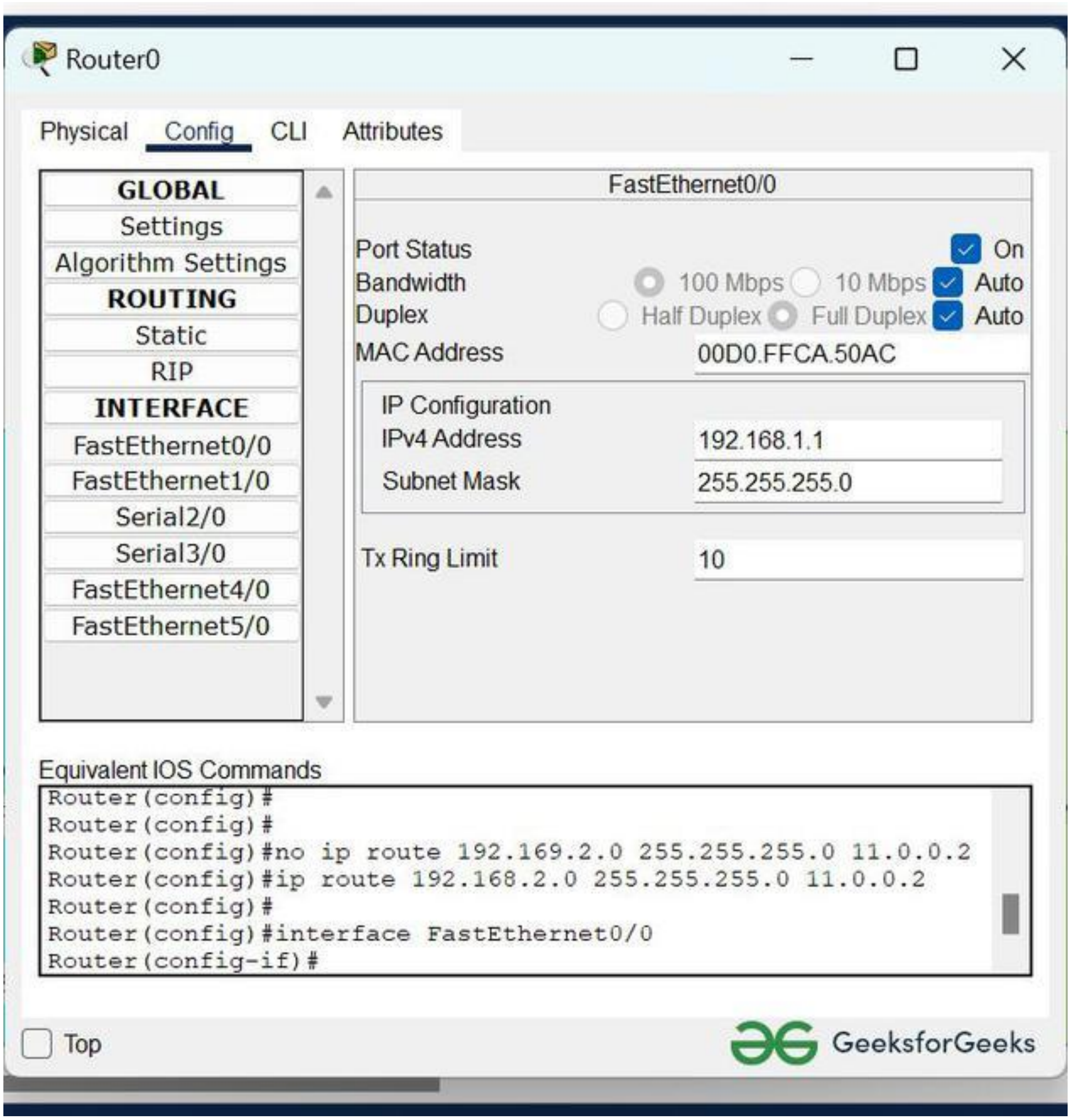
Example: ipconfig 192.168.1.3 255.255.255.0 192.168.1.1

- Repeat the same procedure with other PCs to configure them thoroughly.

Step 4: Configure router with IP address and subnet mask.

S.NO	Device	Interface	IPv4 Addressing	Subnet Mask
1.	router0	FastEthernet0/0	192.168.1.1	255.255.255.0
		Serial2/0	11.0.0.1	255.255.255.0
2.	router1	FastEthernet0/0	192.168.2.1	255.255.255.0
		Serial2/0	11.0.0.2	255.255.255.0

- To assign an IP address in router0, click on router0.
- Then, go to config and then Interfaces.
- Then, configure the IP address in FastEthernet and serial ports according to IP addressing Table.
- Fill IPv4 address and subnet mask.



FastEthernet0/0

Port Status	[✓ On]
Bandwidth	100 Mbps 10 Mbps [✓ Auto]
Duplex	Half Duplex Full Duplex [✓ Auto]
MAC Address	00D0.FFCA.50AC
IP Configuration	
IPv4 Address	192.168.1.1
Subnet Mask	255.255.255.0
Tx Ring Limit	10

Equivalent IOS Commands

```
Router(config)#
Router(config)#
Router(config)#no ip route 192.169.2.0 255.255.255.0 11.0.0.2
Router(config)#ip route 192.168.2.0 255.255.255.0 11.0.0.2
Router(config)#
Router(config)#interface FastEthernet0/0
Router(config-if)#
```

- Repeat the same procedure with other routers to configure them thoroughly.

Step 5: After configuring all of the devices we need to assign the routes to the routers.
To assign static routes to the particular router:

- First, click on router0 then Go to CLI.
- Then type the commands and IP information given below.

CLI command : ip route <network id> <subnet mask><next hop>

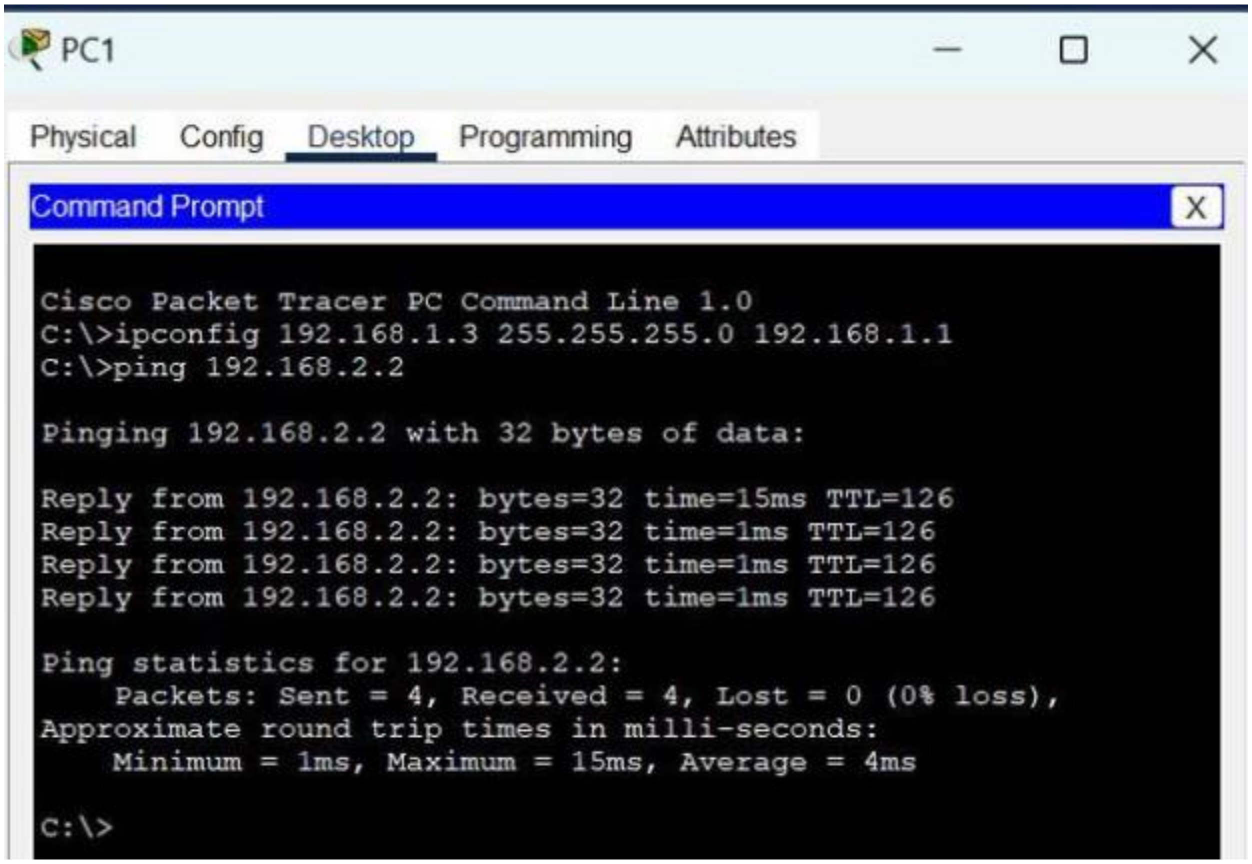
Router(config)#ip route 192.168.2.0 255.255.255.0 11.0.0.2

Router(config)#ip route 192.168.1.0 255.255.255.0 11.0.0.1

Step 6: Verifying the network by pinging the IP address of any PC. We will use the ping command to do so.

- First, click on PC1 then Go to the command prompt
- Then type ping <IP address of targeted node>
- As we can see in the below image we are getting replies which means the connection is working very fine

Example : ping 192.168.2.2



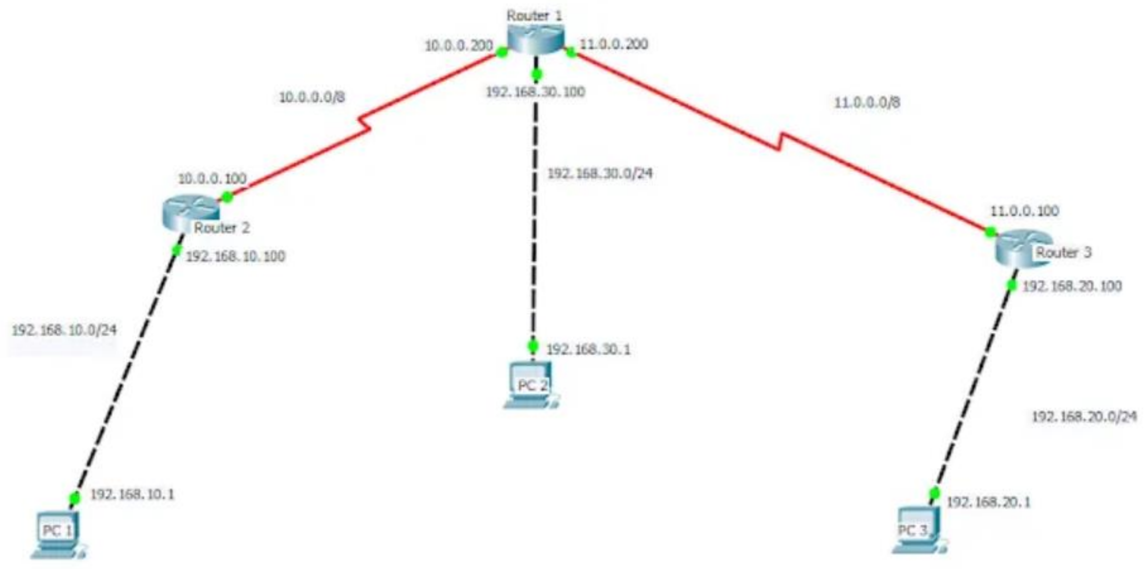
```
Cisco Packet Tracer PC Command Line 1.0
C:\>ipconfig 192.168.1.3 255.255.255.0 192.168.1.1
C:\>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Reply from 192.168.2.2: bytes=32 time=15ms TTL=126
Reply from 192.168.2.2: bytes=32 time=1ms TTL=126
Reply from 192.168.2.2: bytes=32 time=1ms TTL=126
Reply from 192.168.2.2: bytes=32 time=1ms TTL=126

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 15ms, Average = 4ms

C:\>
```

Dynamic Routing :

Dynamic routing is all about configuring a network using dynamic routing protocols.

Dynamic Routing Protocol is divided into two main parts.
1.Interior Gateway Protocol
2.Exterior Gateway Protocol

-this is an autonomous system and handled by only one admin.
-this protocol is also divided into two parts,
1. Distant Vector Protocols(Bellman-Ford Algorithm) — distance is measured by `hop count` and use for simple networks
2. Link State Protocol(Dijkstra Algorithm) — this uses some other information like neighbor router info and this is best for complex network designs



In this Network diagram,
192.168.10.0 , 192.168.30.0 , 192.168.20.0 , 10.0.0.0 , 11.0.0.0 are 5 different networks.
Router 1, Router 2, Router 3 are routers in this network.
PC 1, PC 2, PC 3 are computers(end devices) in this network.
Black dashed lines are Copper Cross-Over cables which use to connect different types of devices.
Red color lines are Serial DCE cables which are building the connection between two routers.

PC 1 — IP Address: 192.168.10.1(Fast Ethernet 0/0)
Default Gateway : 192.168.10.100(Fast Ethernet 0/0)

PC 2 — IP Address: 192.168.30.1(Fast Ethernet 0/0)
Default Gateway : 192.168.30.100(Fast Ethernet 0/0)

PC 3 — IP Address: 192.168.20.1(Fast Ethernet 0/0)
Default Gateway : 192.168.20.100(Fast Ethernet 0/0)

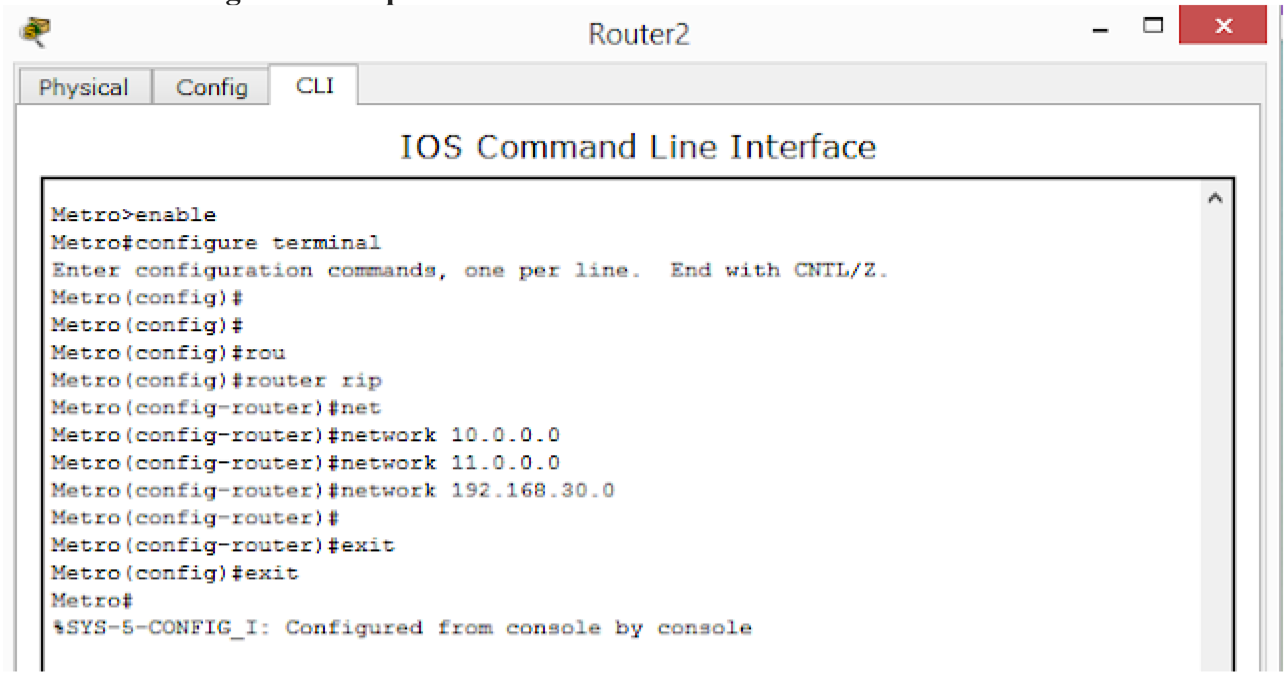
Router 1 — IP Address(SERIAL 0/2/0) : 10.0.0.200
IP Address(SERIAL 0/2/1) : 11.0.0.200

Router 2 — IP Address(SERIAL 0/0/0) : 10.0.0.100
IP Address(Fast Ethernet 0/0) : 192.168.10.100

Router 3 — IP Address(SERIAL 0/0/0) : 11.0.0.100
IP Address(Fast Ethernet 0/0) : 192.168.20.100

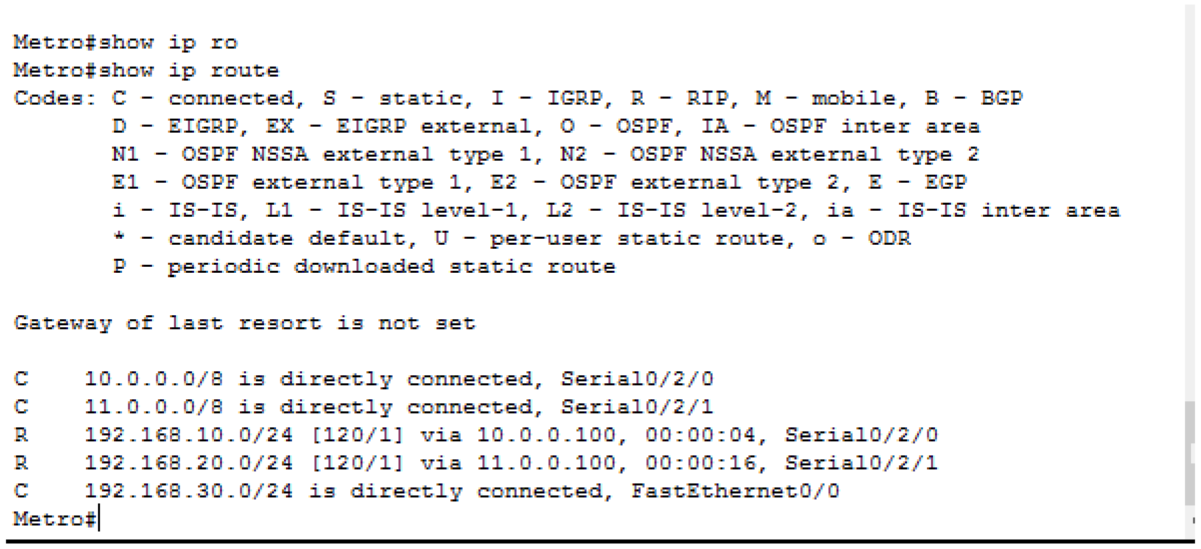
After configuring these IP addresses to the corresponding devices only, we should configure dynamic routing to the network.
When doing dynamic routing using RIP protocol first we should identify the networks which are connected to each router and for those routers we have to connect those networks through RIP.
So below mentioned procedure will guide you to do the RIP dynamic routing.

1. Router 1 configuration steps.



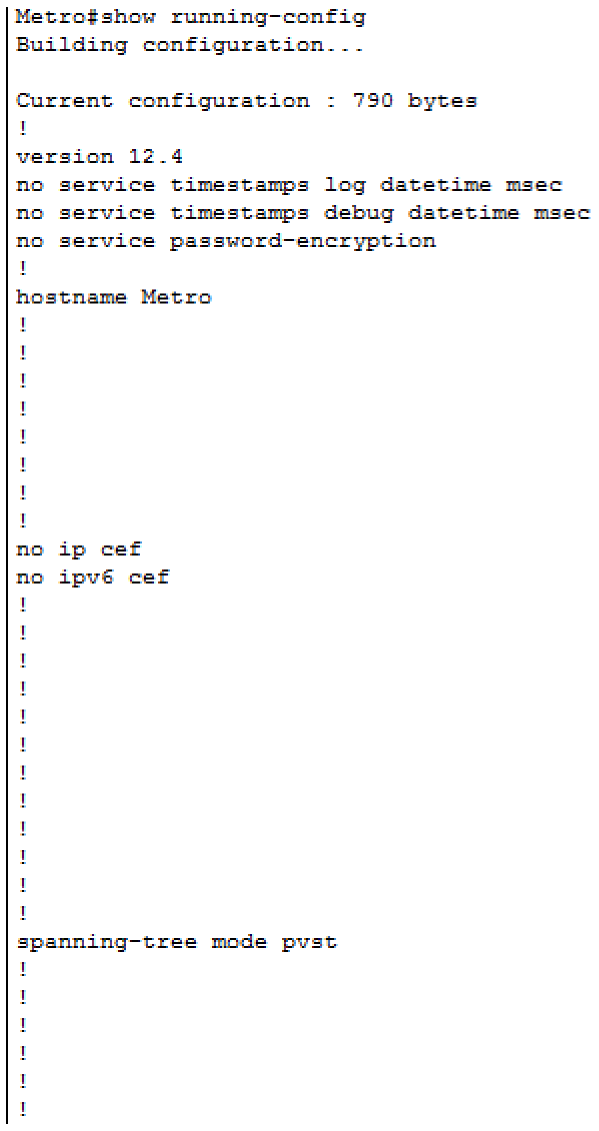
```
Metro>enable
Metro#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Metro(config)#
Metro(config)#
Metro(config)#rou
Metro(config)#router rip
Metro(config-router)#net
Metro(config-router)#network 10.0.0.0
Metro(config-router)#network 11.0.0.0
Metro(config-router)#network 192.168.30.0
Metro(config-router)#
Metro(config-router)#exit
Metro(config)#exit
Metro#
%SYS-5-CONFIG_I: Configured from console by console
```

The below figure shows us the routing table which is updating periodically.
C- directly connected networks are marked as C.
R- networks which connected using the RIP dynamic routing

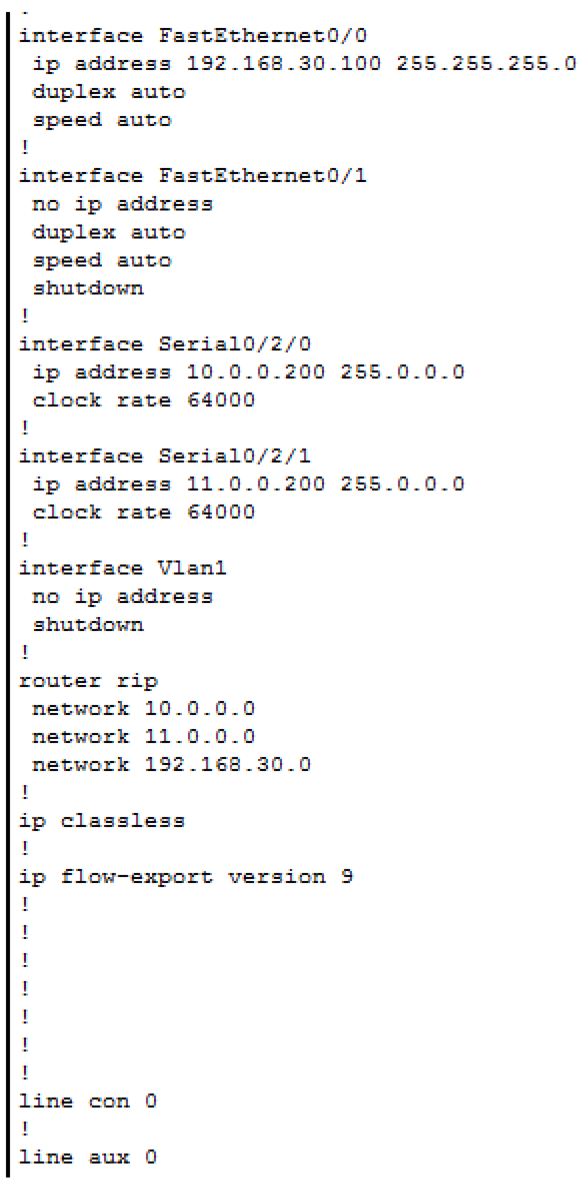


```
Metro#show ip ro
Metro#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, Serial0/2/0
C    11.0.0.0/8 is directly connected, Serial0/2/1
R    192.168.10.0/24 [120/1] via 10.0.0.100, 00:00:04, Serial0/2/0
R    192.168.20.0/24 [120/1] via 11.0.0.100, 00:00:16, Serial0/2/1
C    192.168.30.0/24 is directly connected, FastEthernet0/0
Metro#
```

Below two figures are showing us the **current running configurations** of the router 1.

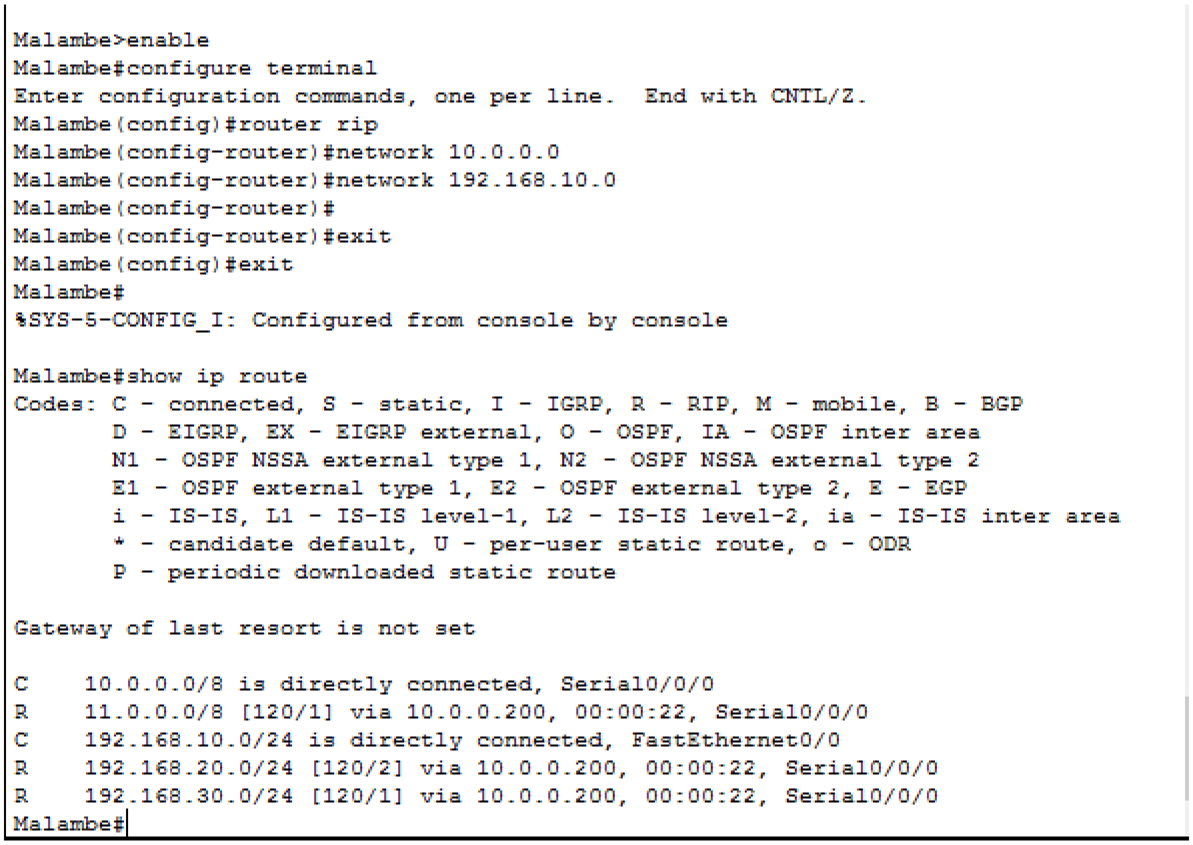


```
Metro#show running-config
Building configuration...

Current configuration : 790 bytes
!
version 12.4
no service timestamps log datetime msec
no service timestamps debug datetime msec
no service password-encryption
!
hostname Metro
!
!
!
!
!
!
!
!
no ip cef
no ipv6 cef
!
!
!
!
!
!
!
!
!
!
!
!
spanning-tree mode pvst
!
!
!
!
!
!
```



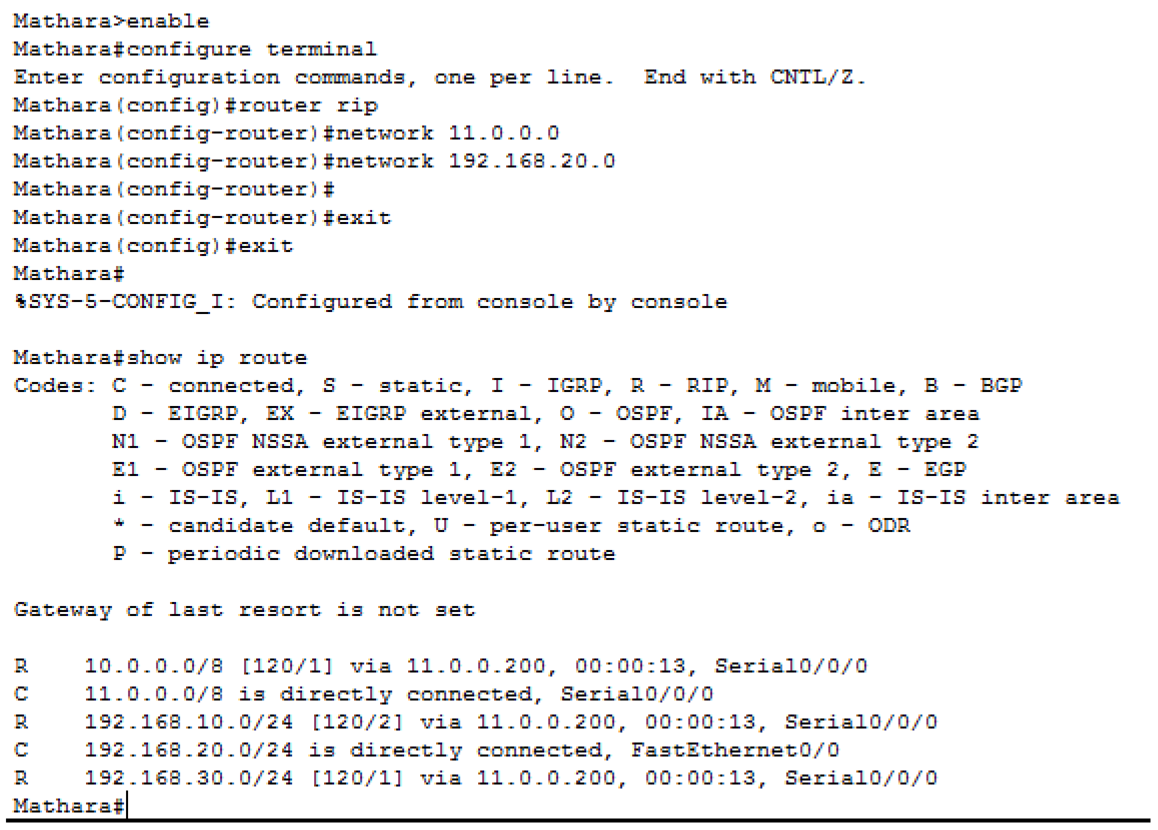
```
interface FastEthernet0/0
 ip address 192.168.30.100 255.255.255.0
 duplex auto
 speed auto
!
interface FastEthernet0/1
 no ip address
 duplex auto
 speed auto
 shutdown
!
interface Serial0/2/0
 ip address 10.0.0.200 255.0.0.0
 clock rate 64000
!
interface Serial0/2/1
 ip address 11.0.0.200 255.0.0.0
 clock rate 64000
!
interface Vlan1
 no ip address
 shutdown
!
router rip
 network 10.0.0.0
 network 11.0.0.0
 network 192.168.30.0
!
ip classless
!
ip flow-export version 9
!
!
!
!
!
!
!
line con 0
!
line aux 0
```

2. Router 2 configuration steps.



```
Malambe>enable
Malambe#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Malambe(config)#router rip
Malambe(config-router)#network 10.0.0.0
Malambe(config-router)#network 192.168.10.0
Malambe(config-router)#
Malambe(config-router)#exit
Malambe(config)#exit
Malambe#
%SYS-5-CONFIG_I: Configured from console by console

Malambe#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, Serial0/0/0
R    11.0.0.0/8 [120/1] via 10.0.0.200, 00:00:22, Serial0/0/0
C    192.168.10.0/24 is directly connected, FastEthernet0/0
R    192.168.20.0/24 [120/2] via 10.0.0.200, 00:00:22, Serial0/0/0
R    192.168.30.0/24 [120/1] via 10.0.0.200, 00:00:22, Serial0/0/0
Malambe#
```

3. Router 3 configuration steps.



```
Mathara>enable
Mathara#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Mathara(config)#router rip
Mathara(config-router)#network 11.0.0.0
Mathara(config-router)#network 192.168.20.0
Mathara(config-router)#
Mathara(config-router)#exit
Mathara(config)#exit
Mathara#
%SYS-5-CONFIG_I: Configured from console by console

Mathara#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

R    10.0.0.0/8 [120/1] via 11.0.0.200, 00:00:13, Serial0/0/0
C    11.0.0.0/8 is directly connected, Serial0/0/0
R    192.168.10.0/24 [120/2] via 11.0.0.200, 00:00:13, Serial0/0/0
C    192.168.20.0/24 is directly connected, FastEthernet0/0
R    192.168.30.0/24 [120/1] via 11.0.0.200, 00:00:13, Serial0/0/0
Mathara#
```

After Configuring three routers as above mentioned, we can now send packets through one end to another. To check whether it is working we can check by ping command.

PING command can check the connection between two end devices and have to execute in the terminal of the source device.

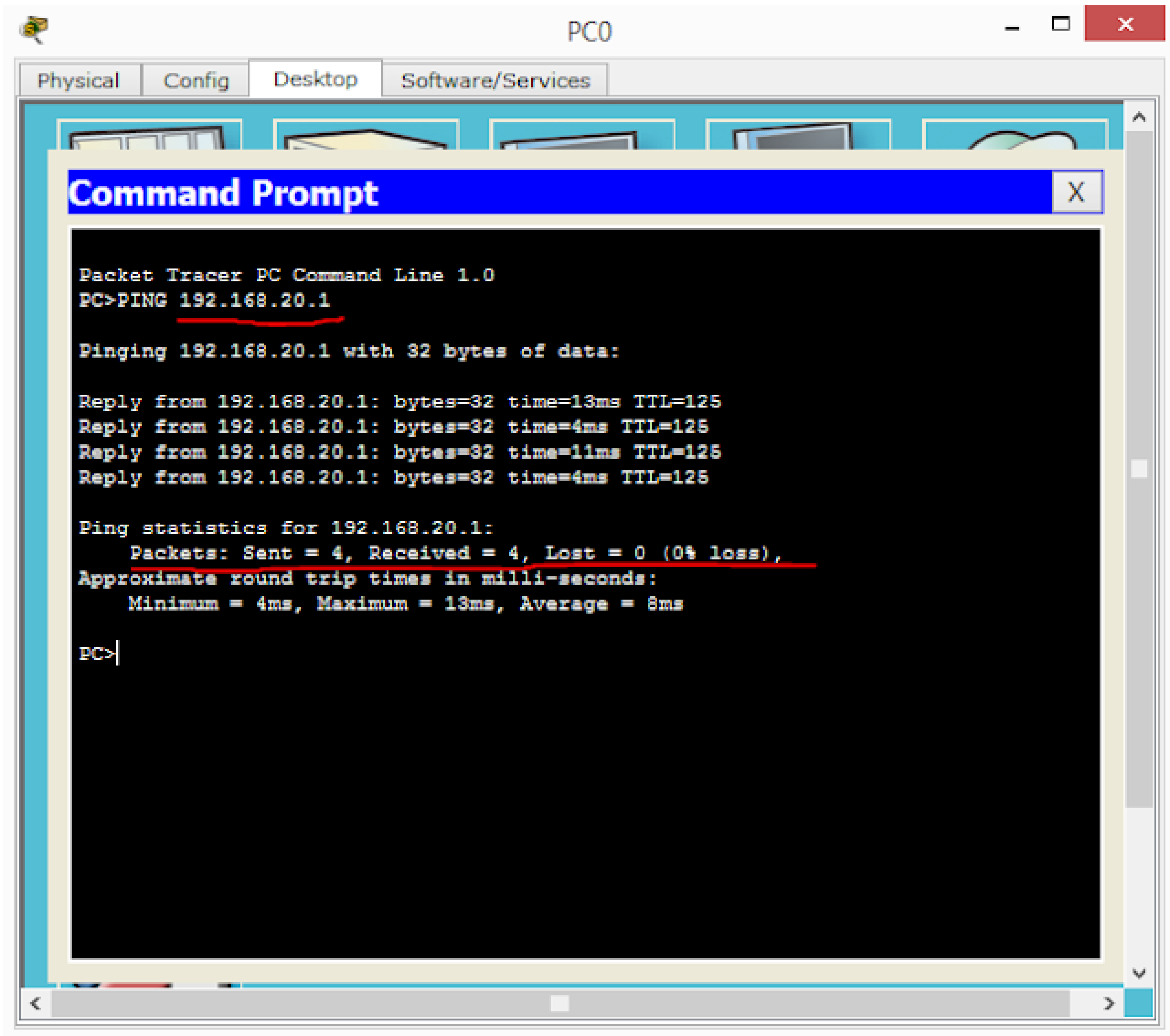
PING <IP address of the destination>

Now we will just get an example to demonstrate this scenario. So for that, we will take **PC 1** as the SOURCE DEVICE and **PC 3** as the DESTINATION DEVICE.

PC 1 IP- 192.168.10.1

PC 3 IP- 192.168.20.1

Below figure we can see that the data packets transfer and the data packets which have lost their path.



```
Packet Tracer PC Command Line 1.0
PC>PING 192.168.20.1

Pinging 192.168.20.1 with 32 bytes of data:

Reply from 192.168.20.1: bytes=32 time=13ms TTL=125
Reply from 192.168.20.1: bytes=32 time=4ms TTL=125
Reply from 192.168.20.1: bytes=32 time=11ms TTL=125
Reply from 192.168.20.1: bytes=32 time=4ms TTL=125

Ping statistics for 192.168.20.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 13ms, Average = 8ms

PC>
```

Week 7: Implement Dijkstra's algorithm to compute the Shortest path through a graph.

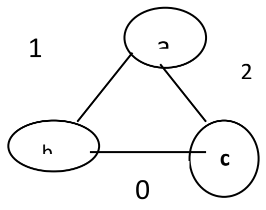
```
//dijkstra’s algorithm or shortest path
#include<stdio.h>
#define INFINITY 9999
#define MAX 10

void dijkstra(int G[MAX][MAX],int n,int startnode);

int main()
{
   int G[MAX][MAX],i,j,n,u;
   printf("Enter no. of vertices:");
   scanf("%d",&n);
   printf("\nEnter the adjacency matrix:\n");

   for(i=0;i<n;i++)
      for(j=0;j<n;j++)
         scanf("%d",&G[i][j]);

   printf("\nEnter the starting node:");
   scanf("%d",&u);
   dijkstra(G,n,u);

   return 0;
}

void dijkstra(int G[MAX][MAX],int n,int startnode)
{

   int cost[MAX][MAX],distance[MAX],pred[MAX];
   int visited[MAX],count,mindistance,nextnode,i,j;

   //pred[] stores the predecessor of each node
   //count gives the number of nodes seen so far
   //create the cost matrix
   for(i=0;i<n;i++)
      for(j=0;j<n;j++)
         if(G[i][j]==0)
            cost[i][j]=INFINITY;
         else
            cost[i][j]=G[i][j];

   //initialize pred[],distance[] and visited[]
   for(i=0;i<n;i++)
   {
      distance[i]=cost[startnode][i];
```

```
    pred[i]=startnode;
    visited[i]=0;
}

distance[startnode]=0;
visited[startnode]=1;
count=1;

while(count<n-1)
{
    mindistance=INFINITY;

    //nextnode gives the node at minimum distance
    for(i=0;i<n;i++)
        if(distance[i]<mindistance&&!visited[i])
        {
            mindistance=distance[i];
            nextnode=i;
        }

        //check if a better path exists through nextnode
        visited[nextnode]=1;
        for(i=0;i<n;i++)
            if(!visited[i])
                if(mindistance+cost[nextnode][i]<distance[i])
                {
                    distance[i]=mindistance+cost[nextnode][i];
                    pred[i]=nextnode;
                }
    count++;
}

//print the path and distance of each node
for(i=0;i<n;i++)
    if(i!=startnode)
    {
        printf("\nDistance of node%d=%d",i,distance[i]);
        printf("\nPath=%d",i);

        j=i;
        do
        {
            j=pred[j];
            printf("<-%d",j);
        }while(j!=startnode);
}
```

```
}
```

OUTPUT:

```
Enter no. of vertices:3
```

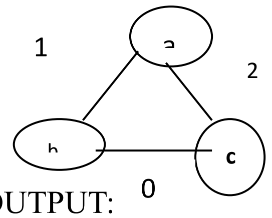


```
Enter the adjacency matrix:
0 1 2
1 0 0
2 0 0

Enter the starting node:0

Distance of node1=1
Path=1<-0
Distance of node2=2
Path=2←0
```

Week 8: Now obtain Routing table art each node using distance vector routing algorithm

```
//Distance vector routing
#include<stdio.h>


struct node
{
   unsigned dist[20];
   unsigned from[20];
}rt[10];


int main()
{
   int dmat[20][20];
   int n,i,j,k,count=0;
   printf("\nEnter the number of nodes : ");
   scanf("%d",&n);
   printf("\nEnter the cost matrix :\n");
   for(i=0;i<n;i++)
      for(j=0;j<n;j++)
      {
         scanf("%d",&dmat[i][j]);
         dmat[i][i]=0;
         rt[i].dist[j]=dmat[i][j];
         rt[i].from[j]=j;
      }
      do
      {
         count=0;
         for(i=0;i<n;i++)
         for(j=0;j<n;j++)
         for(k=0;k<n;k++)
```

```
                if(rt[i].dist[j]>dmat[i][k]+rt[k].dist[j])
                {
                    rt[i].dist[j]=rt[i].dist[k]+rt[k].dist[j];
                    rt[i].from[j]=k;
                    count++;
                }
        }while(count!=0);
        for(i=0;i<n;i++)
        {
            printf("\n\nState value for router %d is \n",i+1);
            for(j=0;j<n;j++)
            {
                printf("\t\nnode %d via %d Distance%d",j+1,rt[i].from[j]+1,rt[i].dist[j]);
            }
        }
    printf("\n\n");
}
```



OUTPUT:

```
Enter the number of nodes : 3




Enter the cost matrix :
0 1 2
1 0 0
2 0 0
```

```
State value for router 1 is

node 1 via 1 Distance0
node 2 via 2 Distance1
node 3 via 2 Distance1

State value for router 2 is

node 1 via 1 Distance1
node 2 via 2 Distance0
node 3 via 3 Distance0

State value for router 3 is

node 1 via 2 Distance1
node 2 via 2 Distance0
node 3 via 3 Distance0
```

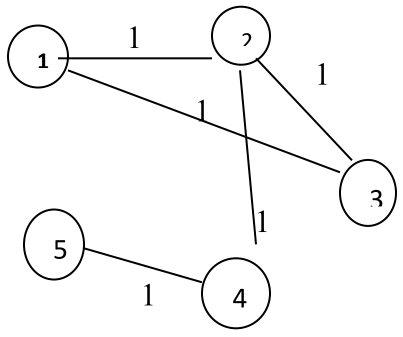
Week 9: Take an example subnet of hosts. Obtain broadcast tree for it.

```
//implement broad cast routing
#include<stdio.h>
#include<conio.h>
int a[10][10],n;
void adj(int k);
void main()
{
int i,j,root;
clrscr();
printf("Enter no.of nodes:");
scanf("%d",&n);
printf("Enter adjacent matrix\n");
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
{
printf("Enter connecting of %d>%d::",i,j);
scanf("%d",&a[i][j]);
}
printf("Enter root node:");
scanf("%d",&root);
adj(root);
}
adj(int k)
{
```

```
int i,j;

printf("Adjacent node of root node::\n");

printf("%d\n",k);

for(j=1;j<=n;j++)

{

if(a[k][j]==1 || a[j][k]==1)

printf("%d\t",j);

}

printf("\n");

for(i=1;i<=n;i++)

{

if((a[k][j]==0) && (a[i][k]==0) && (i!=k))

printf("%d",i);

}

}
```

Output:

```
Enter no.of nodes:5
Enter adjacent matrix
Enter connecting of 1–>1::0
Enter connecting of 1–>2::1
Enter connecting of 1–>3::1
Enter connecting of 1–>4::0
Enter connecting of 1–>5::0
Enter connecting of 2–>1::1
Enter connecting of 2–>2::0
Enter connecting of 2–>3::1
Enter connecting of 2–>4::1
Enter connecting of 2–>5::0
Enter connecting of 3–>1::1
Enter connecting of 3–>2::1
Enter connecting of 3–>3::0
```



```
Enter connecting of 3–>4::0
Enter connecting of 3–>5::0
Enter connecting of 4–>1::0
Enter connecting of 4–>2::1
Enter connecting of 4–>3::0
Enter connecting of 4–>4::0
Enter connecting of 4–>5::1
Enter connecting of 5–>1::0
Enter connecting of 5–>2::0
Enter connecting of 5–>3::0
Enter connecting of 5–>4::1
Enter connecting of 5–>5::0
Enter root node:2
Adjacent node of root node::
2
1 3 4
5
```

Week 10: Write a program for congestion control using leaky bucket algorithm.

```
// implement leaky bucket algorithm
#include<stdio.h>
#include<stdlib.h>

struct packet
{
   int time;
   int size;
}p[50];

int main()
{
   int i,n,m,k=0;
   int bsize,bfilled,outrate;
   printf("Enter the number of packets: ");
   scanf("%d",&n);
   printf("Enter packets in the order of they are arrival time\n");
   for(i=0;i<n;i++)
   {
      printf("Enter the time and size: ");
      scanf("%d%d",&p[i].time,&p[i].size);
   }
   printf("Enter the bucket size: ");
   scanf("%d",&bsize);
   printf("Enter the output rate: ");
   scanf("%d",&outrate);
```

```
m=p[n-1].time;
i=1;
k=0;
bfilled=0;
while(i<=m || bfilled!=0)
{
   printf("\n\nAt time %d",i);
   if(p[k].time==i )
   {
      if(bsize>=bfilled + p[k].size)
      {
         bfilled=bfilled + p[k].size;
         printf("\n%d byte packet is inserted",p[k].size);
         k=k+1;
      }
      else
      {
         printf("\n%d byte packet is discarded",p[k].size);
         k=k+1;
      }
   }
   if(bfilled==0)
   {
      printf("\nNo packets to transmitte");
   }
   else if(bfilled>=outrate)
   {
```

```
            bfilled=bfilled-outrate;
            printf("\n%d bytes transfered",outrate);
        }
        else
        {
            printf("\n%d bytes transfered",bfilled);
            bfilled=0;
        }
        printf("\nPackets in the bucket %d byte",bfilled);
        i++;
    }
    return 0;
}
```

OUTPUT:

```
Enter the number of packets: 3
Enter packets in the order of they are arrival time
Enter the time and size: 1 100
Enter the time and size: 2 400
Enter the time and size: 3 600
Enter the bucket size: 500
Enter the output rate: 200

At time 1
100 byte packet is inserted
100 bytes transfered
Packets in the bucket 0 byte
```

At time 2

400 byte packet is inserted

200 bytes transfered

Packets in the bucket 200 byte

At time 3

600 byte packet is discarded

200 bytes transfered

Packets in the bucket 0 byte

Week11: Capture and Anlayze the Packets using Wire shark for the following Protocols TCP,UDP

Introduction

Wireshark is a network packet analyzer. A network packet analyzer presents captured packet data in as much detail as possible

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education.

Wireshark is cross-platform, using the Qt widget toolkit in current releases to implement its user interface, and using pcap to capture packets; it runs on Linux, macOS, BSD, Solaris, some other Unix-like operating systems, and Microsoft Windows

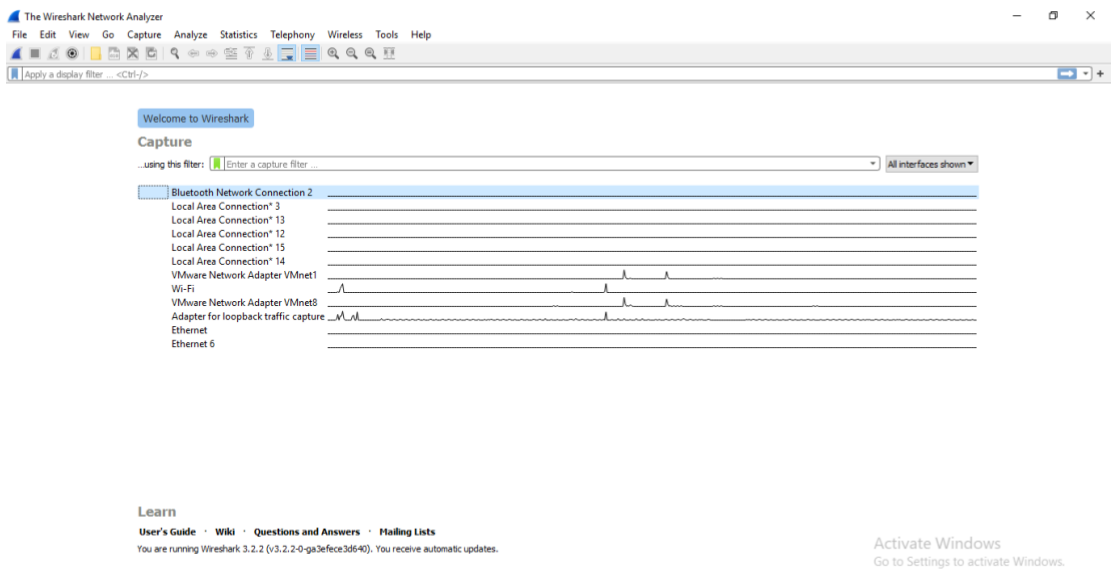
Features:

- Data can be captured "from the wire" from a live network connection or read from a file of already-captured packets.
- Live data can be read from different types of networks, including Ethernet, IEEE 802.11, PPP, and loopback.
- Captured network data can be browsed via a GUI, or via the terminal (command line) version of the utility, TShark.
- Captured files can be programmatically edited or converted via command-line switches to the "editcap" program.
- Data display can be refined using a display filter.

Wireshark can color packets based on rules that match particular fields in packets, to help the user identify the types of traffic at a glance. A default set of rules is provided; users can change existing rules for coloring packets, add new rules, or remove rules.

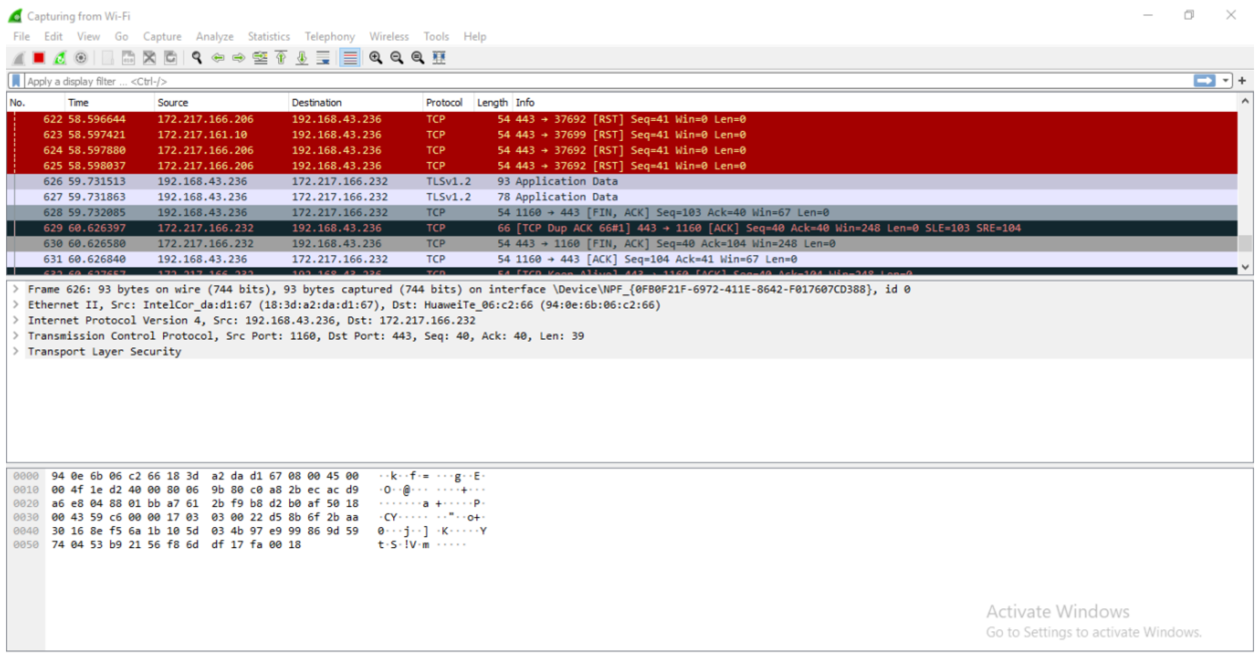
Wireshark uses NMAP's Packet Capture library(called npcap).

Getting Up and Running: After installation launch Wireshark, approve the administrator or superuser privileges and you will be presented with a window that looks like this:


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
All interfaces shown
Bluetooth Network Connection 2
Local Area Connection* 3
Local Area Connection* 13
Local Area Connection* 12
Local Area Connection* 15
Local Area Connection* 14
VMware Network Adapter VMnet1
Wi-Fi
VMware Network Adapter VMnet8
Adapter for loopback traffic capture
Ethernet
Ethernet 6
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
Activate Windows
Go to Settings to activate Windows.

You are running Wireshark 3.2.2 (v3.2.2-0-ga3efece3d640). You receive automatic updates.

This window shows the interfaces on your device. To start sniffing select one interface and click on the bluefin icon on the top left. The data capture screen has three panes. The top pane shows real-time traffic, the middle one shows information about the chosen packet and the bottom pane shows the raw packet data. The top pane shows source address(IPv4 or IPv6) destination address, source and destination ports, protocol to which the packet belongs to and additional information about the packet.


Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Activate Windows
Go to Settings to activate Windows.

```
No.    Time         Source            Destination       Protocol  Length Info
   622 58.596644    172.217.166.206   192.168.43.236    TCP       54 443 → 37692 [RST] Seq=41 Win=0 Len=0
   623 58.597421    172.217.161.10    192.168.43.236    TCP       54 443 → 37699 [RST] Seq=41 Win=0 Len=0
   624 58.597880    172.217.166.206   192.168.43.236    TCP       54 443 → 37692 [RST] Seq=41 Win=0 Len=0
   625 58.598037    172.217.166.206   192.168.43.236    TCP       54 443 → 37692 [RST] Seq=41 Win=0 Len=0
   626 59.731513    192.168.43.236    172.217.166.232   TLSv1.2   93 Application Data
   627 59.731863    192.168.43.236    172.217.166.232   TLSv1.2   78 Application Data
   628 59.732085    192.168.43.236    172.217.166.232   TCP       54 1160 → 443 [FIN, ACK] Seq=103 Ack=40 Win=67 Len=0
   629 60.626397    172.217.166.232   192.168.43.236    TCP       66 [TCP Dup ACK 66#1] 443 → 1160 [ACK] Seq=40 Ack=40 Win=248 Len=0 SLE=103 SRE=104
   630 60.626580    172.217.166.232   192.168.43.236    TCP       54 443 → 1160 [FIN, ACK] Seq=40 Ack=104 Win=248 Len=0
   631 60.626840    192.168.43.236    172.217.166.232   TCP       54 1160 → 443 [ACK] Seq=104 Ack=41 Win=67 Len=0
```

```
> Frame 626: 93 bytes on wire (744 bits), 93 bytes captured (744 bits) on interface \Device\NPF_{0FB0F21F-6972-411E-8642-F017607CD388}, id 0
> Ethernet II, Src: IntelCor_da:d1:67 (18:3d:a2:da:d1:67), Dst: HuaweiTe_06:c2:66 (94:0e:6b:06:c2:66)
> Internet Protocol Version 4, Src: 192.168.43.236, Dst: 172.217.166.232
> Transmission Control Protocol, Src Port: 1160, Dst Port: 443, Seq: 40, Ack: 40, Len: 39
> Transport Layer Security
```

```
0000  94 0e 6b 06 c2 66 18 3d  a2 da d1 67 08 00 45 00   ··k··f·= ···g··E·
0010  00 4f 1e d2 40 00 80 06  9b 80 c0 a8 2b ec ac d9   ·O··@··· ····+···
0020  a6 e8 04 88 01 bb a7 61  2b f9 b8 d2 b0 af 50 18   ·······a +·····P·
0030  00 43 59 c6 00 00 17 03  03 00 22 d5 8b 6f 2b aa   ·CY····· ··"··o+·
0040  30 16 8e f5 6a 1b 10 5d  03 4b 97 e9 99 86 9d 59   0···j··] ·K·····Y
0050  74 04 53 b9 21 56 f8 6d  df 17 fa 00 18            t·S·!V·m ·····
```

As mentioned, Wireshark uses a color-coding system for data visualization. Each packet is marked with a different color that represents different types of traffic. For example, TCP traffic is usually highlighted with blue, while black is used to indicate packets containing errors.

Of course, you don’t have to memorize the meaning behind each color. Instead, you can check on the spot:

1. Right-click on the packet you wish to examine.

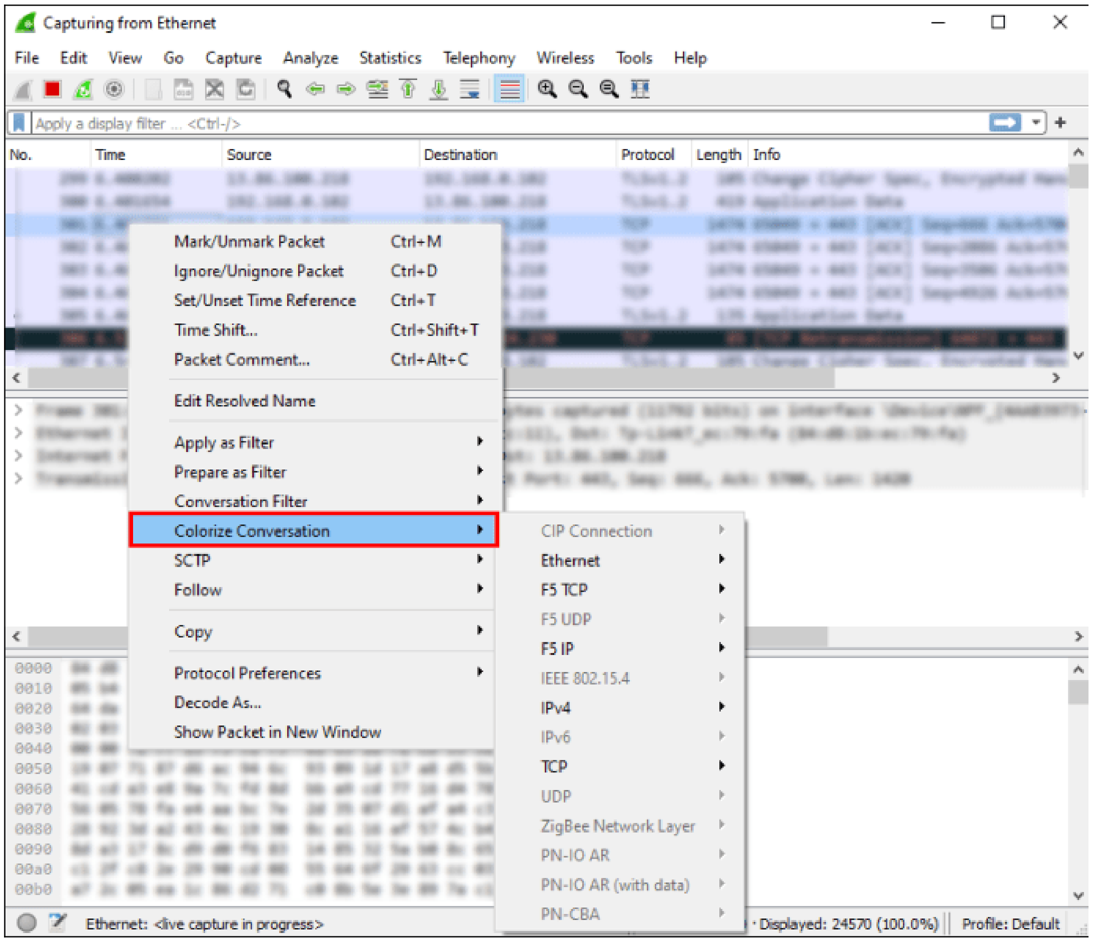
2. Select the “View” tab from the toolbar at the top of the screen.

3. Choose “Coloring Rules” from the drop-down panel.

You’ll see the option to customize the colorization to your liking. However, if you only want to change the coloring rules temporarily, follow these steps:

1. Right-click on the packet in the packet list pane.

2. From the list of options, select “Colorize With Filter.”


Capturing from Ethernet
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
Mark/Unmark Packet Ctrl+M
Ignore/Unignore Packet Ctrl+D
Set/Unset Time Reference Ctrl+T
Time Shift... Ctrl+Shift+T
Packet Comment... Ctrl+Alt+C
Edit Resolved Name
Apply as Filter
Prepare as Filter
Conversation Filter
Colorize Conversation
SCTP
Follow
Copy
Protocol Preferences
Decode As...
Show Packet in New Window
CIP Connection
Ethernet
F5 TCP
F5 UDP
F5 IP
IEEE 802.15.4
IPv4
IPv6
TCP
UDP
ZigBee Network Layer
PN-IO AR
PN-IO AR (with data)
PN-CBA
Ethernet: <live capture in progress>
Displayed: 24570 (100.0%)
Profile: Default

3. Choose the color with which you want to label it.

Protocol

A protocol is a guideline that determines the data transmission between different devices that are connected to the same network. Each Wireshark packet contains a protocol, and you can bring it up by using the display filter. Here’s how:

1. At the top of the Wireshark window, click on the “Filter” dialog box.
2. Enter the name of the protocol you want to examine. Typically, protocol titles are written in lowercase letters.
3. Click “Enter” or “Apply” to enable the display filter.

Length

The length of a Wireshark packet is determined by the number of bytes captured in that particular network snippet. That number usually corresponds with the number of raw data bytes listed at the bottom of the Wireshark window.

If you want to examine the distribution of lengths, open the “Packet Lengths” window. All the info is divided into the following columns:

- Packet lengths
- Count
- Average
- Min Val/Max Val
- Rate
- Percent
- Burst rate
- Burst start

Info

If there are any anomalies or similar items within a particular captured packet, Wireshark will note it. The information will then be displayed in the packet list pane for further examination. That way, you’ll have a clear picture of atypical network behavior, which will result in speedier reactions.

How can I filter the packet data?

Filtering is an efficient feature that allows you to look into the specifics of a particular data sequence. There are two types of Wireshark filters: capture and display. Capture filters are there to restrict the packet capture to fit specific demands. In other words, you can sift through different types of traffic by applying a capture filter. As the name suggests, display filters allow you to hone in on a particular element of the packet, from packet length to protocol.

Applying a filter is a pretty straightforward process. You can type the filter title in the dialog box at the top of the Wireshark window. In addition, the software will usually auto-complete the name of the filter.

Analyzing Packets

Once the packets are captured, Wireshark organizes them in a detailed packet list pane that's incredibly easy to read. If you want to access the information regarding a single packet, all you have to do is locate it on the list and click. You can also further expand the tree to access the details of each protocol contained within the packet.

Open the "Analyze" tab in the toolbar at the top of the Wireshark window.

From the drop-down list, select "Display Filter."

Browse through the list and click on the one you want to apply.

Finally, here are some common Wireshark filters that can come in handy:

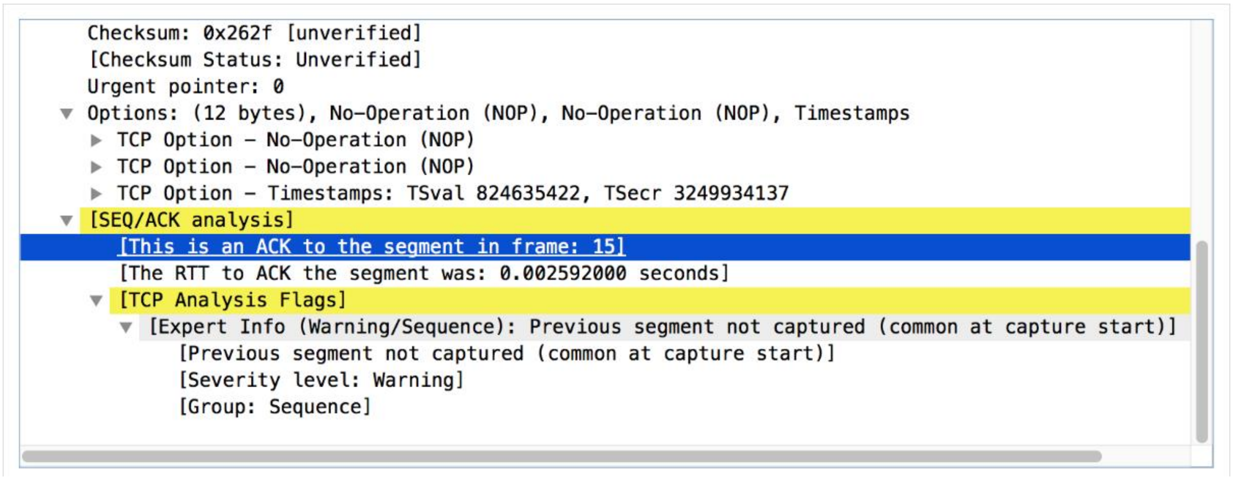
• To only view the source and destination IP address, use: "ip.src==IP-address and ip.dst==IP-address"

• To only view SMTP traffic, type: "tcp.port eq 25"

• To capture all subnet traffic, apply: "net 192.168.0.0/24"

• To capture everything but the ARP and DNS traffic, use: "port not 53 and not arp"

TCP Analysis



```
    Checksum: 0x262f [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▼ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
    ▶ TCP Option - No-Operation (NOP)
    ▶ TCP Option - No-Operation (NOP)
    ▶ TCP Option - Timestamps: TSval 824635422, TSecr 3249934137
  ▼ [SEQ/ACK analysis]
      [This is an ACK to the segment in frame: 15]
      [The RTT to ACK the segment was: 0.002592000 seconds]
    ▼ [TCP Analysis Flags]
      ▼ [Expert Info (Warning/Sequence): Previous segment not captured (common at capture start)]
          [Previous segment not captured (common at capture start)]
          [Severity level: Warning]
          [Group: Sequence]
```

TCP Analysis flags are added to the TCP protocol tree under “SEQ/ACK analysis”

Next expected sequence number

The last-seen sequence number plus segment length. Set when there are no analysis flags and for zero window probes. This is initially zero and calculated based on the previous packet in the same TCP flow. Note that this may not be the same as the tcp.nxtseq protocol field.

Next expected acknowledgment number

The last-seen sequence number for segments. Set when there are no analysis flags and for zero window probes.

Last-seen acknowledgment number

Always set. Note that this is not the same as the next expected acknowledgment number.

Last-seen acknowledgment number

Always updated for each packet. Note that this is not the same as the next expected acknowledgment number.

TCP Dup ACK *<frame>#<acknowledgment number>*

- The segment size is zero.

- The window size is non-zero and hasn't changed.
- The next expected sequence number and last-seen acknowledgment number are non-zero (i.e., the connection has been established).
- SYN, FIN, and RST are not set.

TCP Fast Retransmission

- This is not a keepalive packet.
- In the forward direction, the segment size is greater than zero or the SYN or FIN is set.
- The next expected sequence number is greater than the current sequence number.
- We have at least two duplicate ACKs in the reverse direction.
- The current sequence number equals the next expected acknowledgment number.
- We saw the last acknowledgment less than 20ms ago.

TCP Keep-Alive

Set when the segment size is zero or one, the current sequence number is one byte less than the next expected sequence number, and none of SYN, FIN, or RST are set.

Supersedes "Fast Retransmission", "Out-Of-Order", "Spurious Retransmission", and "Retransmission".

TCP Window Update

- The segment size is zero.
- The window size is non-zero and not equal to the last-seen window size.
- The sequence number is equal to the next expected sequence number.
- The acknowledgment number is equal to the last-seen acknowledgment number,
- or to the next expected sequence number when answering to a ZeroWindowProbe.
- None of SYN, FIN, or RST are set.

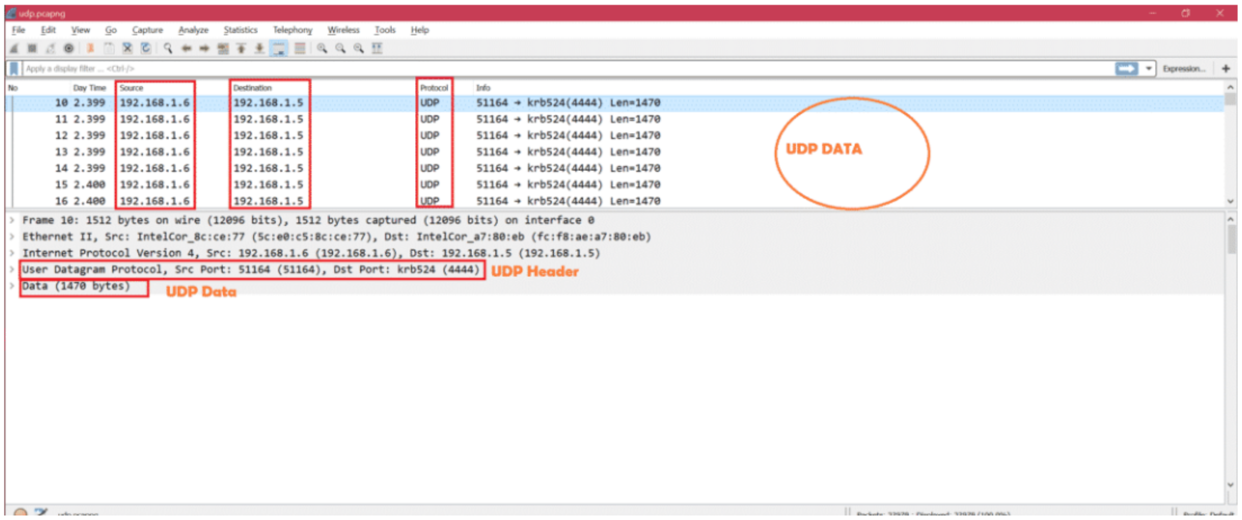
UDP:

Source port: The source port number of the packet. Example: 4444.

Destination port: The destination port number of packet. Example: 51164.

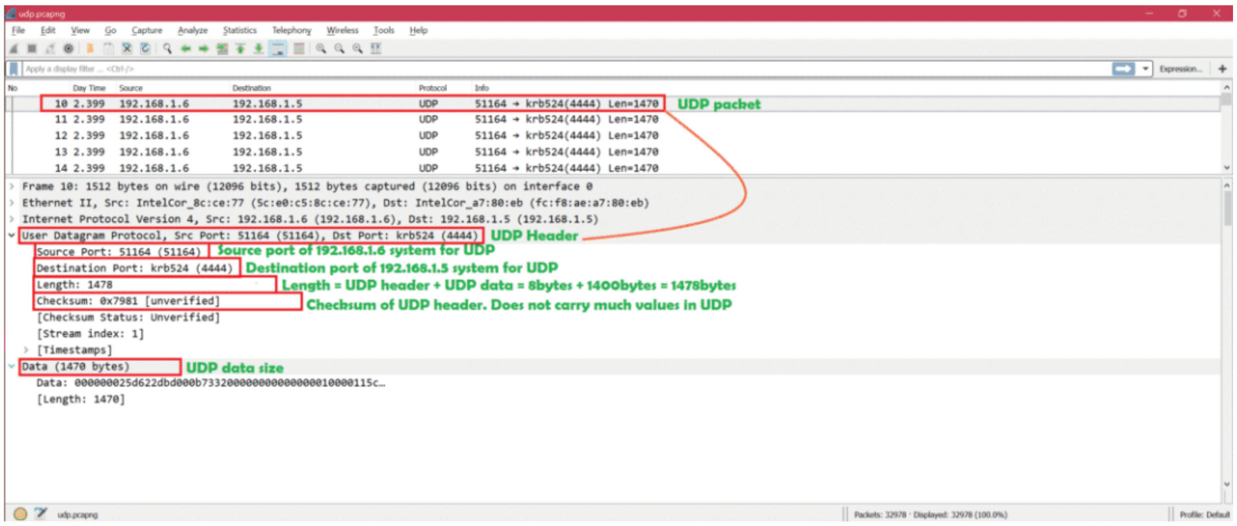
Length: The length of UDP Data + UDP header.

Checksum: Checksum is present to detect error. Unlike TCP, Checksum calculation is not mandatory in UDP. No Error control or flow control is provided by UDP. Hence UDP depends on IP and ICMP for error reporting.



udp.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
UDP DATA
UDP Header
UDP Data

```
No   Day Time  Source        Destination   Protocol  Info
  10 2.399     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470
  11 2.399     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470
  12 2.399     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470
  13 2.399     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470
  14 2.399     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470
  15 2.400     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470
  16 2.400     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470

> Frame 10: 1512 bytes on wire (12096 bits), 1512 bytes captured (12096 bits) on interface 0
> Ethernet II, Src: IntelCor_8c:ce:77 (5c:e0:c5:8c:ce:77), Dst: IntelCor_a7:80:eb (fc:f8:ae:a7:80:eb)
> Internet Protocol Version 4, Src: 192.168.1.6 (192.168.1.6), Dst: 192.168.1.5 (192.168.1.5)
> User Datagram Protocol, Src Port: 51164 (51164), Dst Port: krb524 (4444)
> Data (1470 bytes)
```



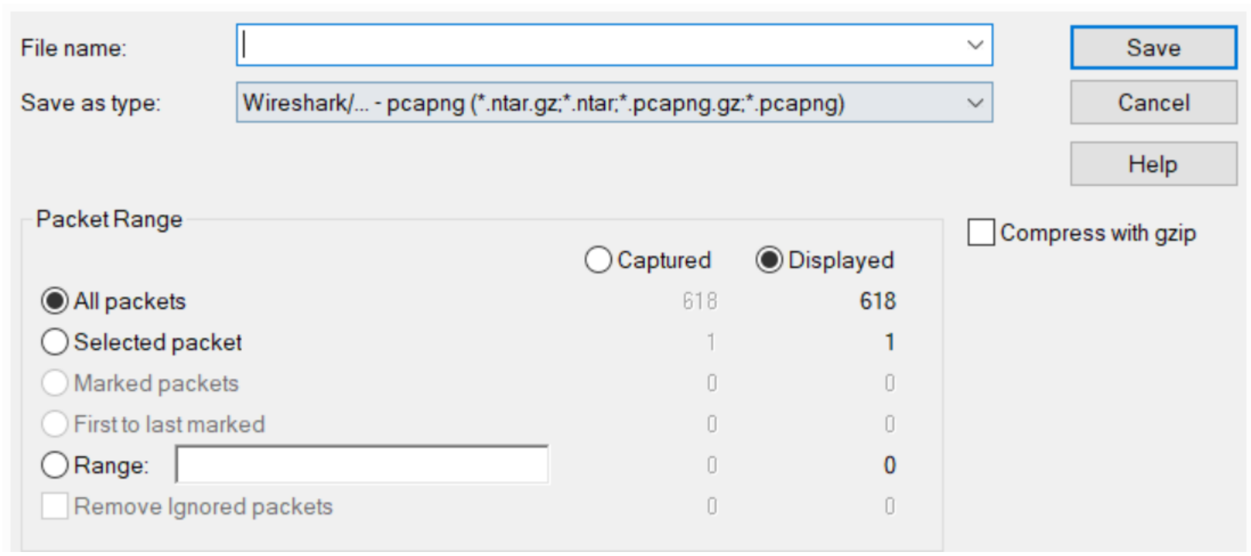
udp.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
UDP packet
UDP Header
Source port of 192.168.1.6 system for UDP
Destination port of 192.168.1.5 system for UDP
Length = UDP header + UDP data = 8bytes + 1400bytes = 1478bytes
Checksum of UDP header. Does not carry much values in UDP
UDP data size
udp.pcapng
Packets: 32978 · Displayed: 32978 (100.0%)
Profile: Default

```
No   Day Time  Source        Destination   Protocol  Info
  10 2.399     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470
  11 2.399     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470
  12 2.399     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470
  13 2.399     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470
  14 2.399     192.168.1.6   192.168.1.5   UDP       51164 → krb524(4444) Len=1470

> Frame 10: 1512 bytes on wire (12096 bits), 1512 bytes captured (12096 bits) on interface 0
> Ethernet II, Src: IntelCor_8c:ce:77 (5c:e0:c5:8c:ce:77), Dst: IntelCor_a7:80:eb (fc:f8:ae:a7:80:eb)
> Internet Protocol Version 4, Src: 192.168.1.6 (192.168.1.6), Dst: 192.168.1.5 (192.168.1.5)
v User Datagram Protocol, Src Port: 51164 (51164), Dst Port: krb524 (4444)
    Source Port: 51164 (51164)
    Destination Port: krb524 (4444)
    Length: 1478
    Checksum: 0x7981 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 1]
  > [Timestamps]
v Data (1470 bytes)
    Data: 000000025d622dbd000b733200000000000000010000115c…
    [Length: 1470]
```

Applications:

- DNS, DHCP, BOOTP, TFTP, RIP etc.
- Real time protocol which cannot tolerate delay.
- Used in some multicasting

Save as :



File name:

Save as type: Wireshark/... - pcapng (*.ntar.gz;*.ntar;*.pcapng.gz;*.pcapng)

Packet Range

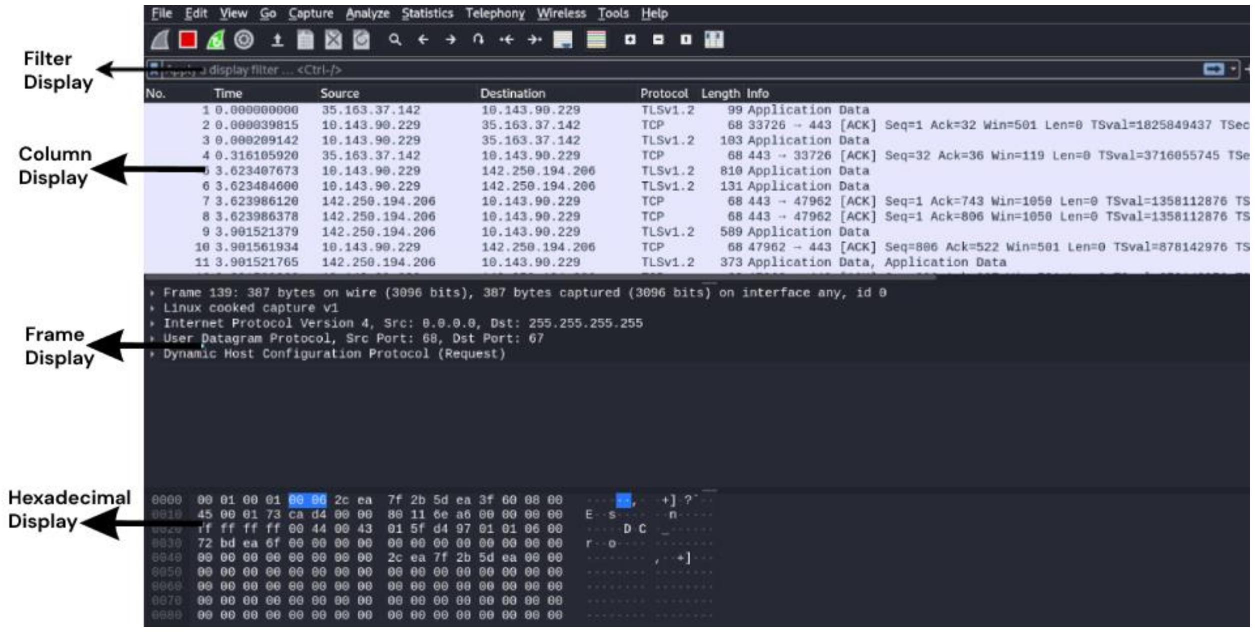
	Captured	[✓ Displayed]
[✓ All packets]	618	618
Selected packet	1	1
Marked packets	0	0
First to last marked	0	0
Range:	0	0
Remove Ignored packets	0	0

Compress with gzip

Week12: Capture and Anlayze the Packets using Wire shark for the following Protocols HTTP ,DNS

we can analyze DNS queries easily. We shall be following the below steps:

- In the menu bar, Capture → Interfaces.
- Select a particular Ethernet adapter and click start.
- After this, browse to any web address and then return to Wireshark. Browsing would get packets captured and in Wireshark click the stop in the Capture menu to stop the capture.
- If you haven't got the packet list by now, you can access it using **Edit → Find Packets**. This will give you the packet list.
- Since we are going to analyze DNS we shall be studying only DNS packets and to get DNS packets, only you can apply DNS in the filters above.


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
[illegible] a display filter ... <Ctrl-/>
Filter Display
Column Display
Frame Display
Hexadecimal Display

```
No.     Time          Source            Destination       Protocol  Length Info
      1 0.000000000   35.163.37.142     10.143.90.229     TLSv1.2     99 Application Data
      2 0.000039815   10.143.90.229     35.163.37.142     TCP         68 33726 → 443 [ACK] Seq=1 Ack=32 Win=501 Len=0 TSval=1825849437 TSec
      3 0.000209142   10.143.90.229     35.163.37.142     TLSv1.2    103 Application Data
      4 0.316105920   35.163.37.142     10.143.90.229     TCP         68 443 → 33726 [ACK] Seq=32 Ack=36 Win=119 Len=0 TSval=3716055745 TSe
      5 3.623407673   10.143.90.229     142.250.194.206   TLSv1.2    810 Application Data
      6 3.623484600   10.143.90.229     142.250.194.206   TLSv1.2    131 Application Data
      7 3.623986120   142.250.194.206   10.143.90.229     TCP         68 443 → 47962 [ACK] Seq=1 Ack=743 Win=1050 Len=0 TSval=1358112876 TS
      8 3.623986378   142.250.194.206   10.143.90.229     TCP         68 443 → 47962 [ACK] Seq=1 Ack=806 Win=1050 Len=0 TSval=1358112876 TS
      9 3.901521379   142.250.194.206   10.143.90.229     TLSv1.2    589 Application Data
     10 3.901561934   10.143.90.229     142.250.194.206   TCP         68 47962 → 443 [ACK] Seq=806 Ack=522 Win=501 Len=0 TSval=878142976 TS
     11 3.901521765   142.250.194.206   10.143.90.229     TLSv1.2    373 Application Data, Application Data
```

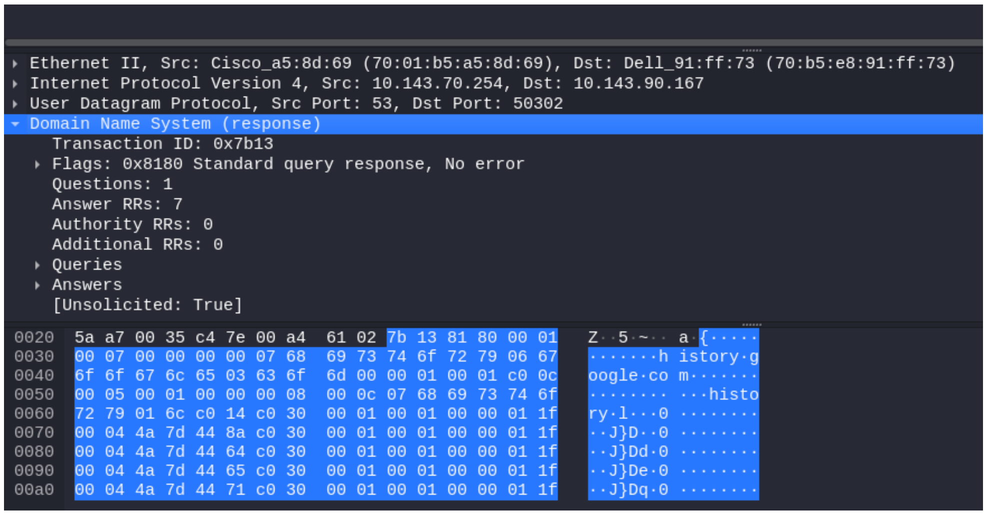
```
▸ Frame 139: 387 bytes on wire (3096 bits), 387 bytes captured (3096 bits) on interface any, id 0
▸ Linux cooked capture v1
▸ Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
▸ User Datagram Protocol, Src Port: 68, Dst Port: 67
▸ Dynamic Host Configuration Protocol (Request)
```

```
0000  00 01 00 01 00 06 2c ea  7f 2b 5d ea 3f 60 08 00   ······,·  ·+]·?`··
0010  45 00 01 73 ca d4 00 00  80 11 6e a6 00 00 00 00   E··s····  ··n·····
0020  ff ff ff ff 00 44 00 43  01 5f d4 97 01 01 06 00   ·····D·C  ·_······
0030  72 bd ea 6f 00 00 00 00  00 00 00 00 00 00 00 00   r··o····  ········
0040  00 00 00 00 00 00 00 00  2c ea 7f 2b 5d ea 00 00   ········  ,··+]···
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········  ········
```

A basic **DNS response** has:

1. **Transaction Id**-for identification of the communication done.
2. **Flags**-for verification of response whether it is valid or not.
3. **Questions**-default is 1 for any request sent or received. It mainly denotes whether you have queried for something or not.
4. **Answers**-default is 0 if the response is sent, and it's 1 if received. If the received packet is viewed then the Answers section has the IP address of the desired domain name along with **Time to Live** which is basically a counter which expires after its allotted time.

See the details of dns : **In command Prompt : ipconfig /displaydns**



```
▸ Ethernet II, Src: Cisco_a5:8d:69 (70:01:b5:a5:8d:69), Dst: Dell_91:ff:73 (70:b5:e8:91:ff:73)
▸ Internet Protocol Version 4, Src: 10.143.70.254, Dst: 10.143.90.167
▸ User Datagram Protocol, Src Port: 53, Dst Port: 50302
▾ Domain Name System (response)
    Transaction ID: 0x7b13
  ▸ Flags: 0x8180 Standard query response, No error
    Questions: 1
    Answer RRs: 7
    Authority RRs: 0
    Additional RRs: 0
  ▸ Queries
  ▸ Answers
    [Unsolicited: True]

0020  5a a7 00 35 c4 7e 00 a4  61 02 7b 13 81 80 00 01   Z··5·~·· a·{·····
0030  00 07 00 00 00 00 07 68  69 73 74 6f 72 79 06 67   ·······h istory·g
0040  6f 6f 67 6c 65 03 63 6f  6d 00 00 01 00 01 c0 0c   oogle·co m·······
0050  00 05 00 01 00 00 00 08  00 0c 07 68 69 73 74 6f   ········ ···histo
0060  72 79 01 6c c0 14 c0 30  00 01 00 01 00 00 01 1f   ry·l···0 ········
0070  00 04 4a 7d 44 8a c0 30  00 01 00 01 00 00 01 1f   ··J}D··0 ········
0080  00 04 4a 7d 44 64 c0 30  00 01 00 01 00 00 01 1f   ··J}Dd·0 ········
0090  00 04 4a 7d 44 65 c0 30  00 01 00 01 00 00 01 1f   ··J}De·0 ········
00a0  00 04 4a 7d 44 71 c0 30  00 01 00 01 00 00 01 1f   ··J}Dq·0 ········
```

HTTP:

The Hyper Text Transport Protocol is a text-based request-response client-server protocol. A HTTP client (e.g. a web browser such as Mozilla) performs a HTTP request to a HTTP server (e.g. the Apache HTTP server), which in return will issue a HTTP response. The HTTP protocol header is text-based, where headers are written in text lines.

Request from client

```
GET / HTTP/1.1
Host: www.freebsd.org
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.7) Gecko/20050414 Firefox/1.0.3
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
If-Modified-Since: Mon, 09 May 2005 21:01:30 GMT
If-None-Match: "26f731-8287-427fcfaa"
```

Response from server

```
HTTP/1.1 200 OK
Date: Fri, 13 May 2005 05:51:12 GMT
Server: Apache/1.3.x LaHonda (Unix)
Last-Modified: Fri, 13 May 2005 05:25:02 GMT
ETag: "26f725-8286-42843a2e"
Accept-Ranges: bytes
Content-Length: 33414
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html
```

Display Filter

A complete list of HTTP display filter fields can be found in the display filter reference

Show only the http based traffic:

```
http
```

Show only the famous "404: page not found" responses:

```
http.response.code == 404
```

Capture Filter

You cannot directly filter HTTP protocols while capturing. However, if you know the TCP port used (see above), you can filter on that one.

Capture HTTP traffic over the default port (80):

```
tcp port 80
```

Capture HTTP traffic over the default SSL port (443):

```
tcp port 443
```