PRESIDENCY UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Network Programing lab (CSE 257)

VI B.Tech 6th Semester A.Y (2021-22)

Instructor Incharge **:** Ms. Bhavana A

Course Instructors **: Ms. Bhavana A, Dr. G Shanmugarathinam, Dr. Anandraj S P, Ms. Impa, Mr. Shankar J, Ms. Akshatha Y, Mr. Murthy D H R, Ms. Kokila, Mr.Yogesh Gajmal**

Course Credit Structure **: 0-4-2 (2 Credits)**

Module-1

DOS Commands

1.PING Command

How to check internet connection in CMD

To check whether your internet connection works, you can use Command Prompt to test your connection to a certain website or internet location. To do that, you can use the ping network command, followed by a web address or IP address. For instance, you can check the connectivity to GOOGLE without opening a web browser, by typing the command " ping www.google.com." Then press Enter on your keyboard.

Ping is used to check the connectivity with other devices on the network, for example computers, routers, switches etc. Select Start > Programs > Accessories > Command Prompt. This will give you a window like the one below.

Type *C:\>ping x.x.x.x*

By default, ping sends four ICMP Echo Request packets each of 32 bytes. The response packets are called ICMP Echo Reply Packets.

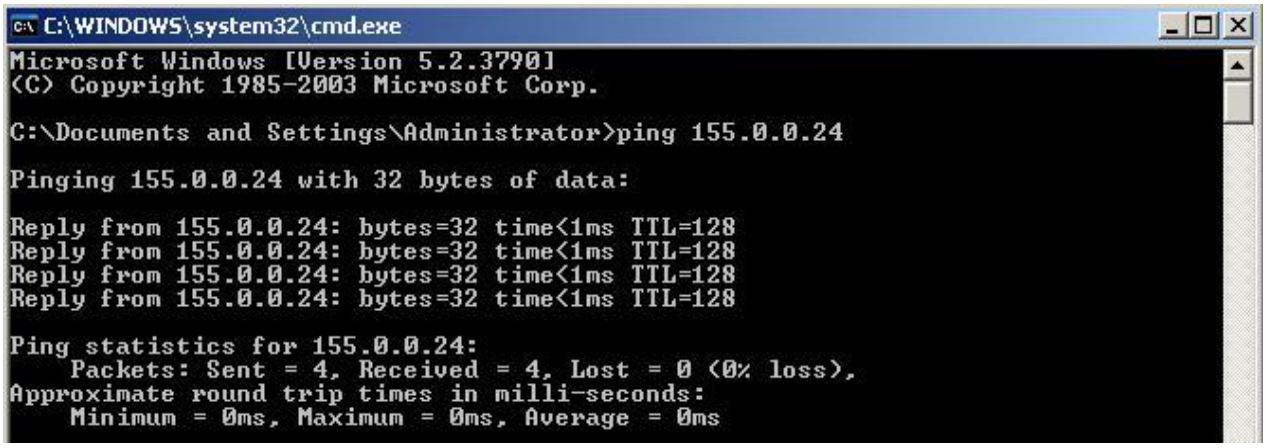
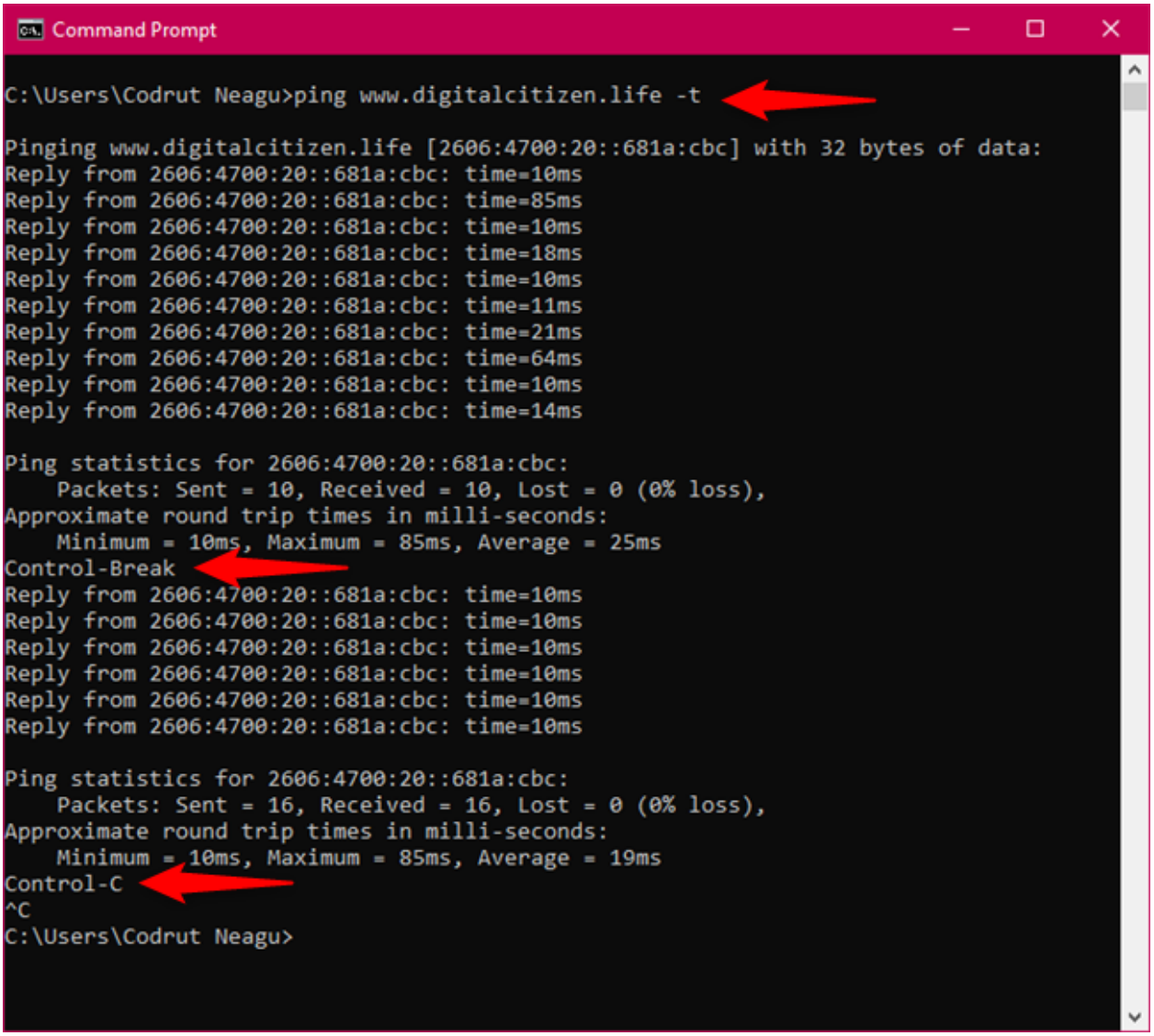


Fig : The Ping Command

Now Type *C:\>ping x.x.x.x –t*

- The ping command also allows you to use the handy "-t" parameter, which enables you to ping the specified address forever until it's manually stopped. For instance, we typed "ping -t www.digitalcitizen.life." After some time, we wanted to see some connection statistics and we used the keyboard combination "CTRL + Break." This shows the averages of the ping commands run until then.

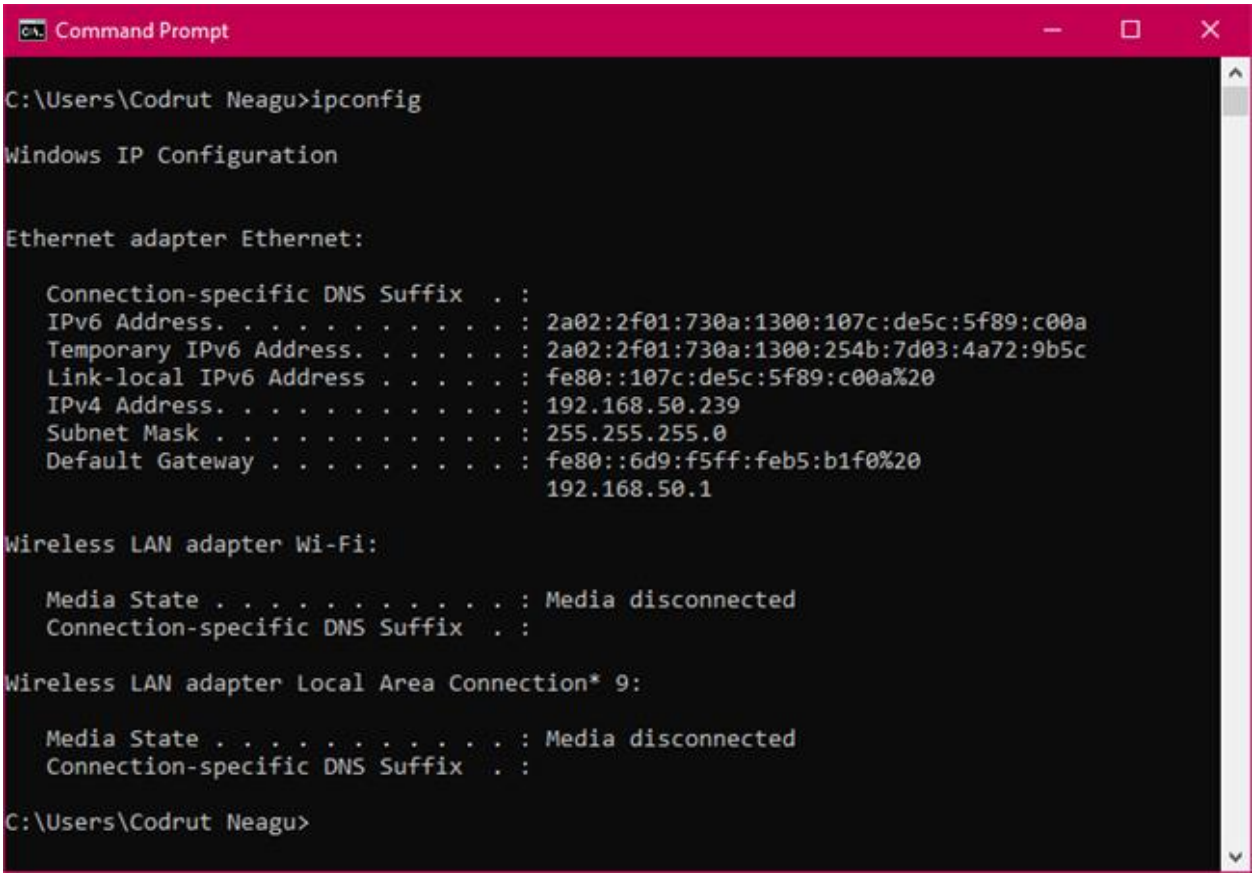
- "–t" switch will continue to send packets to the destination until user stops this by pressing *Ctrl + C*



2. IPCONFIG Command

How can I see all the network adapters on my computer using CMD?

To obtain detailed information about your network adapters and connections, use the ipconfig command. Open Command Prompt, type ipconfig, and press Enter. As you can see in the screenshot below, when you run this command, Windows displays the list of all the active network devices, whether they're connected or disconnected, and their IP addresses. You also get details such as their default gateway IP addresses, subnet masks and the state of each network adapter.



Displays full TCP/IP configuration of all network adapters (Ethernet cards) installed in your system. Type the following command in the command prompt.

C:\ipconfig

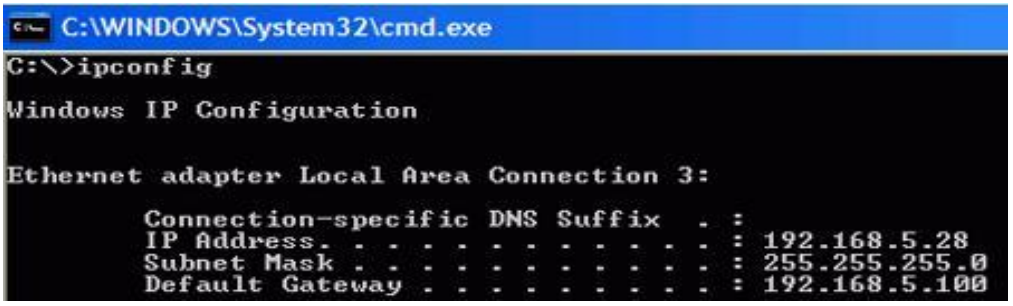
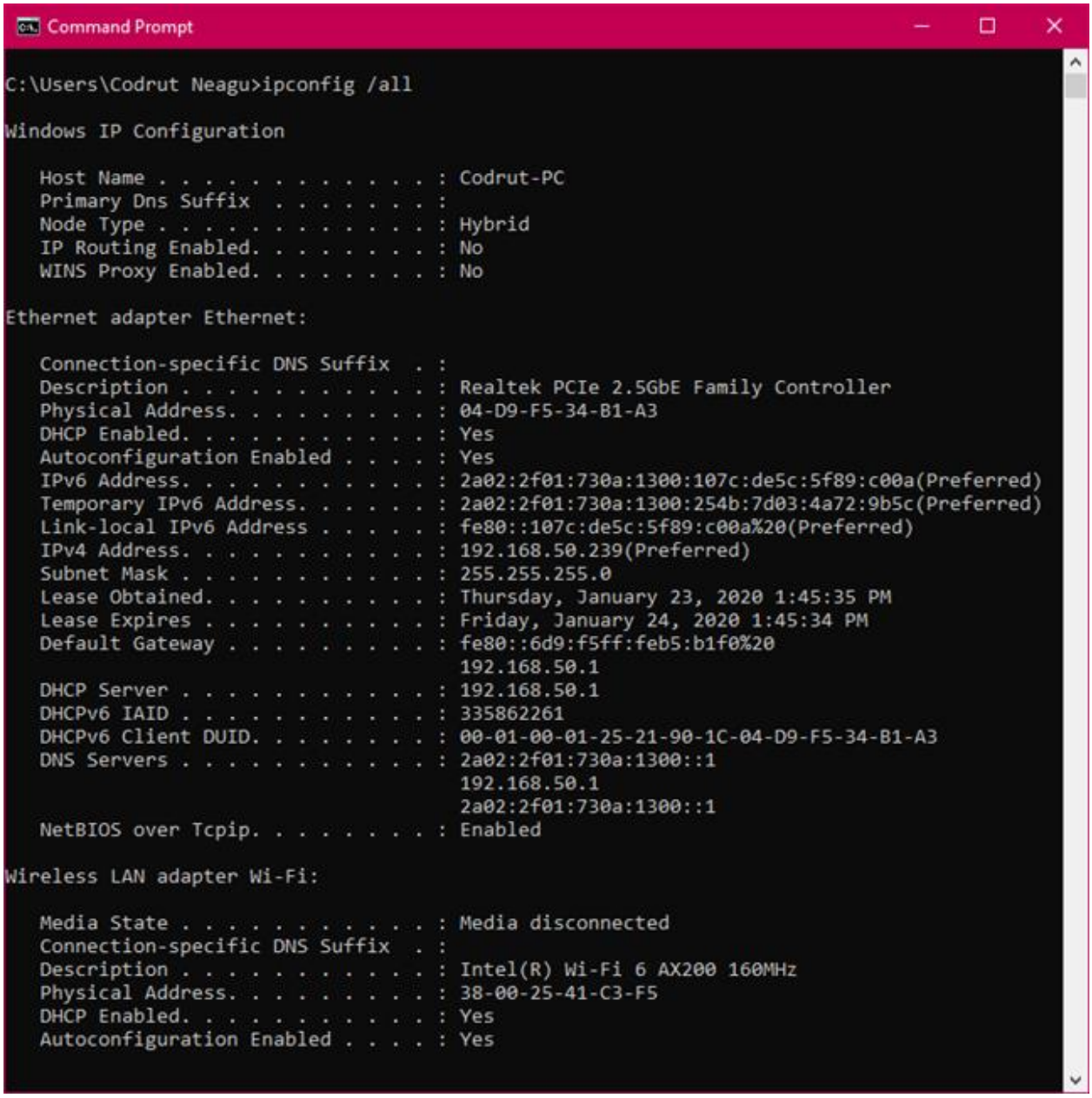


Figure 2: The IPCONFIG Command

Now type *C:\ipconfig /all*

If you add the /all switch to the ipconfig command, you can get to a whole new level of detail: DNS information, the MAC (Media Access Control) (in the Physical Address field), and other information about each network component. Check out the picture below to see a sample of what you get from the "ipconfig /all" command.



Ip config has a number of switches the most common are**:**
ipconfig /all – displays more information about the network setup on your systems including the MAC address.
ipconfig /release – release the current IP address
ipconfig /renew – renew IP address
ipconfig /? -shows help
ipconfig/flushdns – flush the dns cache

How to check your network connection in CMD

If you want to check whether your network connection to the router is operating as it should, you can use a combination of the commands ipconfig and ping. First, get some cmd nic info about your adapter. In other words, open Command Prompt and run ipconfig. In the list of results, identify the network

adapter that's used for connecting to the network you want to test. Then, in its details, find the IP address of your router and note it down. For example, if we'd want to check our Ethernet network connection, we'd run ipconfig and see that our router's IP address is 192.168.50.1.

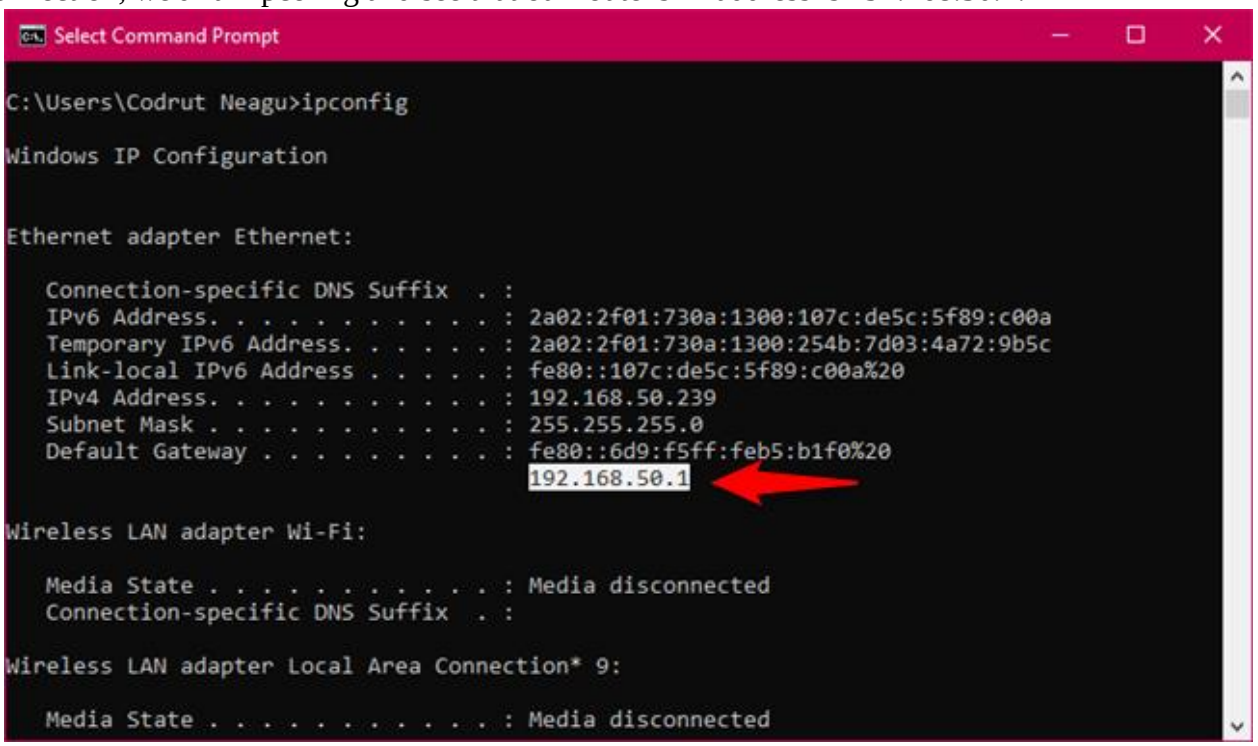


Figure :Running ipconfig to identify the IP address of the router

The next step is to check that the network connection between the router and the computer is OK. To do that, it's enough to run the ping command on the router's IP address. In our example, that would mean that we have to run this command in CMD: ping 192.168.50.1.

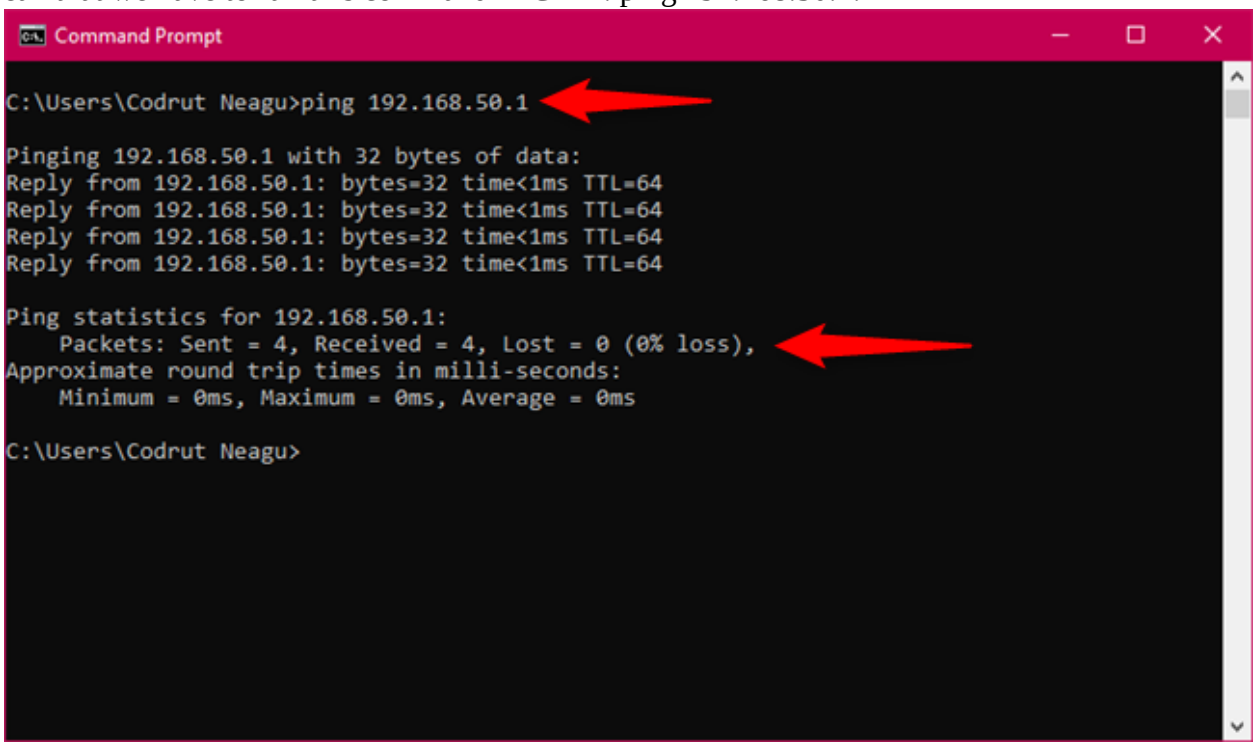


Figure:Pinging the router to check the network connection

If there are no packets lost, then the network connection tested is running well. Otherwise, there's a problem somewhere between your computer and the router, in which case you should check that your PC's network adapter is configured correctly, that the Ethernet cable is OK (if you're using a wired connection), and that the router is configured properly.

How to renew the IP address of your network adapter

When your network connection doesn't work as it should, your network adapter might not have the right IP address assigned. A quick way of trying to solve this issue is to renew its IP address and, fortunately, you can do that quickly, straight from the Command Prompt. Open CMD and run the following commands: ipconfig /release and ipconfig /renew. The first one (ipconfig /release) forces your network adapter to drop its assigned IP address, and the second command (ipconfig /renew) renews the network adapter's IP address.

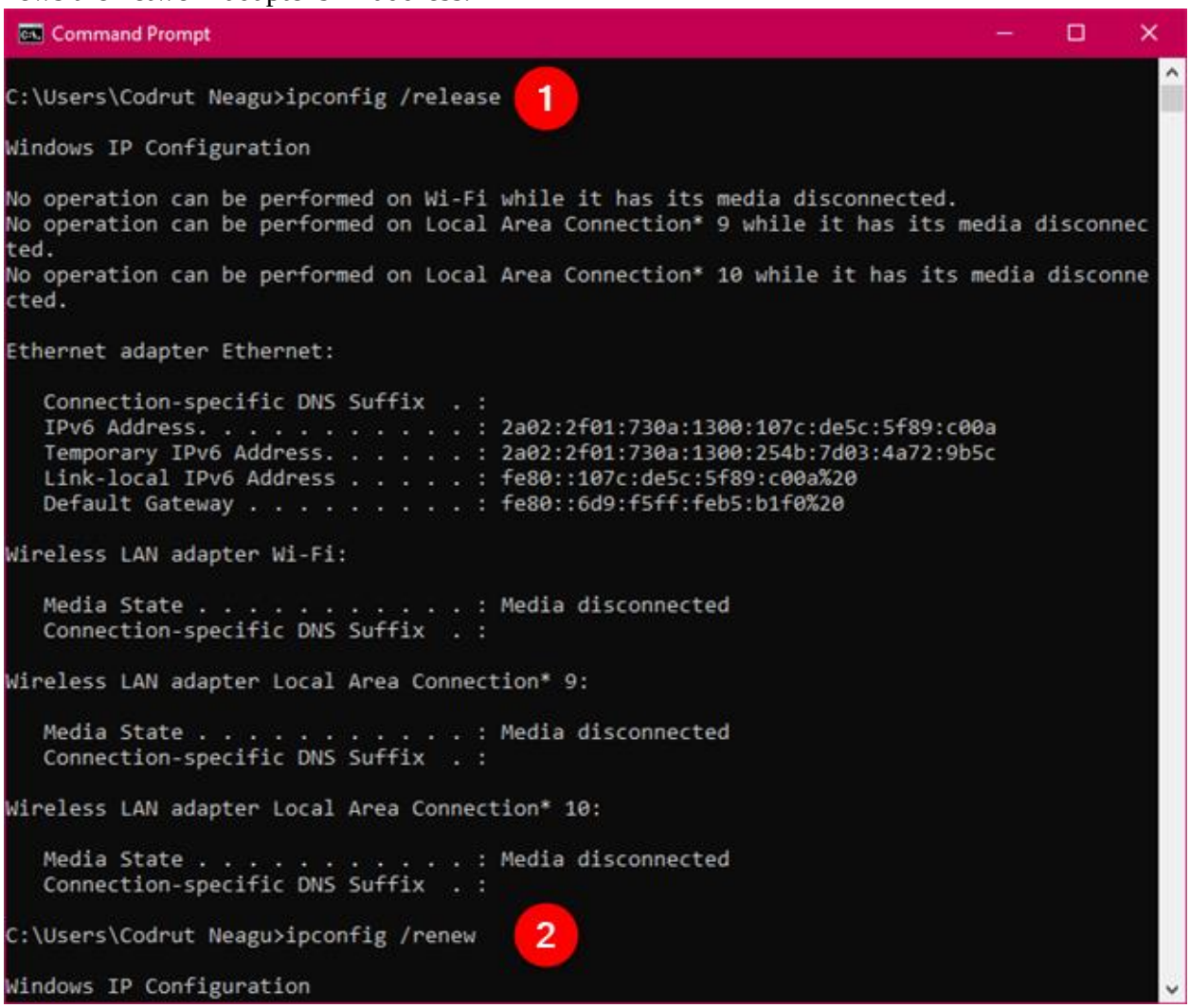
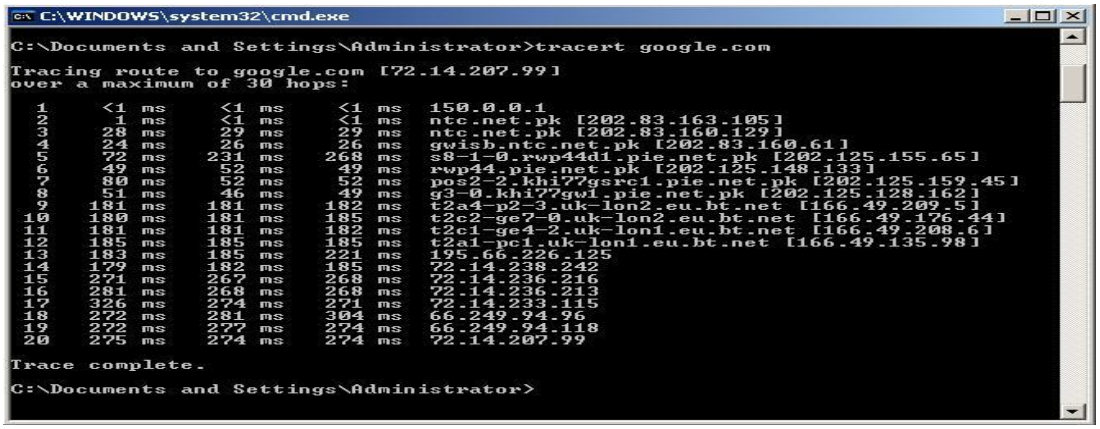


Figure: Running ipconfig /release and ipconfig /renew to reset the IP address

3 TRACERT Command

Tracert command tells you the path a packet takes from your computer to the destination. It will list all the routers from which a packet passes until it reaches its destination.
C:\tracert google.com



4. NSLOOKUP Command

Displays the default DNS server information.

Type the following command

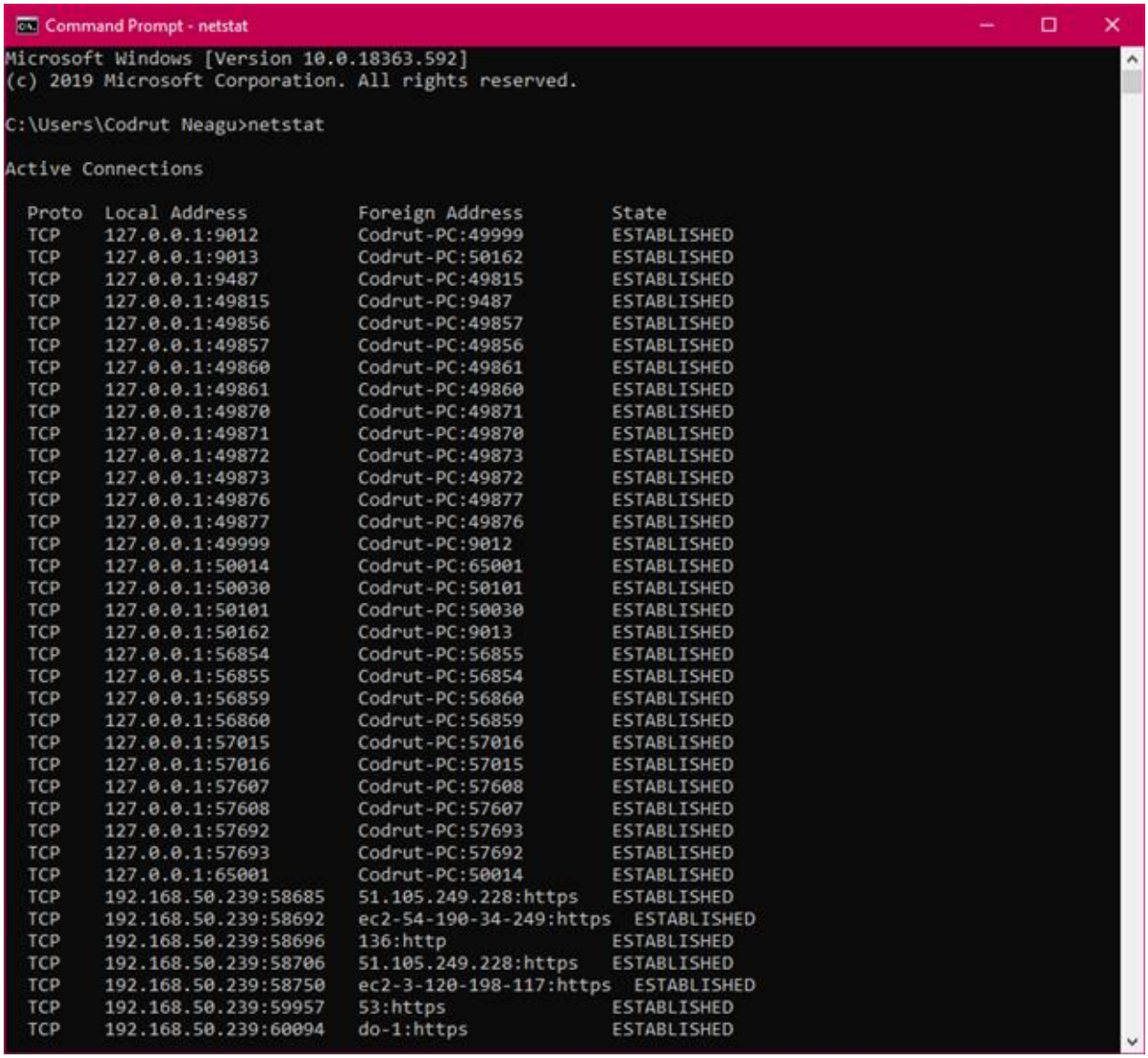
C:\>nslookup

What is your default DNS server's IP address?

5 . NETSTAT Command

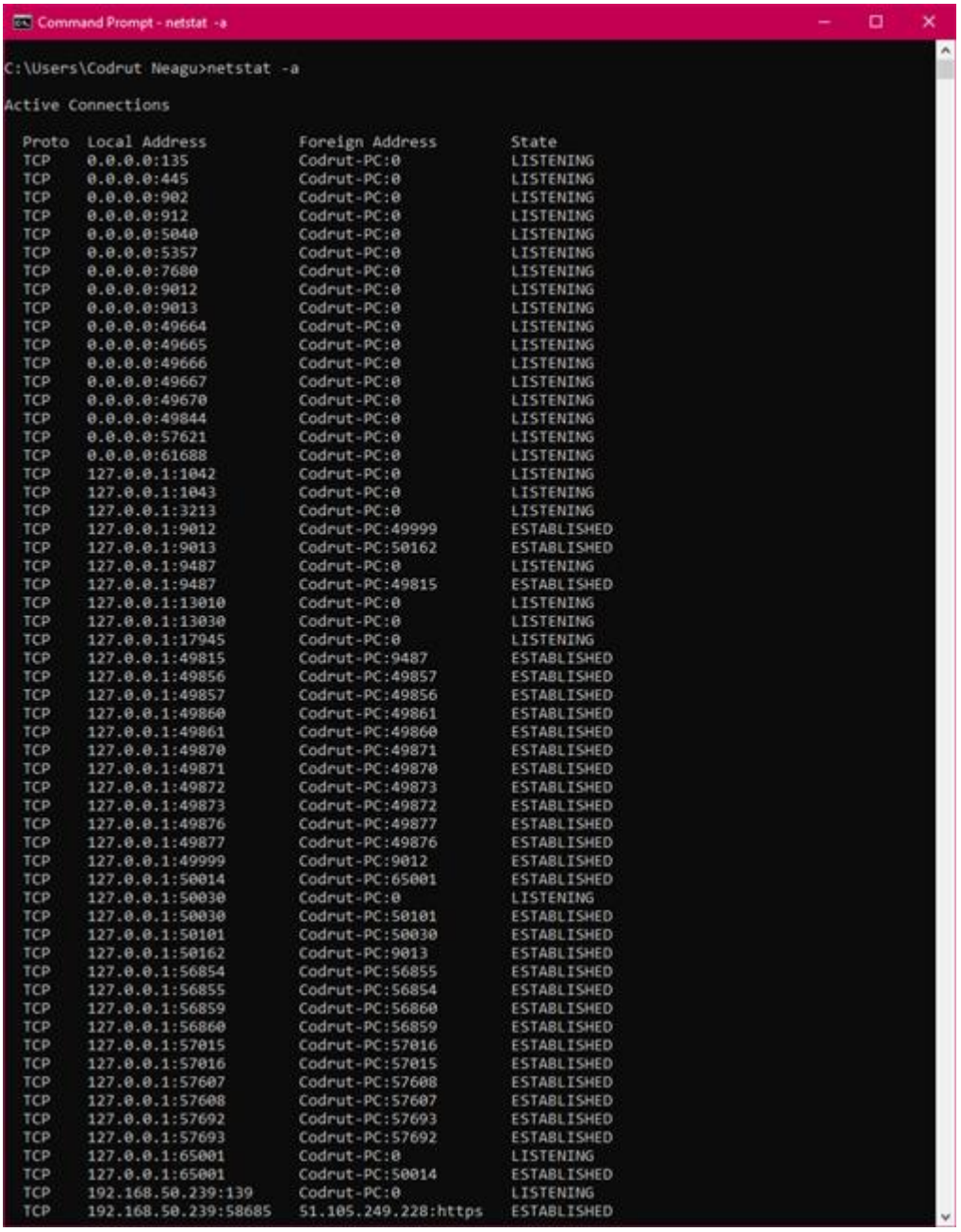
You can get other useful cmd nic info from the netstat command, which lets you see the network connections that are active between your system and any other systems on your network or the internet.

Displays active TCP and UDP connections.
Practice the following commands
C:\>netstat
C:\>netstat -a
C:\>netstat –an



Netstat shows the active network connections and open ports

If you add the -a parameter to the netstat command, you can get a list with all the connections and listening ports, as seen in the image below.

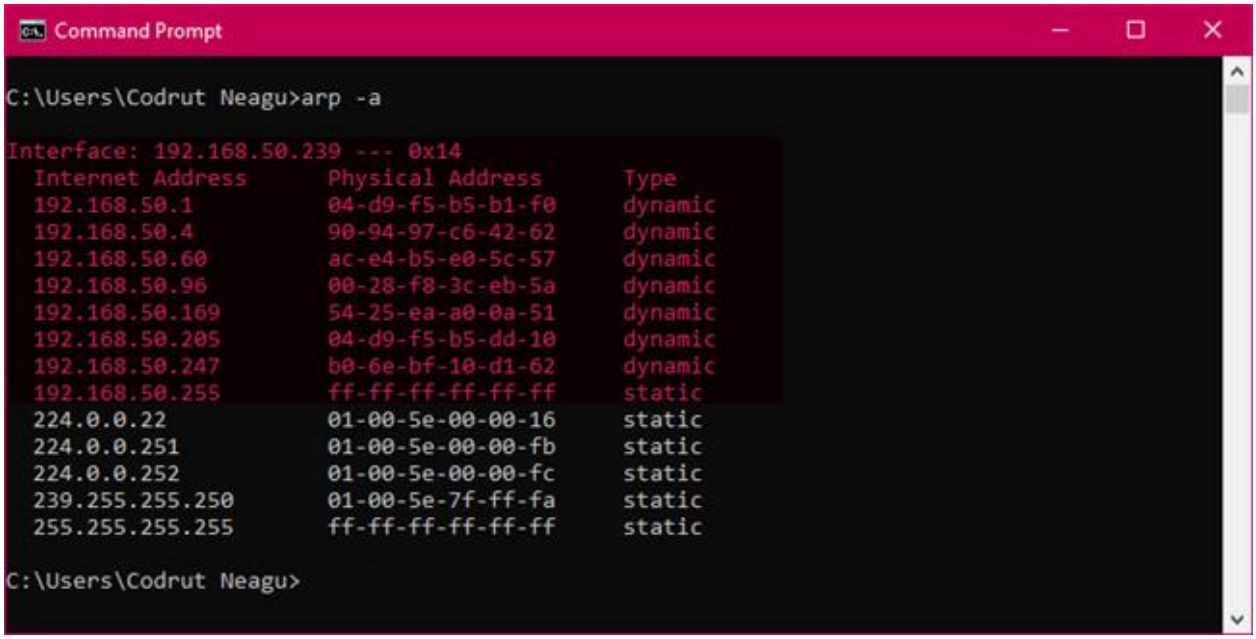


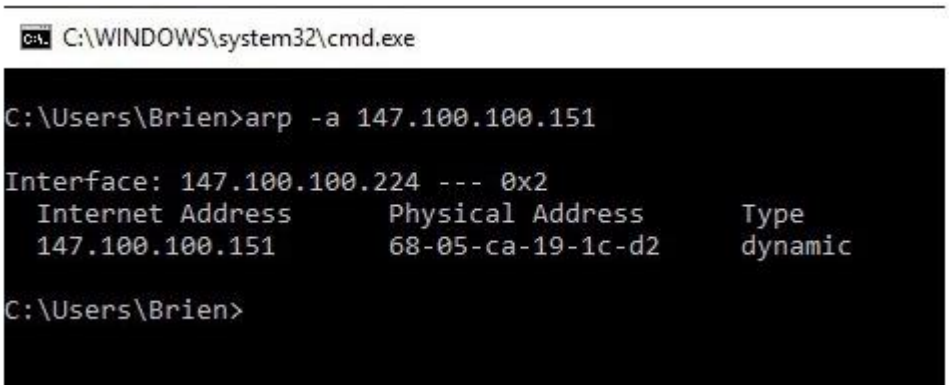
Netstat -a displays the active network connections, open ports and listening ports

6. ARP Command

ARP command corresponds to the Address Resolution Protocol, it is easy to understand of network communications in terms of IP addressing, packet delivery is ultimately dependent on the Media Access Control (MAC) address of the device's network adapter. This is where the Address Resolution Protocol comes into play. Its job is to map IP addresses to MAC addresses.

Windows devices maintain an ARP cache, which contains the results of recent ARP queries. It shows the contents of this cache by using the ARP -A command. If any problems in communicating with one specific host, you can append the remote host's IP address to the ARP -A command.



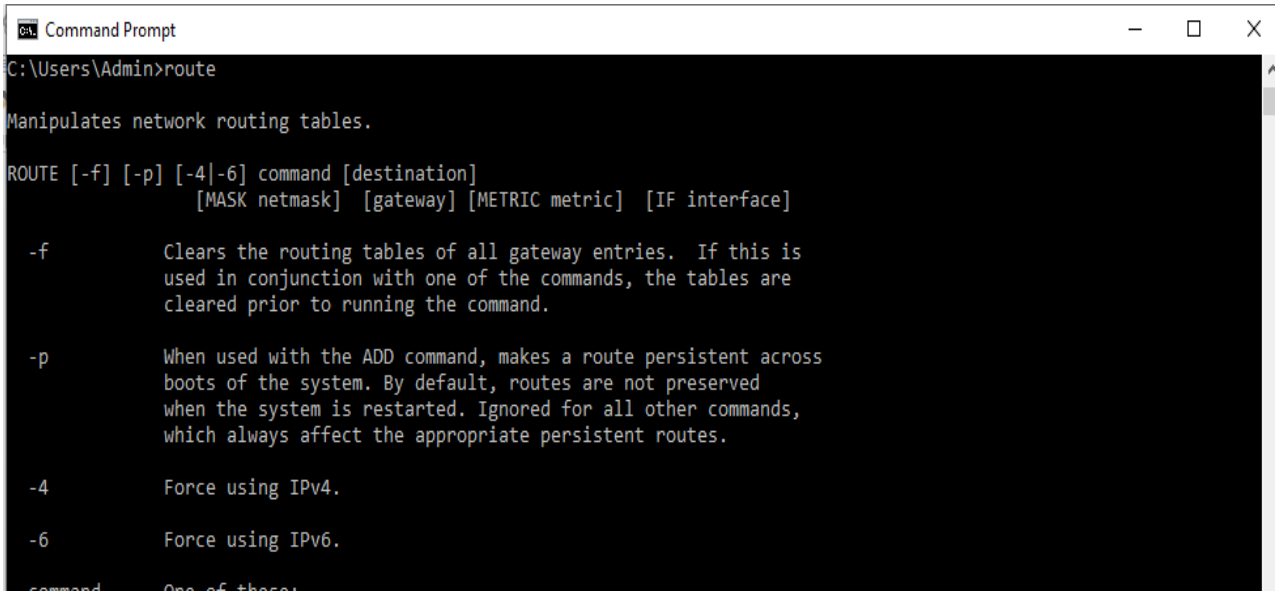


7.NbtStat-n Command

The NbtStat -n command for example, shows the NetBIOS names that are in use by a device. The NbtStat -r command shows how many NetBIOS names the device has been able to resolve recently.

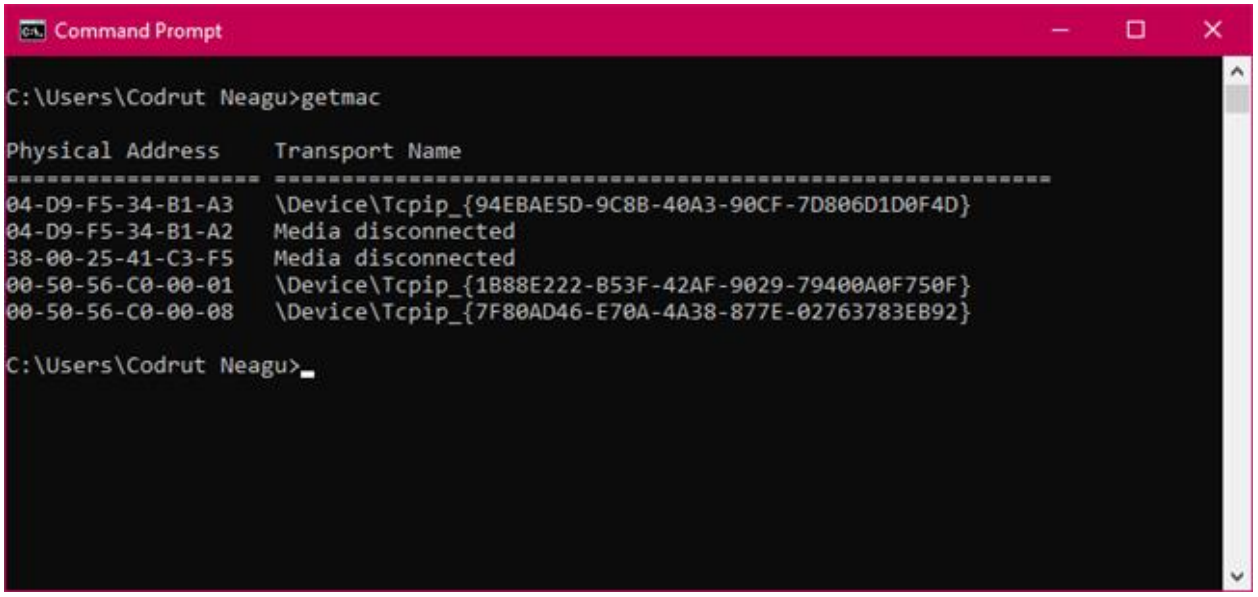
8.Route Command

IP networks use routing tables to direct packets from one subnet to another. The Windows Route utility allows you to view the device's routing tables. The Route command is that it not only shows you the routing table, it lets you make changes. Commands such as Route Add, Route Delete, and Route Change allow you to make routing table modifications on an as needed basis.



9. GETMAC Command

Getmac is a Windows command used to display the Media Access Control (MAC) addresses for each network adapter in the computer. One of the fastest and easiest ways to obtain the MAC addresses of your network adapters is to use the getmac command. In Command Prompt, type getmac and press Enter, as seen in the image below.



10.SYSTEMINFO Command: System Information

If you need to know what brand of network card you have, processor details, or the exact version of your Windows OS, the SYSTEMINFO command can help.This command polls your system and pulls the most important information about your system. It lists the information in a clean format that's easy to read.

Module-2

Cisco Packet Tracer tool

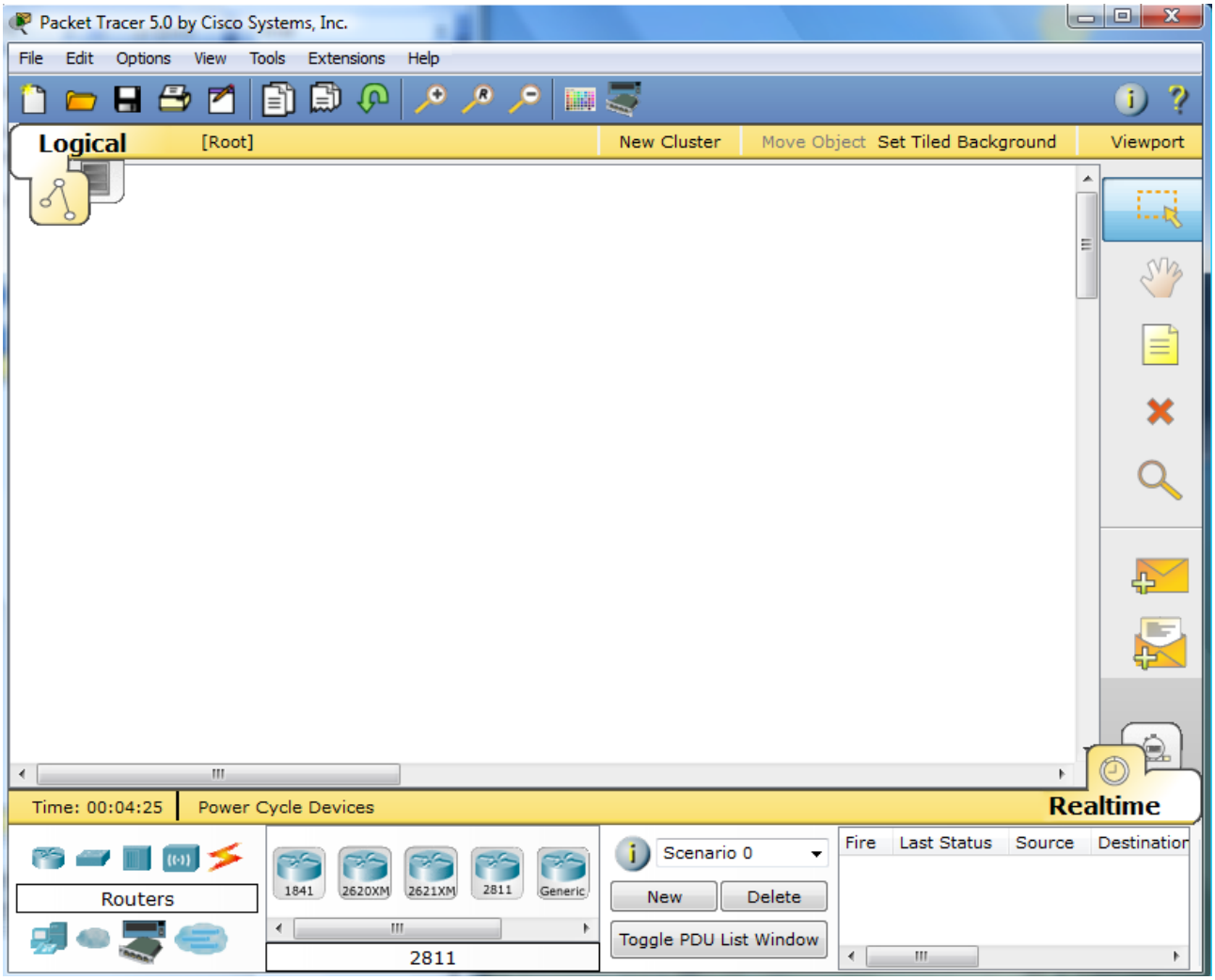
Packet Tracer – Creating a New Topology

What is Packet Tracer? Packet Tracer is a protocol simulator developed by Dennis Frezzo and his team at Cisco Systems. Packet Tracer (PT) is a powerful and dynamic tool that displays the various protocols used in networking, in either Real Time or Simulation mode. This includes layer 2 protocols such as Ethernet and PPP, layer 3 protocols such as IP, ICMP, and ARP, and layer 4 protocols such as TCP and UDP. Routing protocols can also be traced.

Purpose: The purpose of this lab is to become familiar with building topologies in Packet Tracer.

Version: This lab is based on Packet Tracer 5.0, 7.3.0

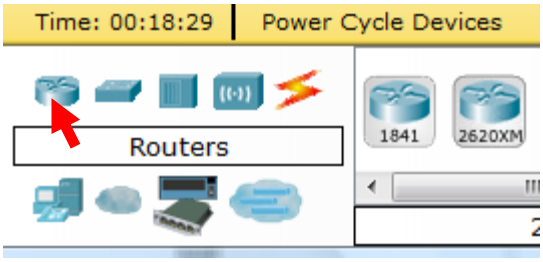
Step 1: Start Packet Tracer

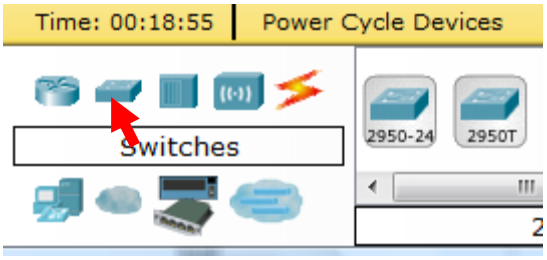


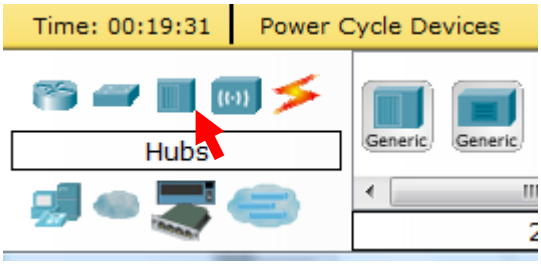
Step 2: Choosing Devices and Connections

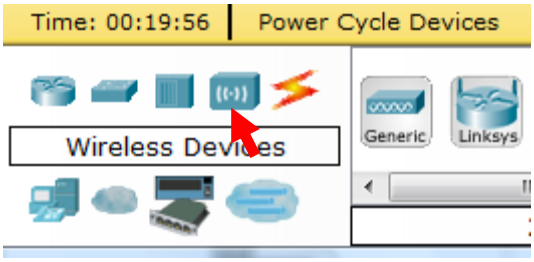
We will begin building our network topology by selecting devices and the media in which to connect them. Several types of devices and network connections can be used. For this lab we will keep it simple by using **End Devices**, **Switches**, **Hubs**, and **Connections**.

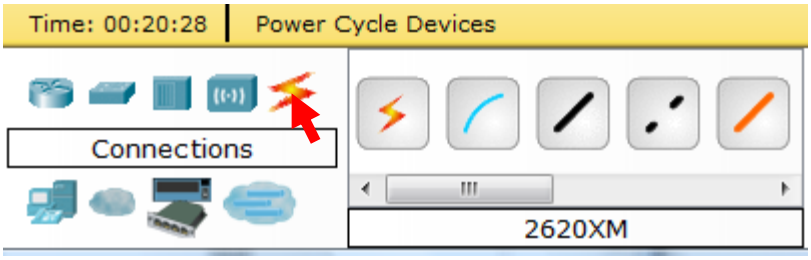
Single click on each group of devices and connections to display the various choices. The devices you see may differ slightly.

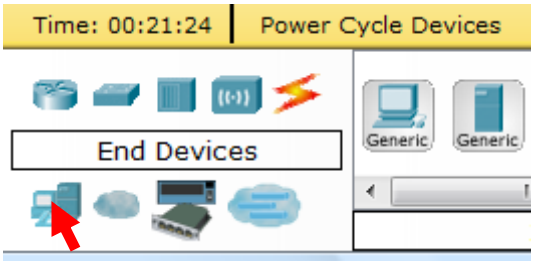


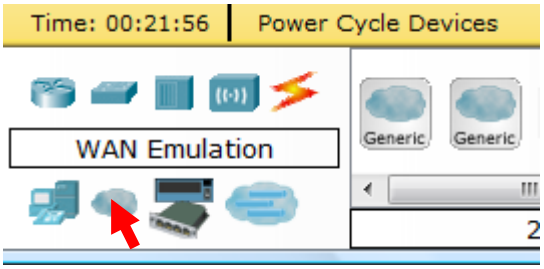


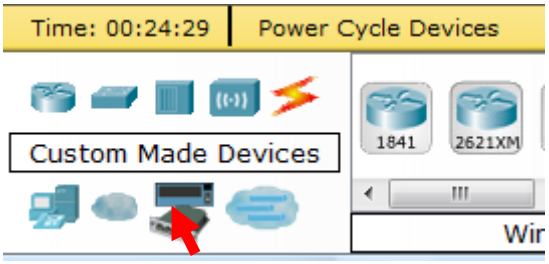


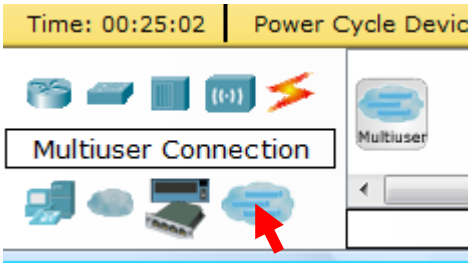






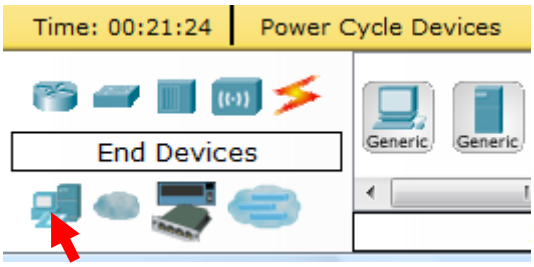




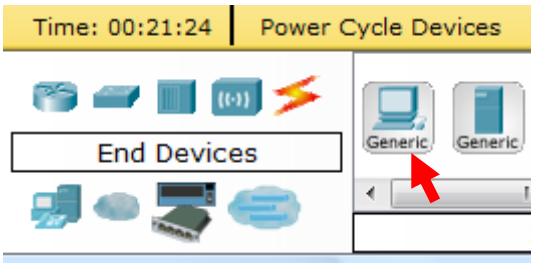


Step 3: Building the Topology – Adding Hosts

Single click on the **End Devices**.



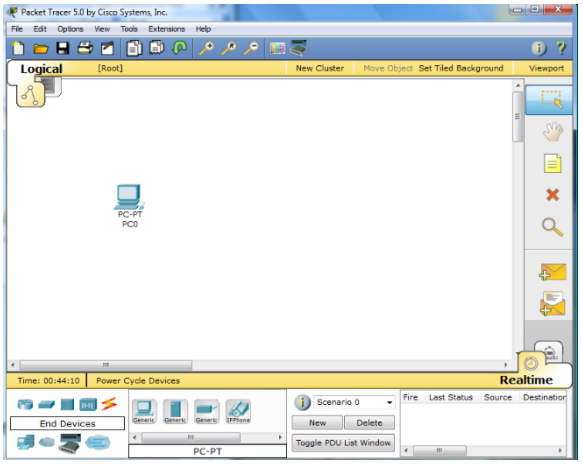
Single click on the **Generic** host.



Move the cursor into topology area. You will notice it turns into a plus "+" sign.

+

Single click in the topology area and it copies the device.

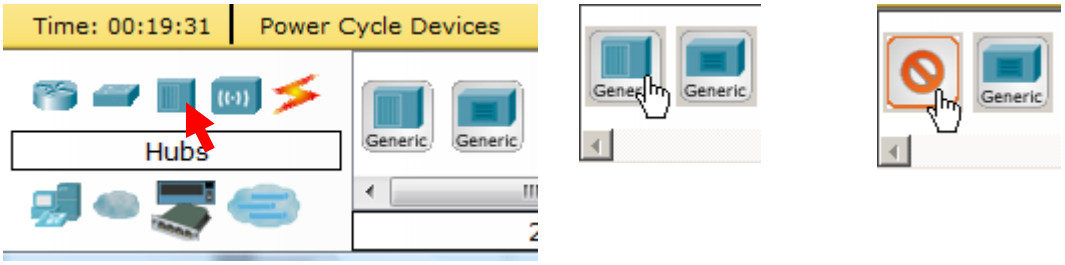


Add three more hosts.

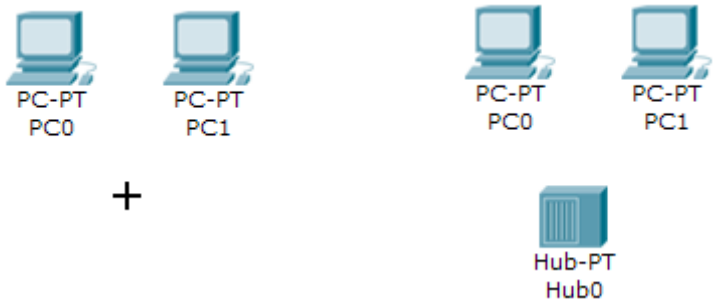
Step 4: Building the Topology – Connecting the Hosts to Hubs and Switches

Adding a Hub

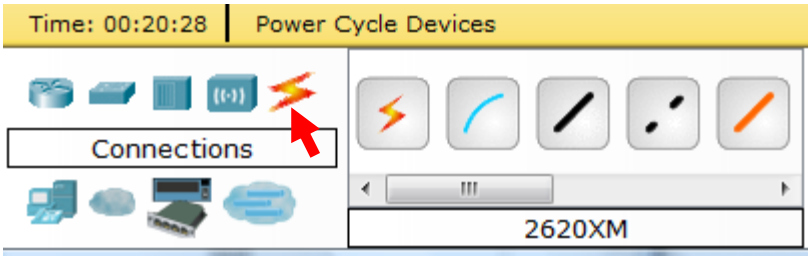
Select a hub, by clicking once on **Hubs** and once on a **Generic** hub.



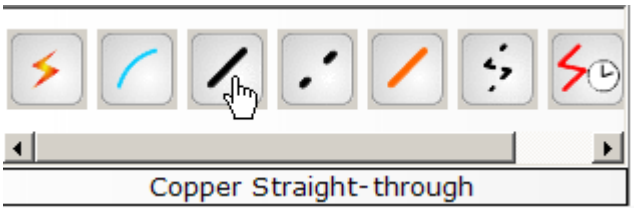
Add the hub by moving the plus sign "+" below PC0 and PC1 and click once.



Connect PC0 to Hub0 by first choosing **Connections.**



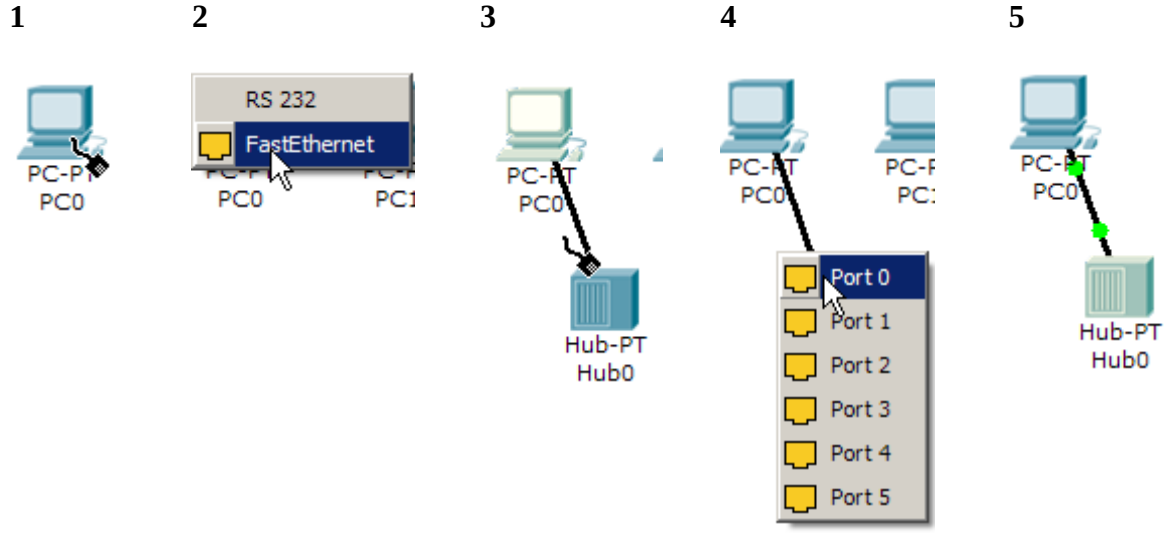
Click once on the **Copper Straight-through** cable.



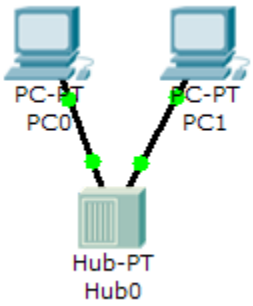
Perform the following steps to connect **PC0** to **Hub0**:

1. Click once on **PC0**
2. Choose **FastEthernet**
3. Drag the cursor to **Hub0**
4. Click once on **Hub0** and choose **Port 0**

5. Notice the green link lights on both the **PC0** Ethernet NIC and the **Hub0** Port 0 showing that the link is active.

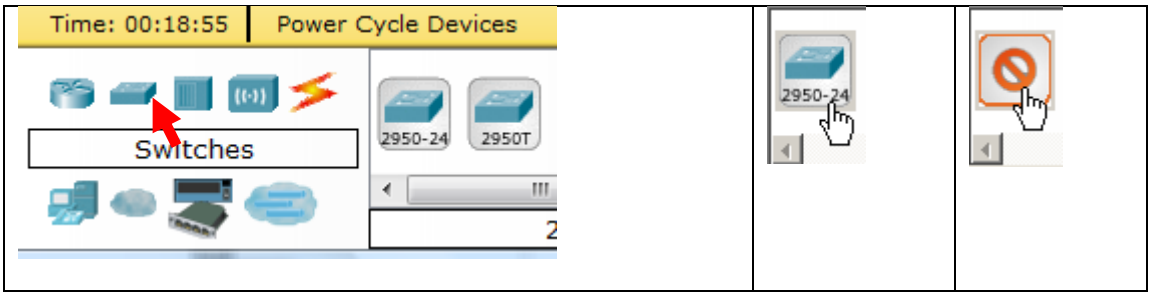


Repeat the steps above for **PC1** connecting it to **Port 1** on **Hub0**. (The actual hub port you choose does not matter.)



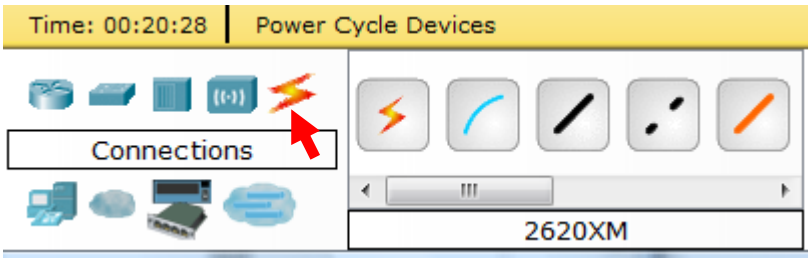
Adding a Switch

Select a switch, by clicking once on **Switches** and once on a **2950-24** switch.

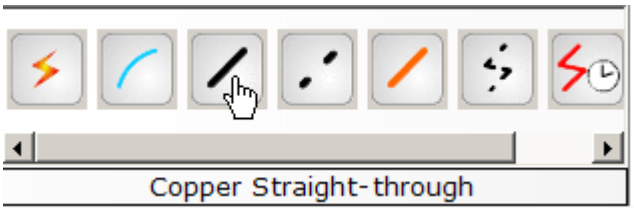


Add the switch by moving the plus sign "+" below PC2 and PC3 and click once.

Connect PC2 to Hub0 by first choosing **Connections.**



Click once on the **Copper Straight-through** cable.

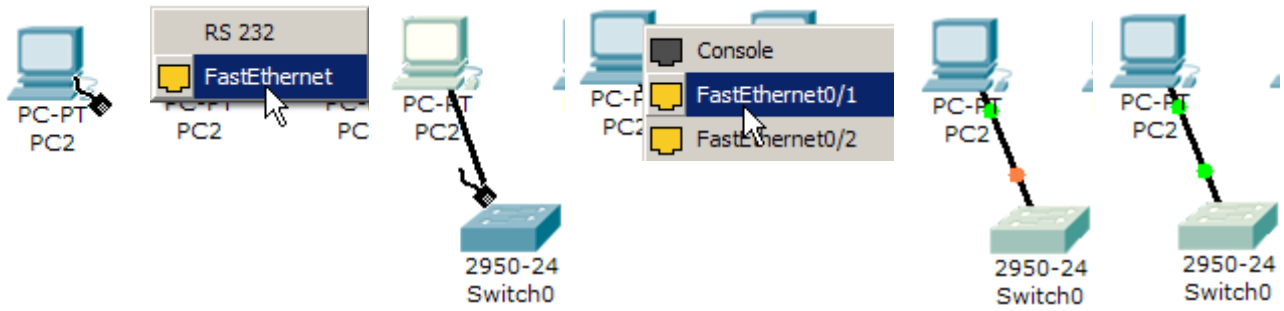


Perform the following steps to connect **PC2** to **Switch0**:

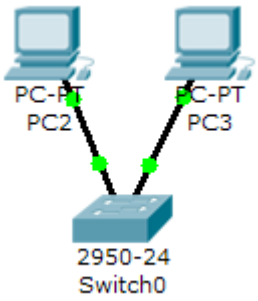
1. Click once on **PC2**
2. Choose **FastEthernet**
3. Drag the cursor to **Switch0**
4. Click once on **Switch0** and choose **FastEthernet0/1**
5. Notice the green link lights on **PC2** Ethernet NIC and amber light **Switch0 FastEthernet0/1 port**. The switch port is temporarily not forwarding frames, while it goes through the stages for the Spanning Tree Protocol (STP) process.
6. After a about 30 seconds the amber light will change to green indicating that the port has entered the forwarding stage. Frames can now forwarded out the switch port.

Note: Spanning Tree Protocol (STP) is discussed later.

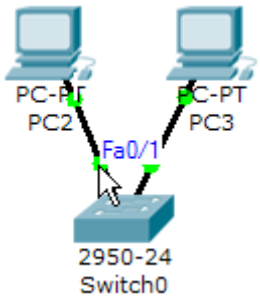
1 2 3 4 5 6



Repeat the steps above for **PC3** connecting it to **Port 3** on **Switch0** on port **FastEtherent0/2**. (The actual switch port you choose does not matter.)



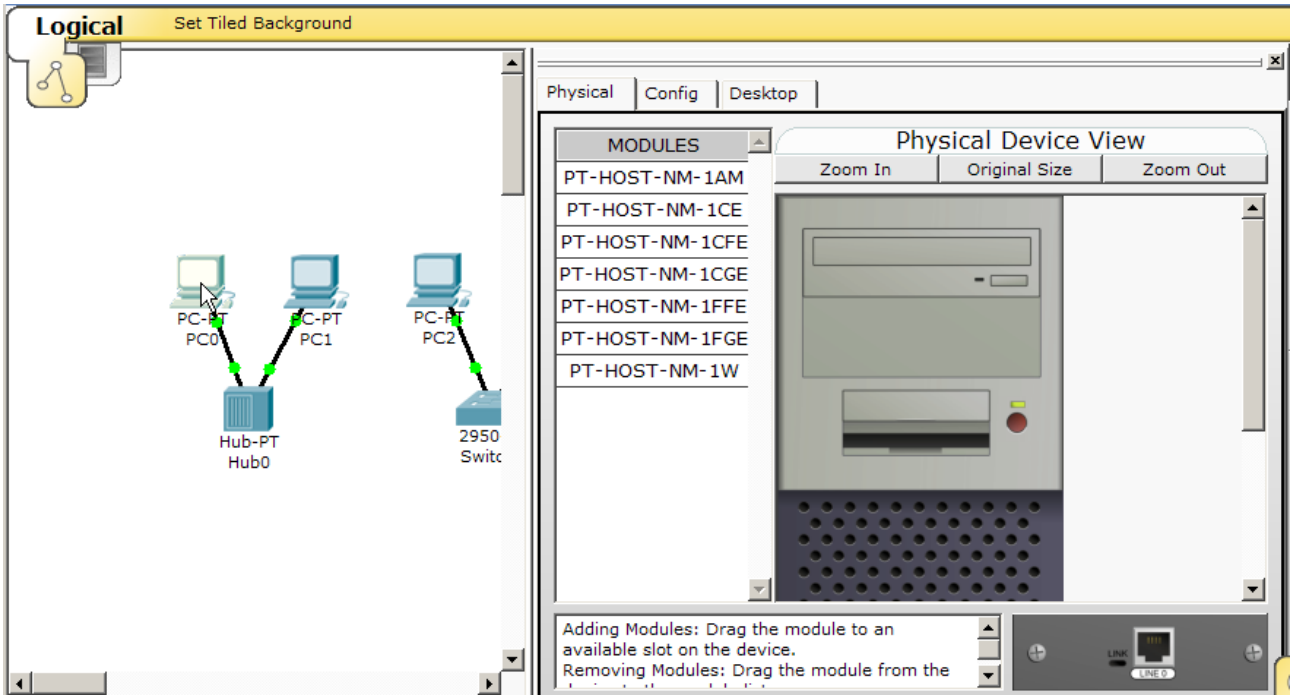
Move the cursor over the link light to view the port number. **Fa** means FastEthernet, 100 Mbps Ethernet.



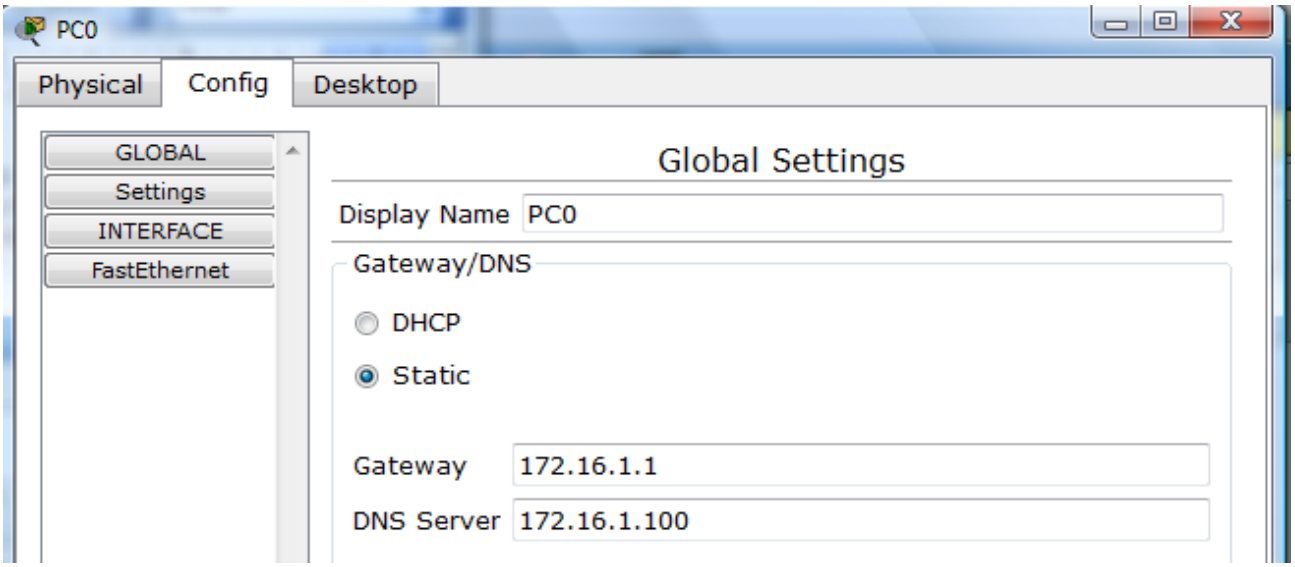
Step 5: Configuring IP Addresses and Subnet Masks on the Hosts

Before we can communicate between the hosts we need to configure IP Addresses and Subnet Masks on the devices.

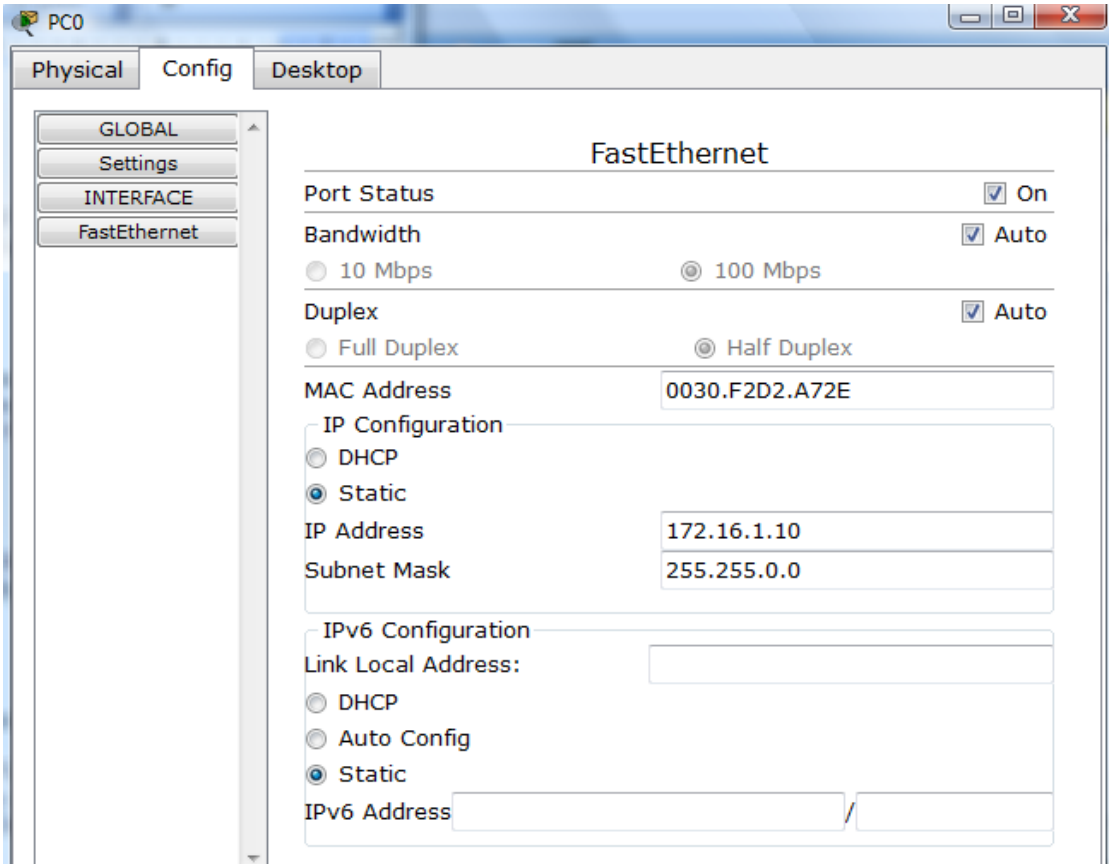
Click once on PC0.



Choose the **Config** tab and click on **Settings**. It is here that you can change the name of PC0. It is also here where you would enter a **Gateway** IP Address, also known as the default gateway and the **DNS Server** IP Address. We will discuss this later, but this would be the IP address of the local router. If you want, you can enter the Gateway IP Address 172.16.1.1 and DNS Server IP Address 172.16.1.100, although it will not be used in this lab.



Click on **Interface** and then **FastEthernet**. Although we have not yet discussed IP Addresses, add the IP Address to 172.16.1.10. Click once in the Subnet Mask field to enter the default Subnet Mask. You can leave this at 255.255.0.0. We will discuss this later.



Also, notice this is where you can change the Bandwidth (speed) and Duplex of the Ethernet NIC (Network Interface Card). The default is Auto (autonegotiation), which means the NIC will negotiate with the hub or switch. The bandwidth and/or duplex can be manually set by removing the check from the **Auto** box and choosing the specific option.

Bandwidth - Auto

If the host is connected to a hub or switch port which can do 100 Mbps, then the Ethernet NIC on the host will choose 100 Mbps (Fast Ethernet). Otherwise, if the hub or switch port can only do 10 Mbps, then the Ethernet NIC on the host will choose 10 Mbps (Ethernet).

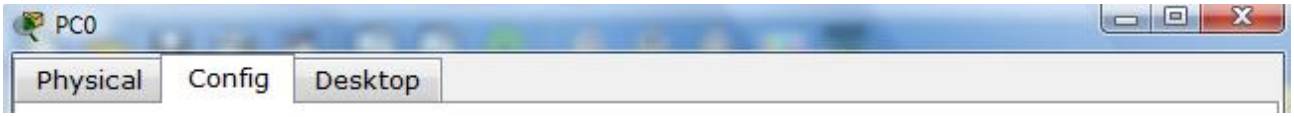
Duplex - Auto

Hub: If the host is connected to a hub, then the Ethernet NIC on the host will choose Half Duplex.

Switch: If the host is connected to a switch, and the switch port is configured as Full Duplex (or Autonegotiation), then the Ethernet NIC on the host will choose Full Duplex. If the switch port is configured as Half Duplex, then the Ethernet NIC on the host will choose Half Duplex. (Full Duplex is a much more efficient option.)

The information is automatically saved when entered.

To close this dialog box, click the "**X**" in the upper right.

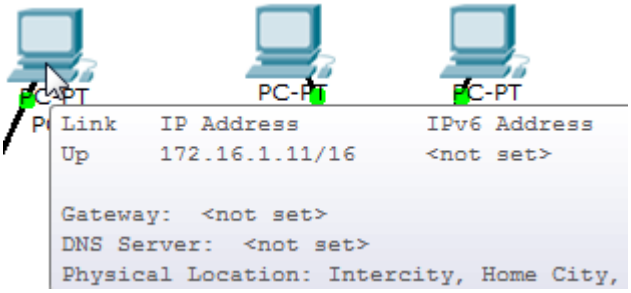


Repeat these steps for the other hosts. Use the information below for IP Addresses and Subnet Masks.

Host	**IP Address**	**Subnet Mask**
PC0	172.16.1.10	255.255.0.0
PC1	172.16.1.11	255.255.0.0
PC2	172.16.1.12	255.255.0.0
PC3	172.16.1.13	255.255.0.0

Verify the information

To verify the information that you entered, move the Select tool (arrow) over each host.

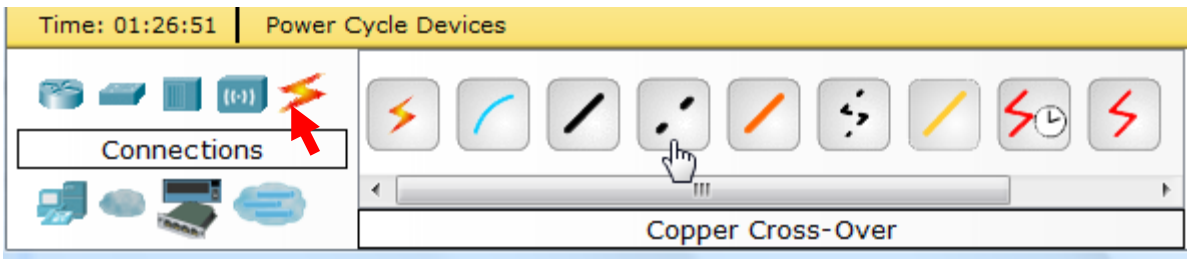


Deleting a Device or Link

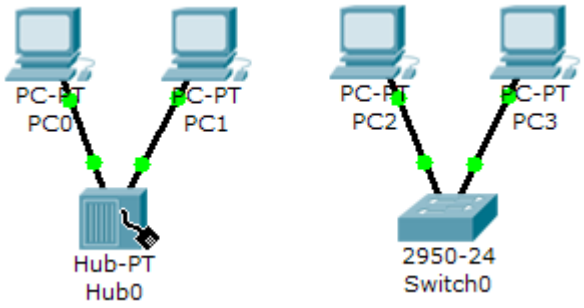
To delete a device or link, choose the **Delete** tool and click on the item you wish to delete.

Step 6: Connecting Hub0 to Switch0

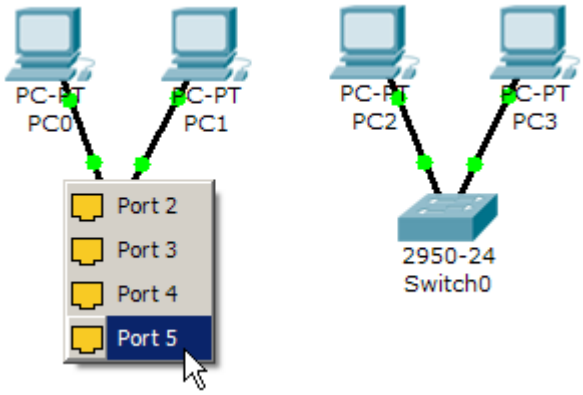
To connect like-devices, like a Hub and a Switch, we will use a Cross-over cable. Click once the **Cross-over** Cable from the **Connections** options.



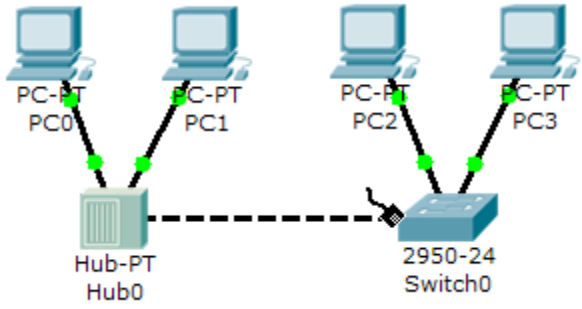
Move the Connections cursor over **Hub0** and click once.



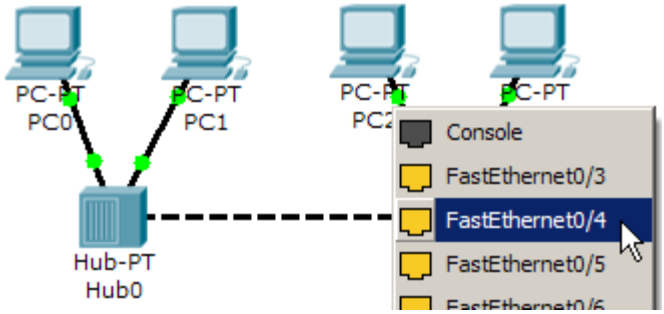
Select **Port 5** (actual port does not matter).



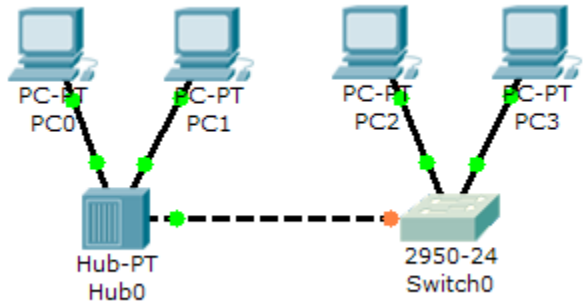
Move the Connections cursor to **Switch0**.

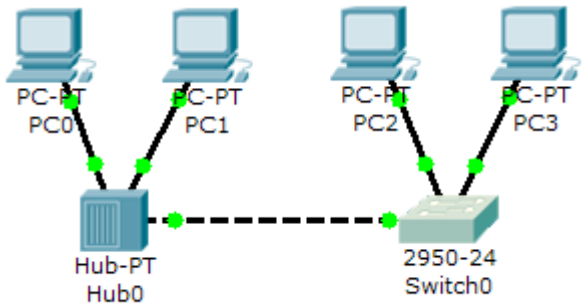


Click once on **Switch0** and choose **FastEthernet0/4** (actual port does not matter).



The link light for switch port **FastEthernet0/4** will begin as amber and eventually change to green as the Spanning Tree Protocol transitions the port to forwarding.



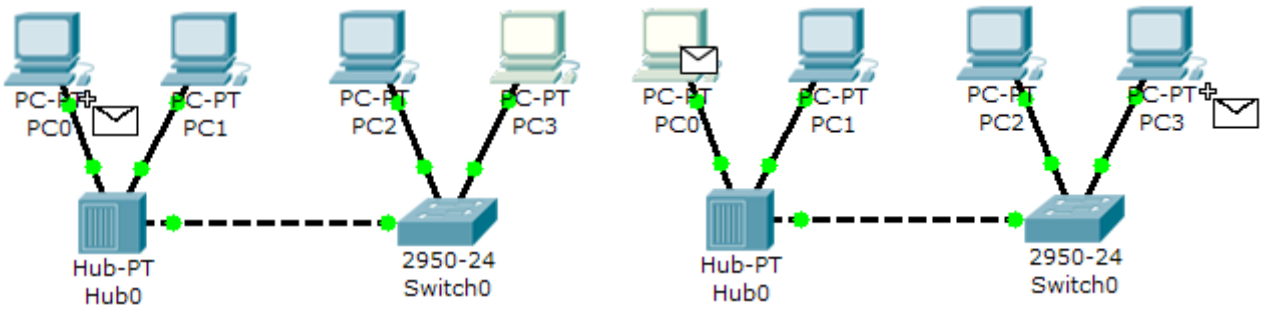


Step 7: Verifying Connectivity in Realtime Mode

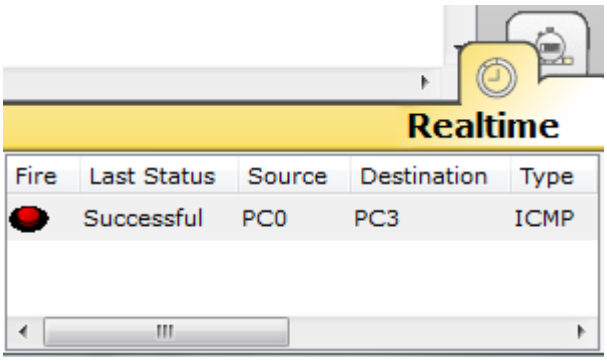
Be sure you are in **Realtime** mode.

Select the **Add Simple PDU** tool used to ping devices..

Click once on PC0, then once on PC3.



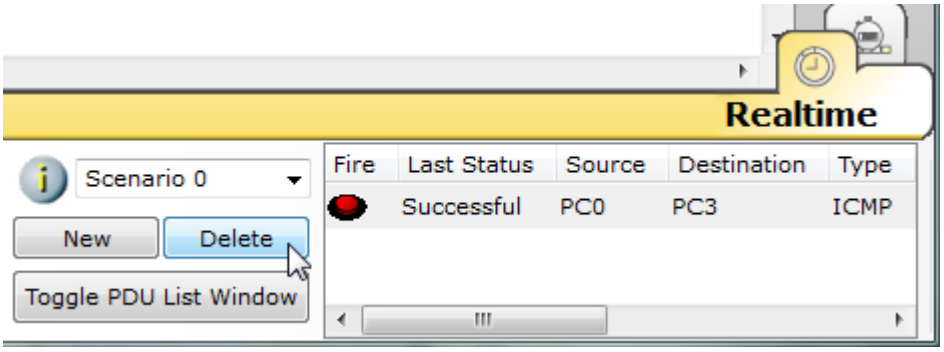
The PDU **Last Status** should show as **Successful**.



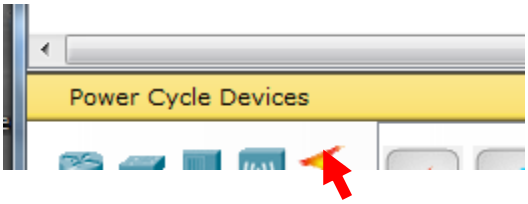
Resetting the Network

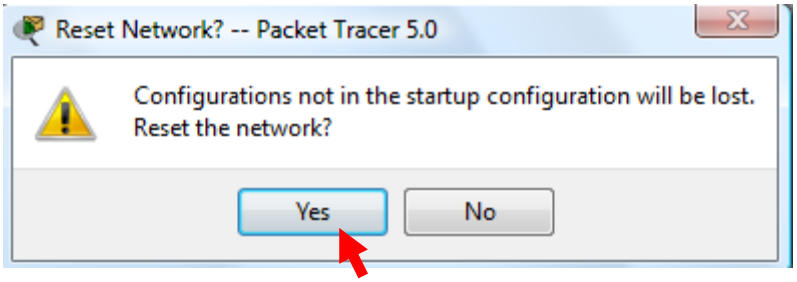
At this point we will want to reset the network, Whenever you want to reset the network and begin the simulation again, perform the following tasks:

Click **Delete** in the PDU area.



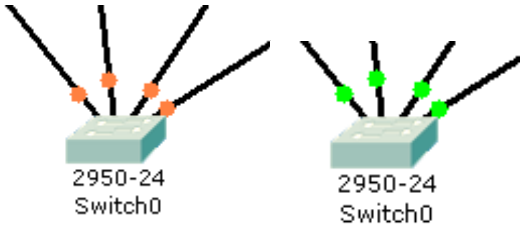
Now, Power Cycle Devices and confirm the action.





Waiting for Spanning Tree Protocol (STP)

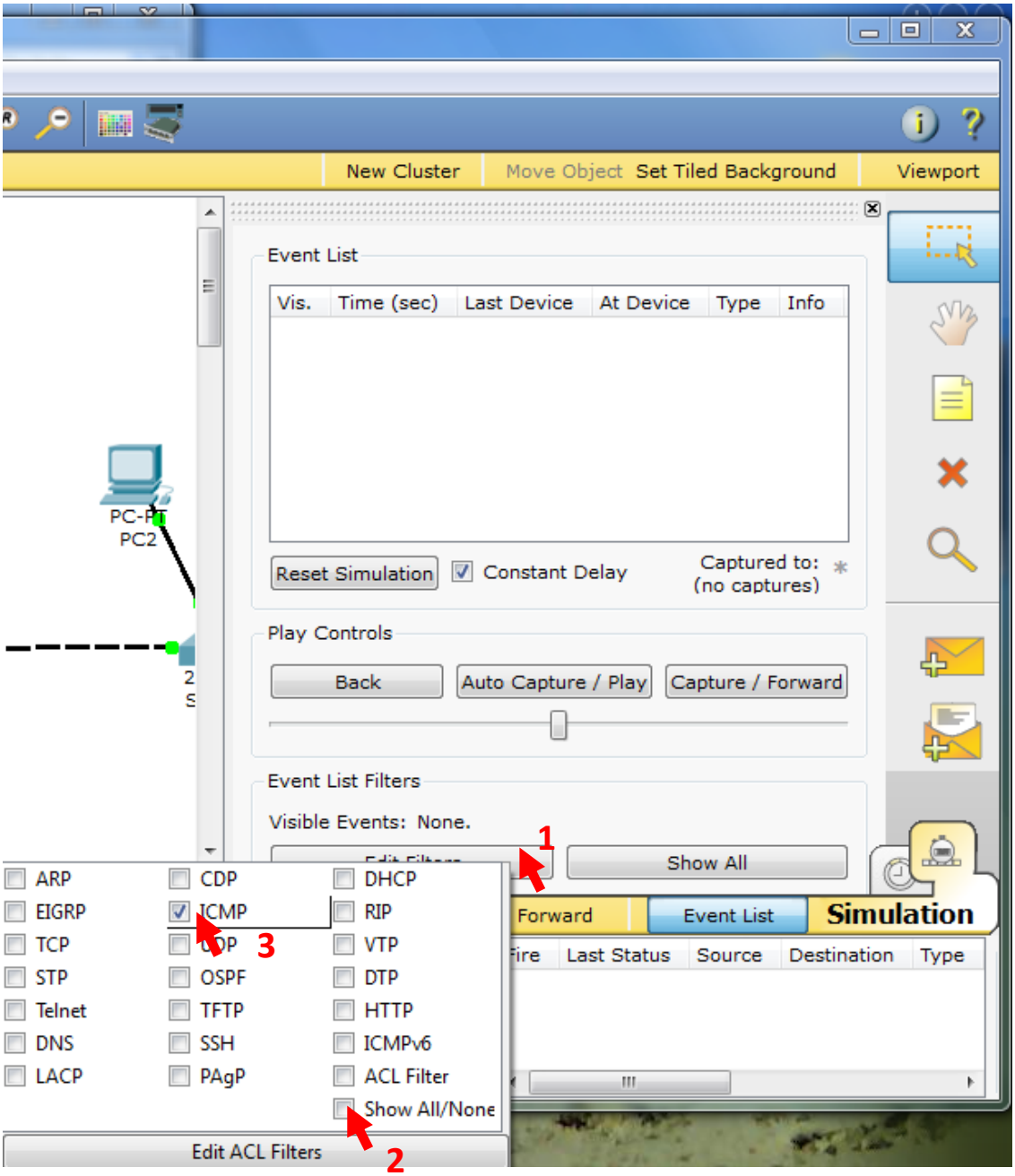
Note: Because Packet Tracer also simulates the Spanning Tree Protocol (later), at times the switch may show amber lights on its interfaces. You will need to wait for the lights to turn green on the switches before they will forward any Ethernet frames.



Step 8: Verifying Connectivity in Simulation Mode

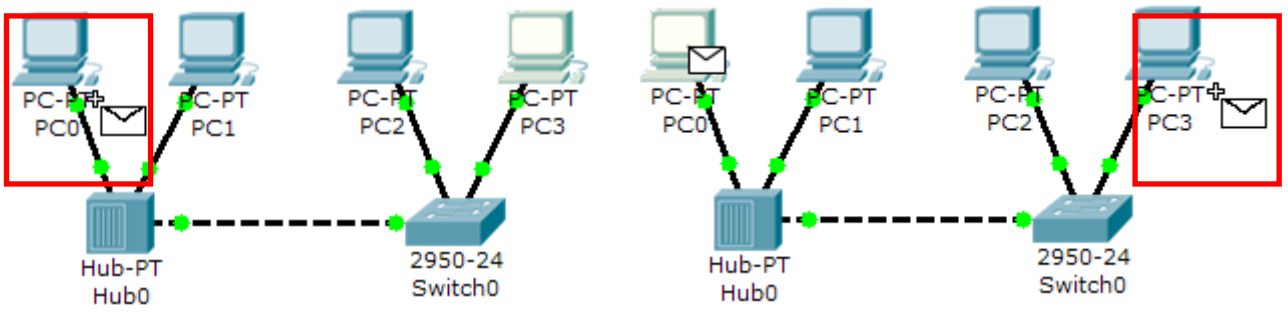
Be sure you are in **Simulation** mode.

Deselect all filters (All/None) and select only **ICMP**.

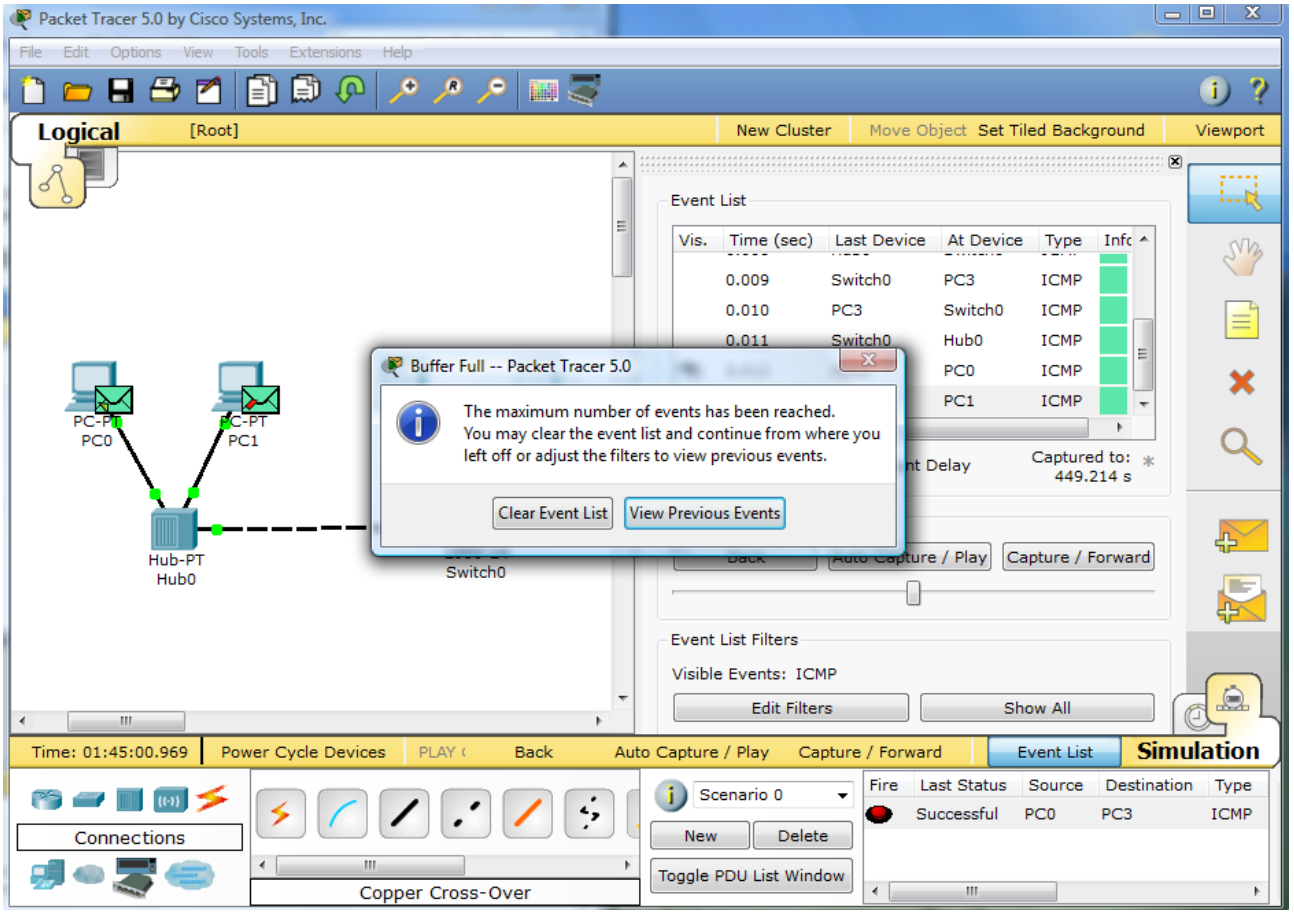


Select the **Add Simple PDU** tool used to ping devices..

Click once on PC0, then once on PC3.

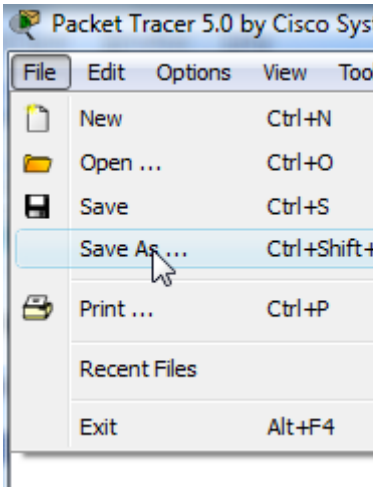


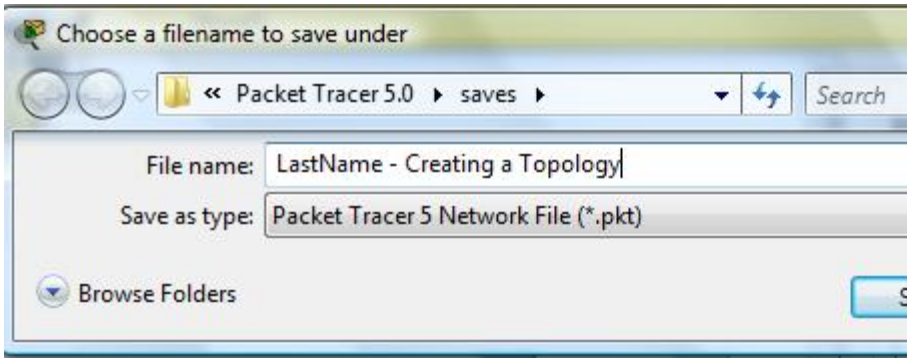
Continue clicking **Capture/Forward** button until the ICMP ping is completed. You should see the ICMP messages move between the hosts, hub and switch. The PDU **Last Status** should show as **Successful**. Click on **Clear Event List** if you do not want to look at the events or click **Preview Previous Events** if you do. For this exercise it does not matter.



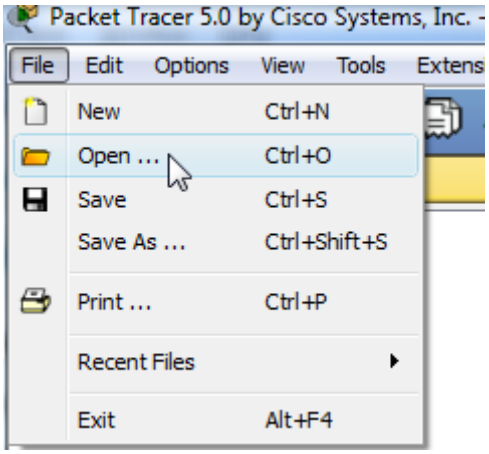
Step 9: Saving the Topology

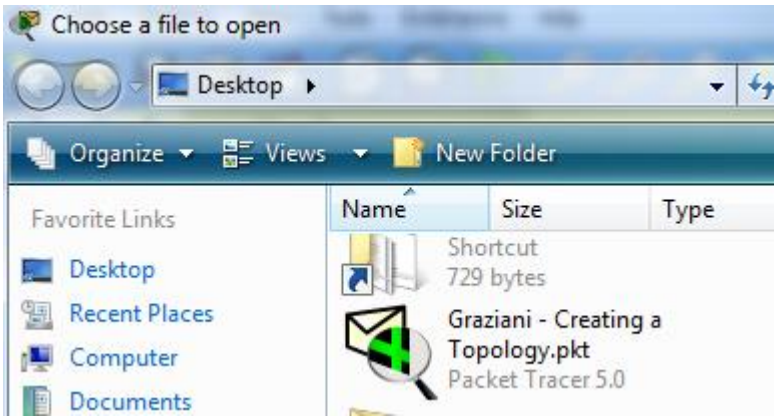
Perform the following steps to save the topology (uses .pkt file extension).



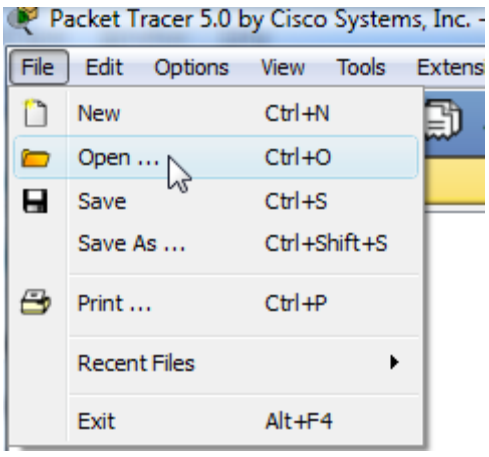


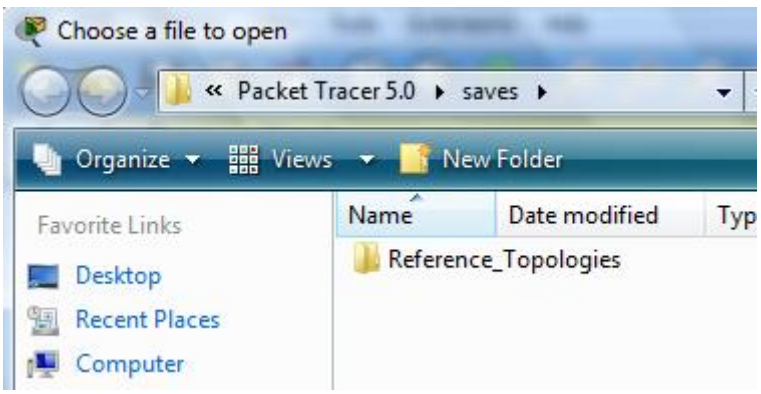
Opening Existing Topologies

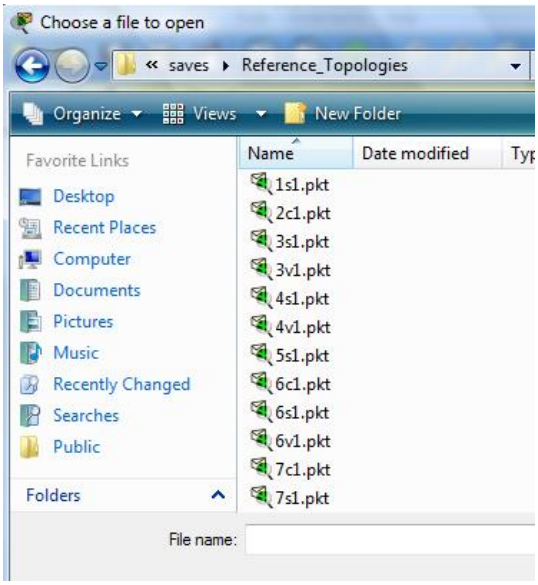




Opening Existing PT Topologies

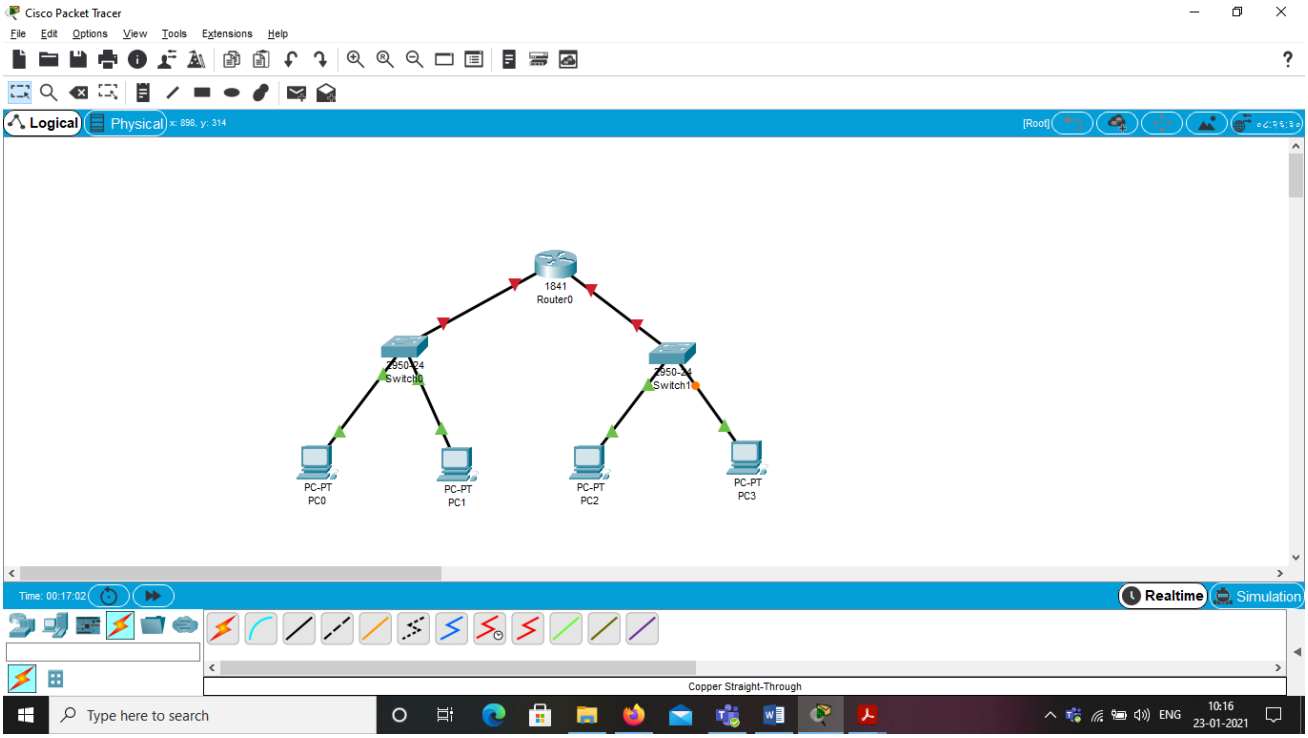




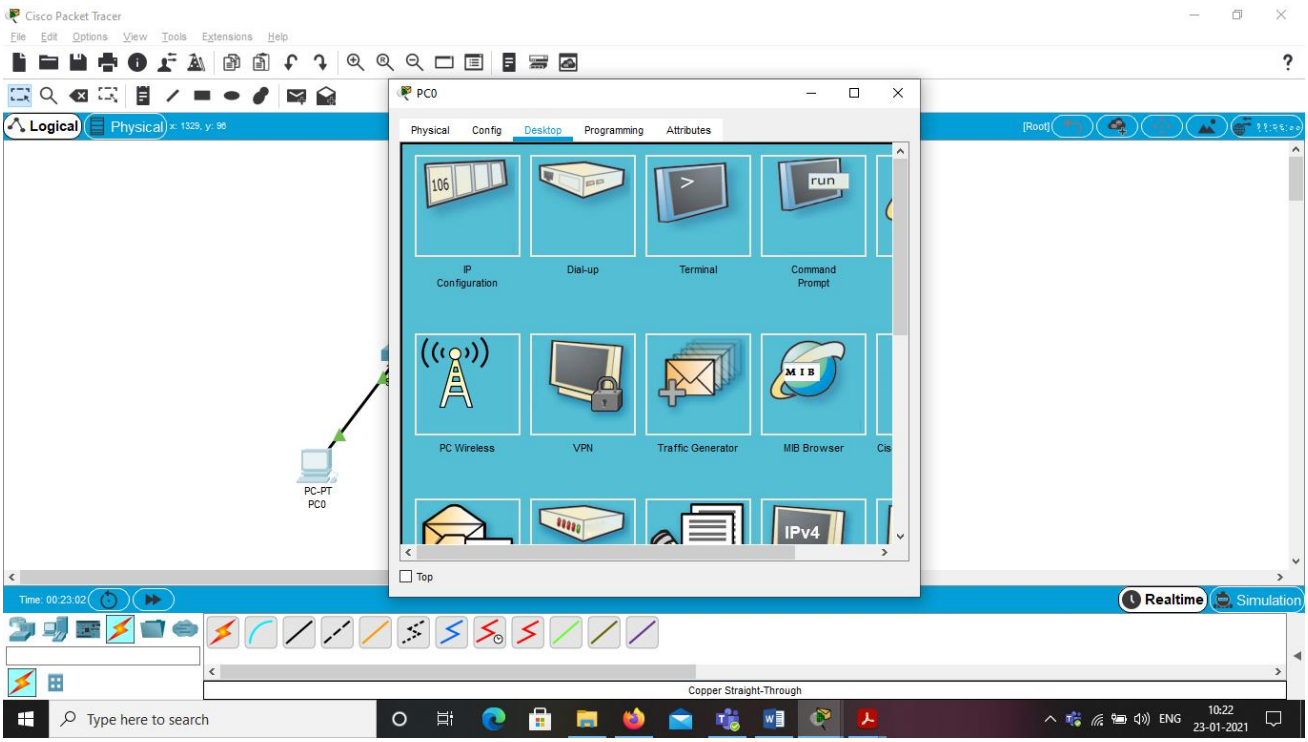

Choose a file to open
saves ▸ Reference_Topologies
Organize
Views
New Folder
Favorite Links
Desktop
Recent Places
Computer
Documents
Pictures
Music
Recently Changed
Searches
Public
Folders
Name
Date modified
1s1.pkt
2c1.pkt
3s1.pkt
3v1.pkt
4s1.pkt
4v1.pkt
5s1.pkt
6c1.pkt
6s1.pkt
6v1.pkt
7c1.pkt
7s1.pkt
File name:

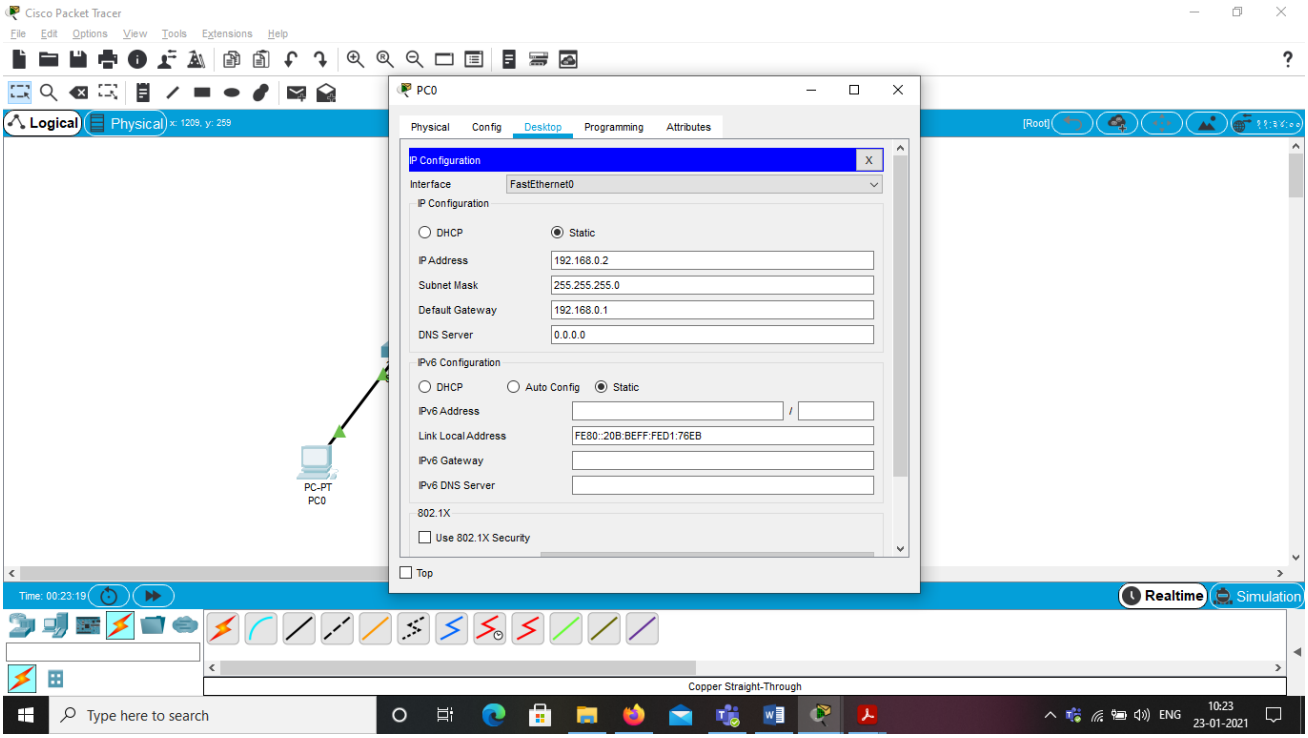
Exp 1: Configuration of Router using cisco packet tracer

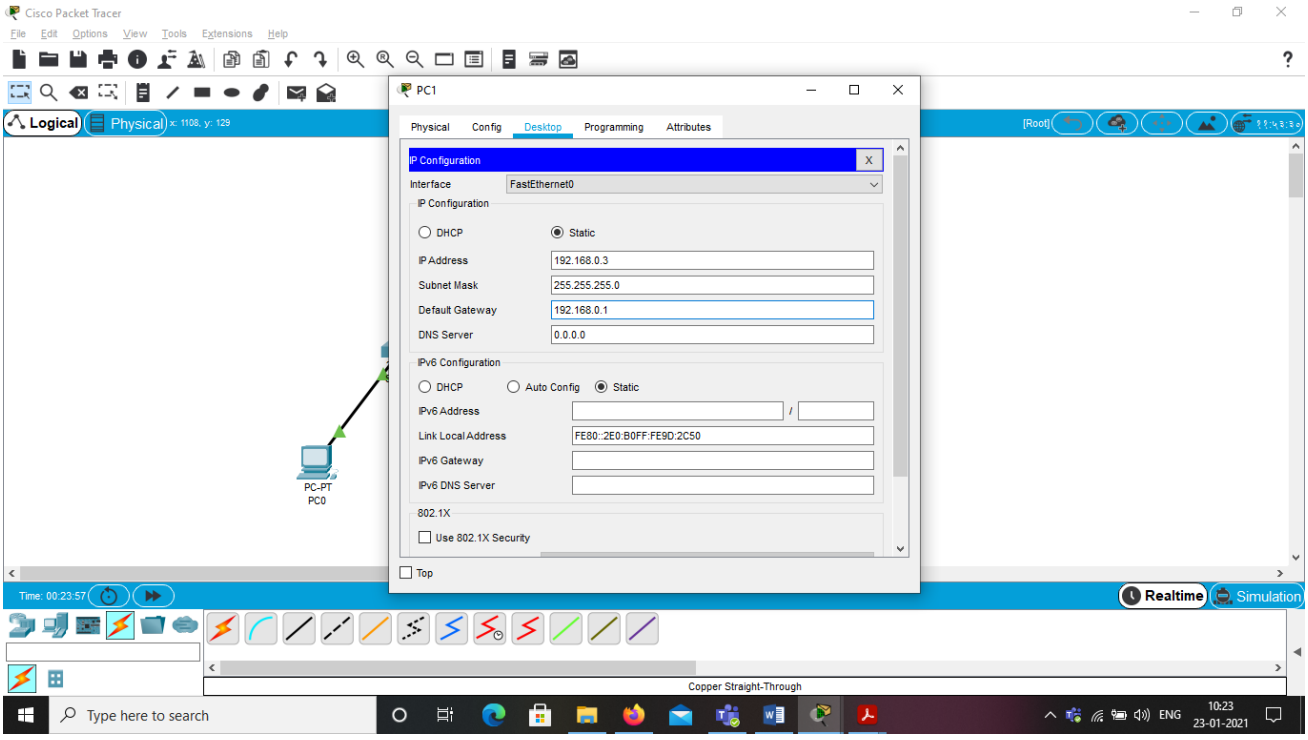
Step 1: Construct the topology



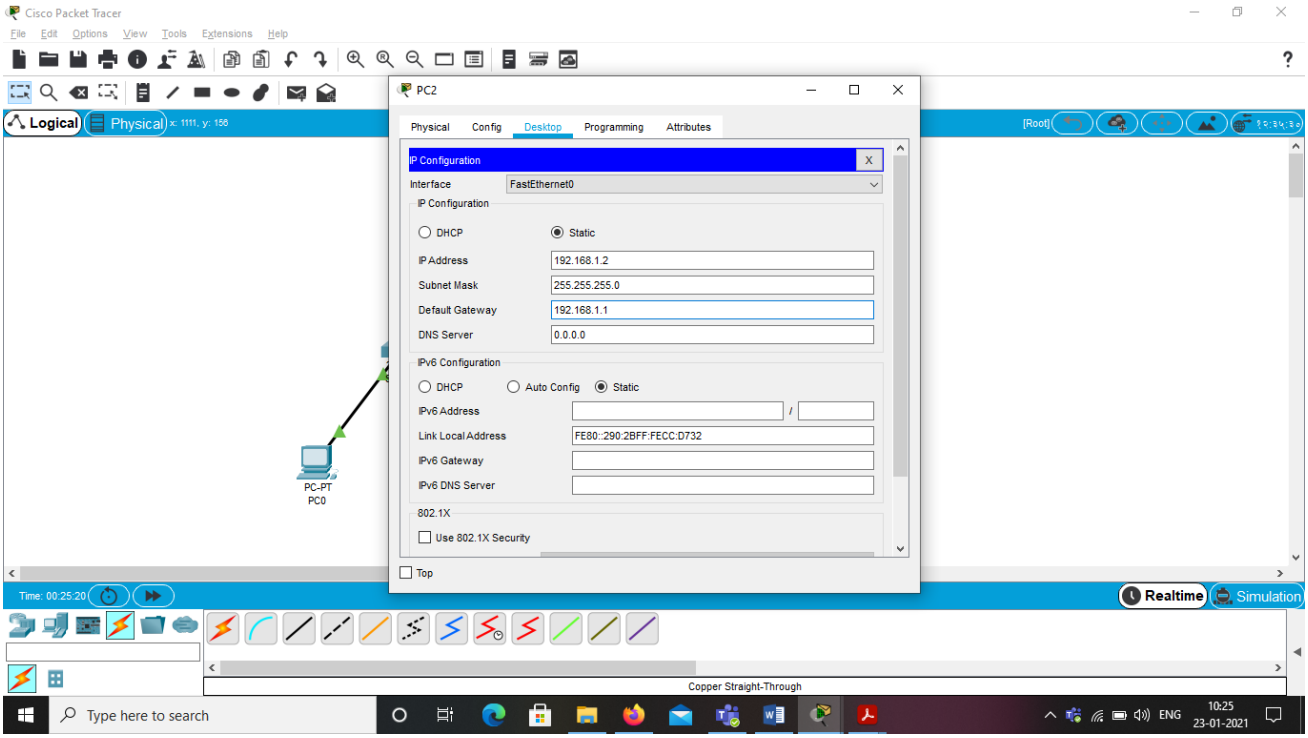
Step 2: Assign IP addresses to all PC's.

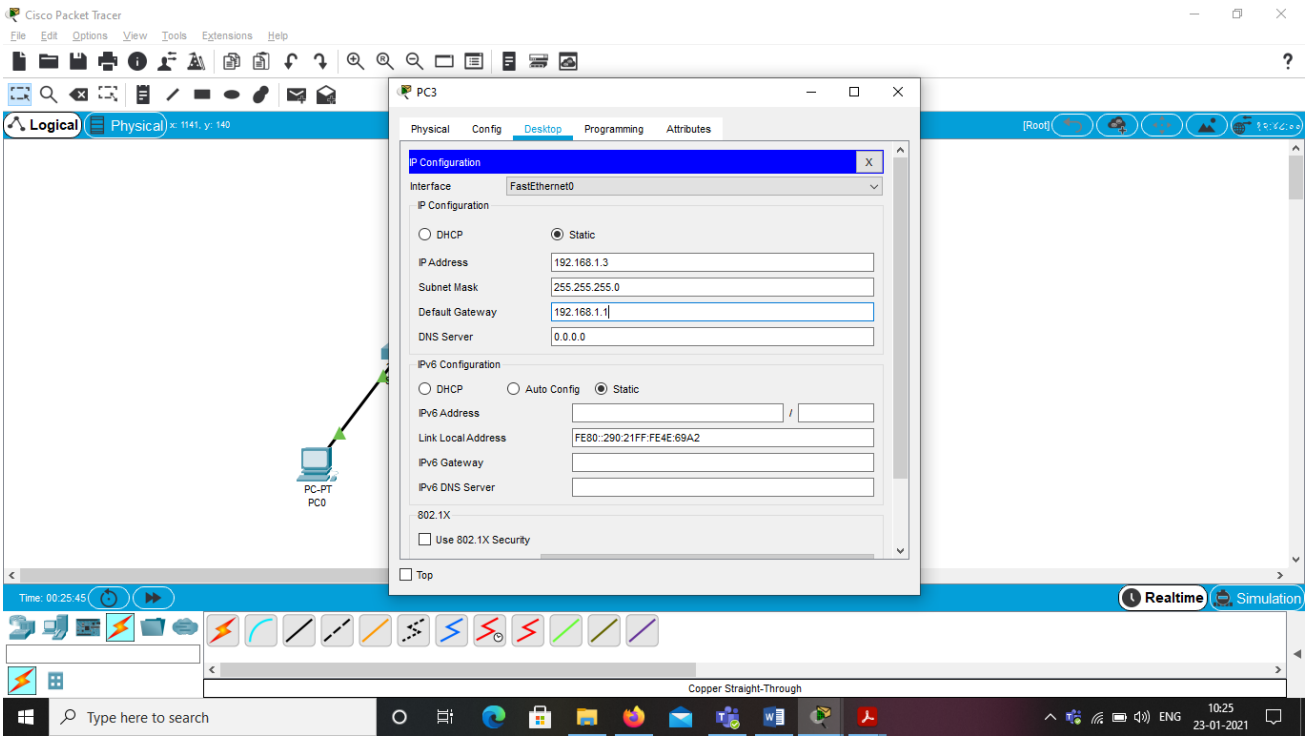






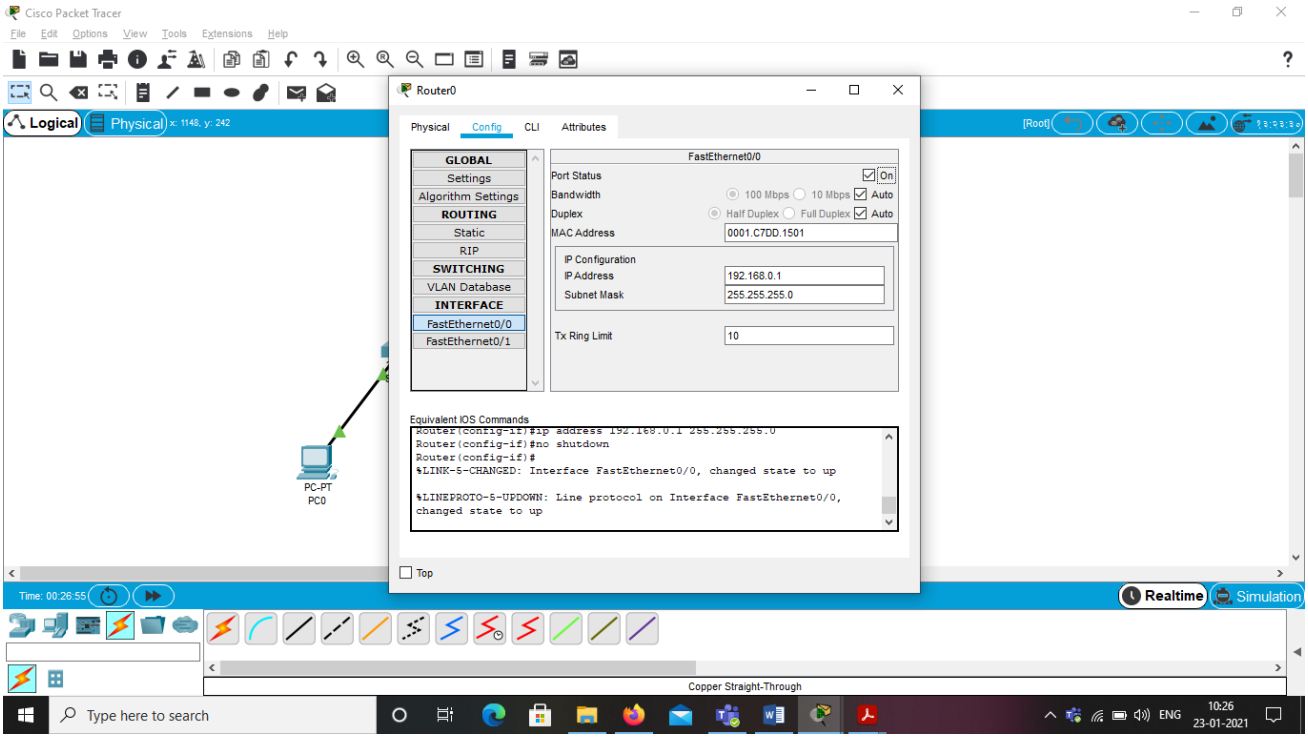
Use another network address for second network.



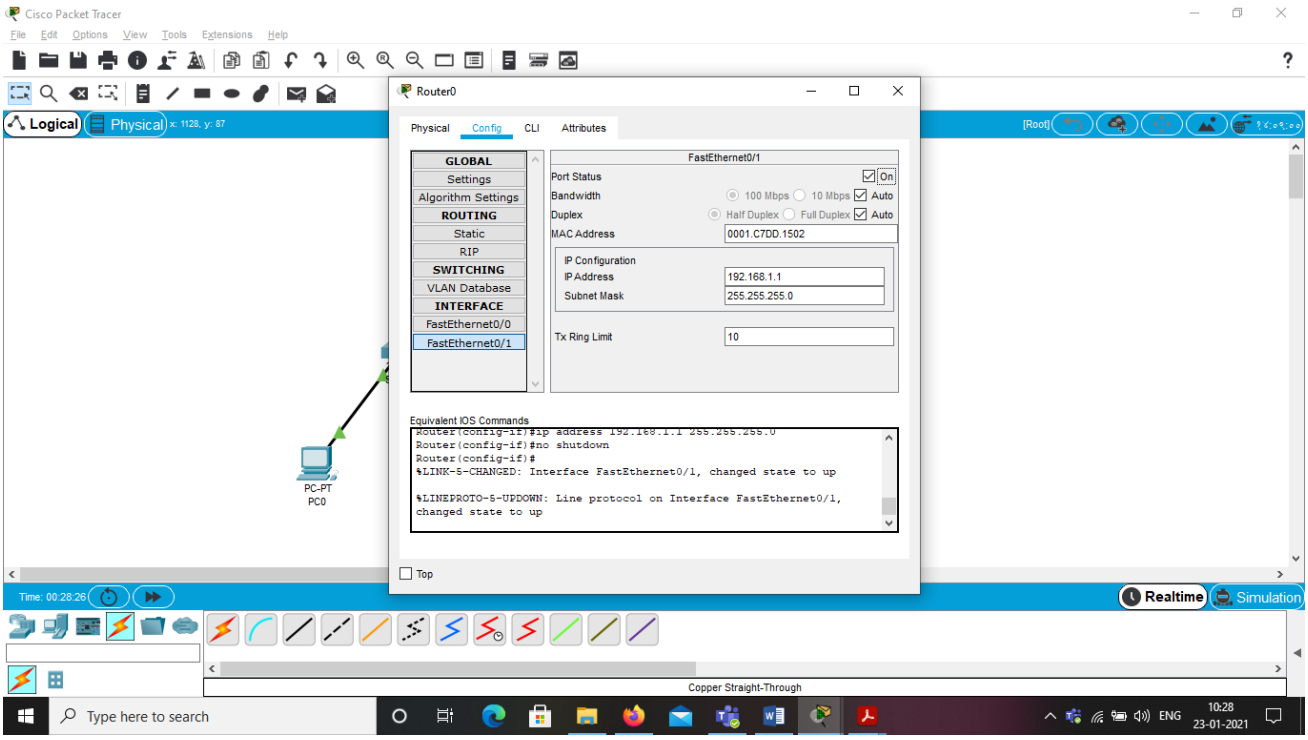


Step 3: Assign the IP address for router

Assign the gateway address of 1st network and don't forget to turn on the port status

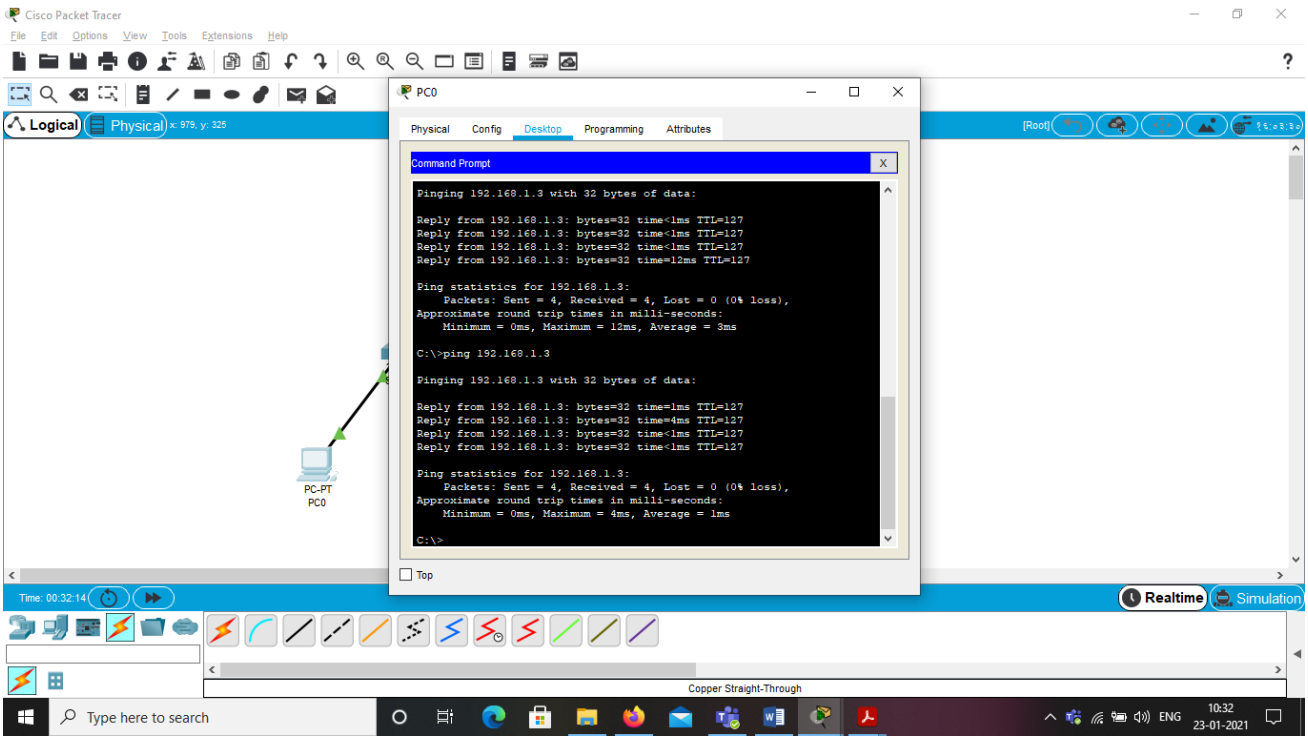


Assign the gateway address of 2nd network for FastEthernet0/1 interface

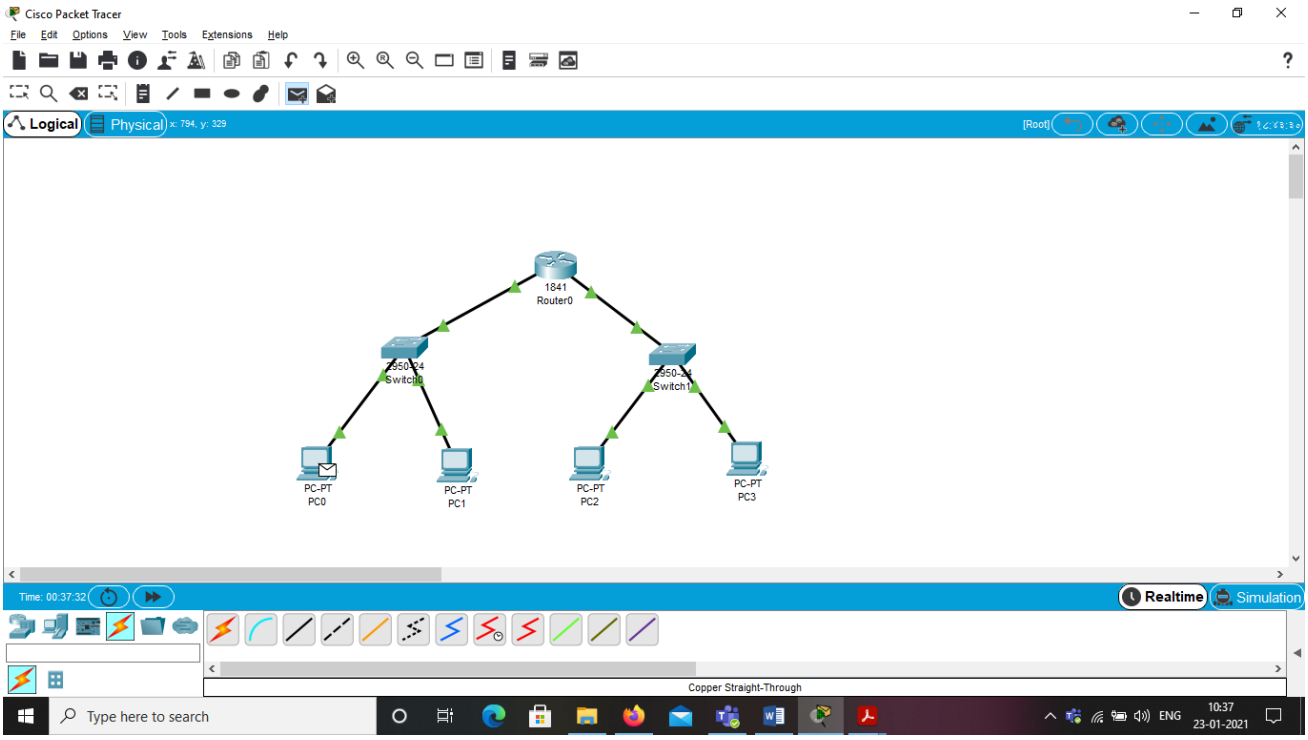


Step 3: Check the connectivity from one network to another network

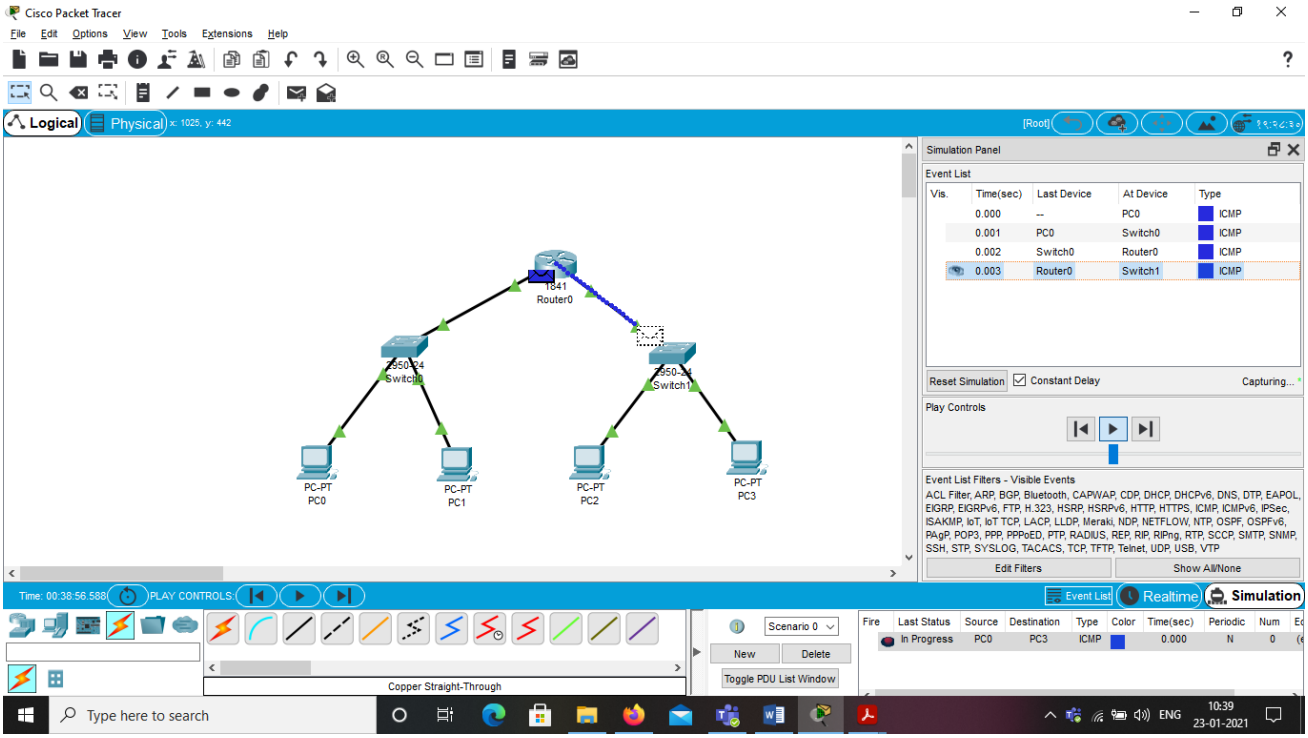
Select any PC from 1st network go to Desktop tab->Command Prompt->execute ping command for the 2nd network.



Step 4: Send Simple PDU.

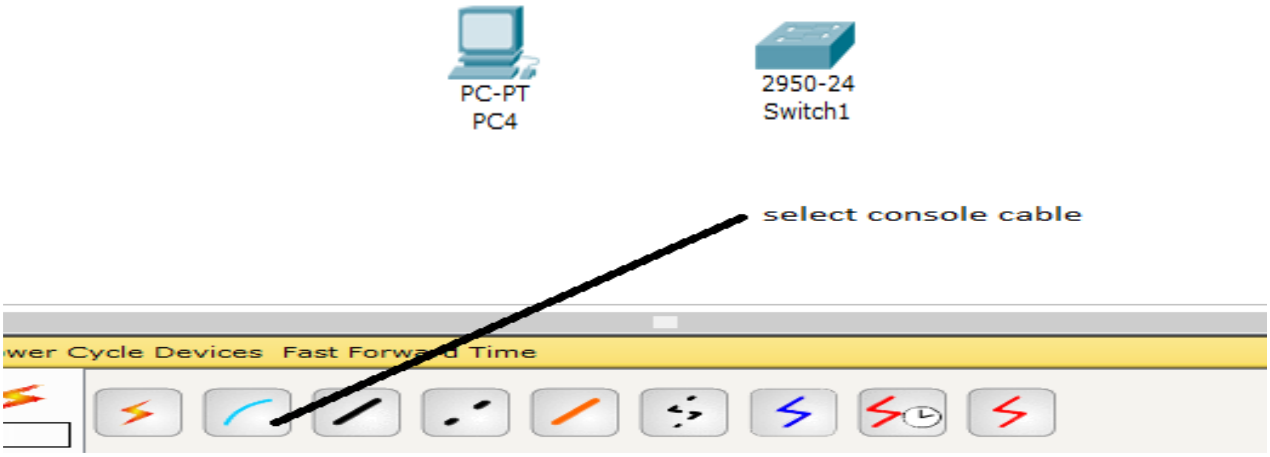


Step 5: Check in simulation mode

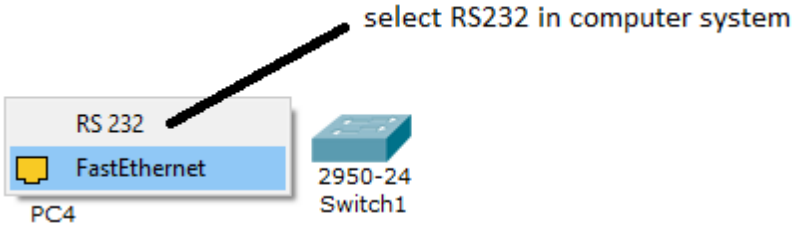

Cisco Packet Tracer
File Edit Options View Tools Extensions Help
Logical Physical
Router0
Switch0
Switch1
PC0
PC1
PC2
PC3
Simulation Panel
Event List
Vis. Time(sec) Last Device At Device Type
0.000 -- PC0 ICMP
0.001 PC0 Switch0 ICMP
0.002 Switch0 Router0 ICMP
0.003 Router0 Switch1 ICMP
Reset Simulation
Constant Delay
Capturing...
Play Controls
Event List Filters - Visible Events
Edit Filters
Show All/None
Realtime
Simulation
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color Time(sec) Periodic Num
In Progress PC0 PC3 ICMP 0.000 N 0
Copper Straight-Through
Type here to search
ENG
10:39
23-01-2021

Exp 2: Configuration of Switch using cisco packet tracer

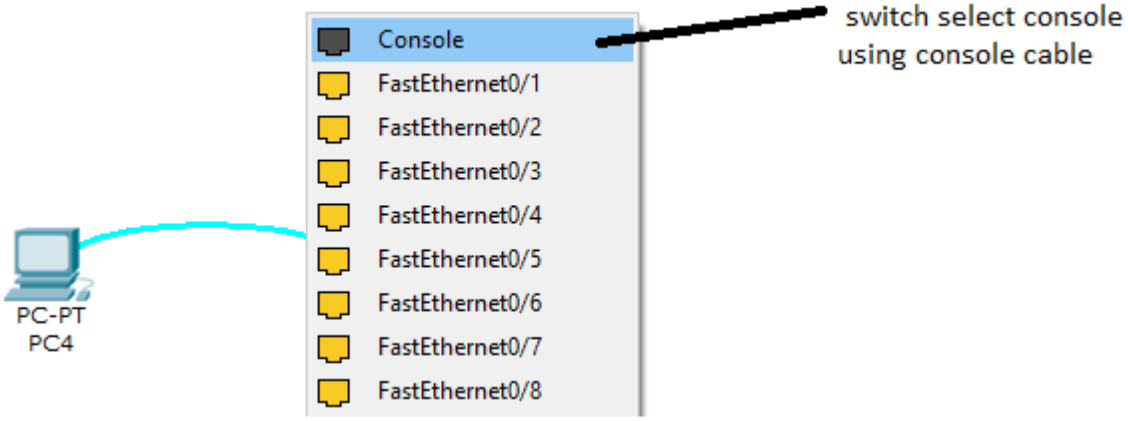
Step 1:



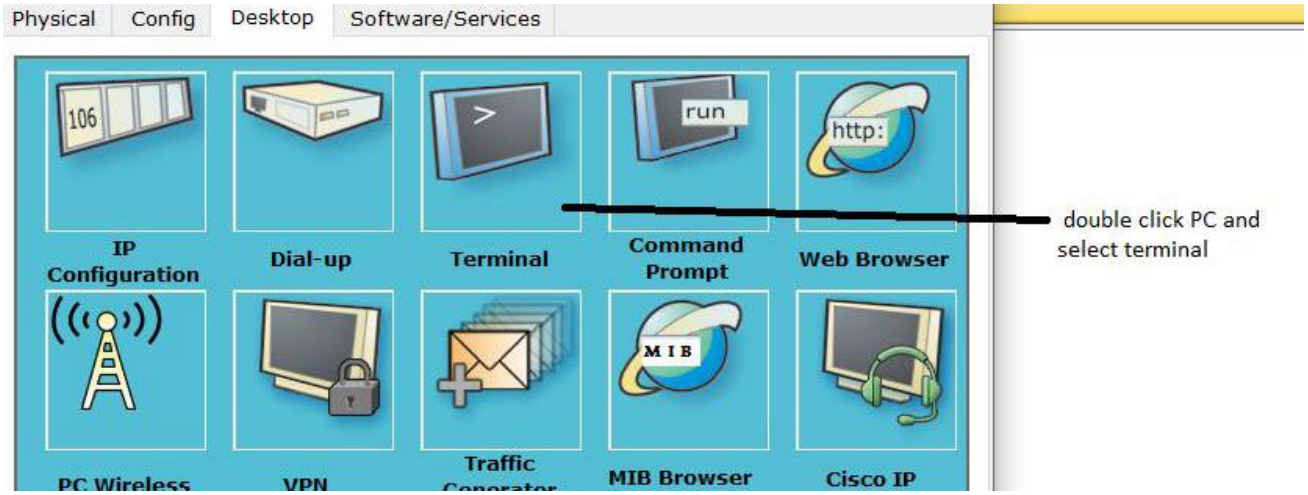
Step 2:

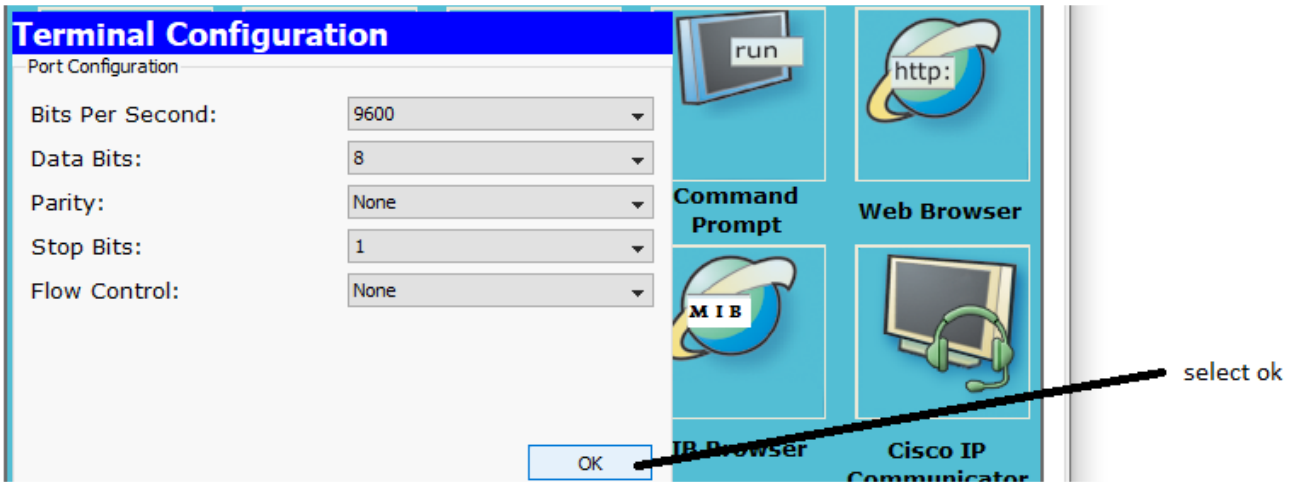


Step 3:



Step 4:





Basic commands:

switch> ---> User Mode

switch>enable --> Enters into the Privilege mode

switch# --> Privilege mode

switch# configure terminal (or) conf t --> Enable Configuration Mode

switch(config)# --> Configuration Mode

Helping commands

switch> ? --> Help to list the available commands in this mode

switch>te? --> Lists all the commands starts with "tel"

switch# ? --> Help

Step 1: Erase the startup configuration file from NVRAM.

Type the erase startup-config command to remove the startup configuration from nonvolatile random access memory (NVRAM).

Switch # erase startup-config

Erasing the nvram filesystem will remove all configuration files! Continue? [confirm]

[OK]

Erase of nvram: complete

Router#

Step 2: Reload the switch.

Issue the reload command to remove an old configuration from memory. When prompted to Proceed with reload, press Enter to confirm the reload. Pressing any other key will abort the reload.

switch# reload

Proceed with reload? [confirm]

*Nov 29 18:28:09.923: %SYS-5-RELOAD: Reload requested by console. Reload Reason: Reload Command.

Note: You may receive a prompt to save the running configuration prior to reloading the router. Respond by typing no and press Enter.

System configuration has been modified. Save? [yes/no]: no

Step 3:

Use the show flash command to determine if any VLANs have been created on the switch.

Switch# show flash

Directory of flash:/

2 -rwx 1919 Mar 1 1993 00:06:33 +00:00 private-config.text

3 -rwx 1632 Mar 1 1993 00:06:33 +00:00 config.text

4 -rwx 13336 Mar 1 1993 00:06:33 +00:00 multiple-fs

5 -rwx 11607161 Mar 1 1993 02:37:06 +00:00 c2960-lanbasek9-mz.150-2.SE.bin

6 -rwx 616 Mar 1 1993 00:07:13 +00:00 vlan.dat

32514048 bytes total (20886528 bytes free)

Switch#

Step 4

Switch#

Switch> show version

Cisco IOS Software, C2960 Software (C2960-LANBASEK9-M), Version 15.0(2)SE, RELEASE SOFTWARE (fc1)

Technical Support: http://www.cisco.com/techsupport

Copyright (c) 1986-2012 by Cisco Systems, Inc.

Compiled Sat 28-Jul-12 00:29 by prod_rel_team

ROM: Bootstrap program is C2960 boot loader

BOOTLDR: C2960 Boot Loader (C2960-HBOOT-M) Version 12.2(53r)SEY3, RELEASE SOFTWARE

(fc1)

Switch uptime is 2 minutes

System returned to ROM by power-on

System image file is "flash://c2960-lanbasek9-mz.150-2.SE.bin"

<output omitted>

Which IOS image version is currently in use by your switch?

Step 5 : Configure the clock.

As you learn more about networking, you will see that configuring the correct time on a Cisco switch can be helpful when you are troubleshooting problems. The following steps manually configure the internal clock of the switch.

a. Display the current clock settings.

Switch> **show clock**

*00:30:05.261 UTC Mon Mar 1 1993

b. Configure the clock setting. The question mark (?) provides help and allows you to determine the expected input for configuring the current time, date, and year. Press Enter to complete the clock configuration.

Switch# **clock set ?**

hh:mm:ss Current Time

Switch# **clock set 15:08:00 ?**

<1-31> Day of the month

MONTH Month of the year

Switch# **clock set 15:08:00 Oct 26 ?**

<1993-2035> Year

Switch# **clock set 15:08:00 Oct 26 2012**

Switch#

*Oct 26 15:08:00.000: %SYS-6-CLOCKUPDATE: System clock has been updated from 00:31:43

UTC Mon Mar 1 1993 to 15:08:00 UTC Fri Oct 26 2012, configured from console by

console.

c. Enter the show clock command to verify that the clock setting has updated.

Switch# **show clock**

15:08:07.205 UTC Fri Oct 26 2012

Step 6 : Give the switch a name.

Use the hostname command to change the switch name to S1.

Switch(config)# hostname S1

S1(config)#

Step 7:Enter a login MOTD banner.

A login banner, known as the message of the day (MOTD) banner, should be configured to warn anyone accessing the switch that unauthorized access will not be tolerated. The banner motd command requires the use of delimiters to identify the content of the banner message. The delimiting character can be any character as long as it does not occur in the message. For this reason, symbols, such as the #, are often used.

S1(config)# banner motd #

Enter TEXT message. End with the character '#'

Unauthorized access is strictly prohibited and prosecuted to the full extent

of the law. #

S1(config)# exit

S1#

Step 8: Save the configuration.

Use the copy command to save the running configuration to the startup file on non-volatile random access memory (NVRAM).

S1# copy running-config startup-config

Destination filename [startup-config]? [Enter]

Building configuration...

[OK]

S1#

Step 9 : Display the current configuration.

The show running-config command displays the entire running configuration, one page at a time. Use the spacebar to advance paging.

S1# show running-config

Building configuration...

Current configuration : 1409 bytes

!

! Last configuration change at 03:49:17 UTC Mon Mar 1 1993

Step 10 : Display the status of the connected interfaces on the switch.

To check the status of the connected interfaces, use the show ip interface brief command. Press the spacebar to advance to the end of the list.

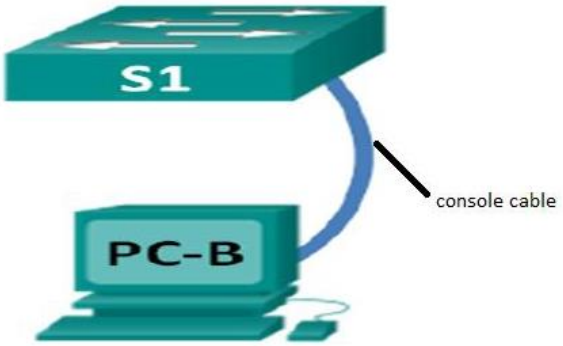
S1# **show ip interface brief**

Interface IP-Address OK? Method Status Protocol
Vlan1 unassigned YES unset up up
FastEthernet0/1 unassigned YES unset up up
FastEthernet0/2 unassigned YES unset down down
FastEthernet0/3 unassigned YES unset down down
FastEthernet0/4 unassigned YES unset down down
FastEthernet0/5 unassigned YES unset down down
FastEthernet0/6 unassigned YES unset up up
FastEthernet0/7 unassigned YES unset down down
FastEthernet0/8 unassigned YES unset down down
FastEthernet0/9 unassigned YES unset down down

Step 11: Show vlan:

Show vlan command will display the VLANs. For administrative purpose, switch automatically create VLAN 1 and assign all its interfaces to it. You can create custom VLANs from global configuration mode and then assign them to interfaces.

Exp 3: Configure the privilege level password and user authentication in switch.



I How to Set Hostname and Configure Console Password

1. To set the host name

Switch(config)# hostname CISCO

2. To set console password

CISCO((config)#

CISCO((config)#line console 0

CISCO((config-line)#password cisco123

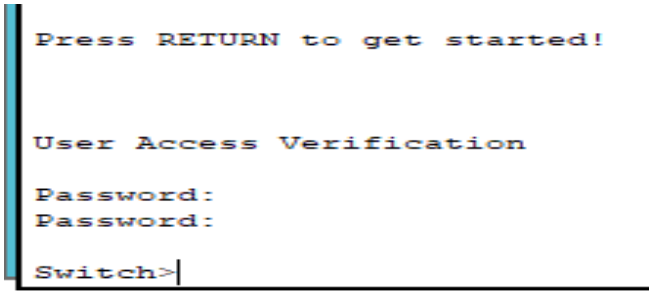
CISCO((config-line)#login

CISCO((config-line)#exit

CISCO((config)#exit

CISCO(#exit

Check the Console Password



II How to Set Privilege level password

1)Set a privilege password

!!! Clear Text Password not encrypted(less priority)

CISCO(config)# enable password muscat

!!! Encrypted password (more Priority)

CISCO(config)# enable secret nizwa

2) Verify the privilege Password

CISCO(config)# exit

CISCO# exit

CISCO con0 is now available

Press RETURN to get started.

User Access Verification

!!! TYPE HERE LINE CONSOLE Password

Password:

CISCO>enable

!!! TYPE HERE Privilege Level Password

Password:

III How to Set User Authentication in Switch

1)Set user authentication

CISCO# conf t

CISCO(config)# line console 0

CISCO(config-line)# login local

CISCO(config-line)# exit

CISCO(config)#username network password pwd

2) Verify the Authentication

CISCO(config)# exit

CISCO# exit

User Access Verification

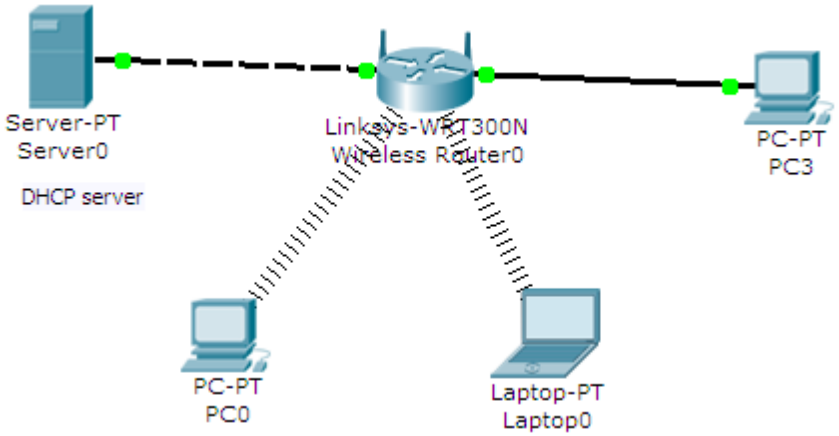
Username: network

Password:

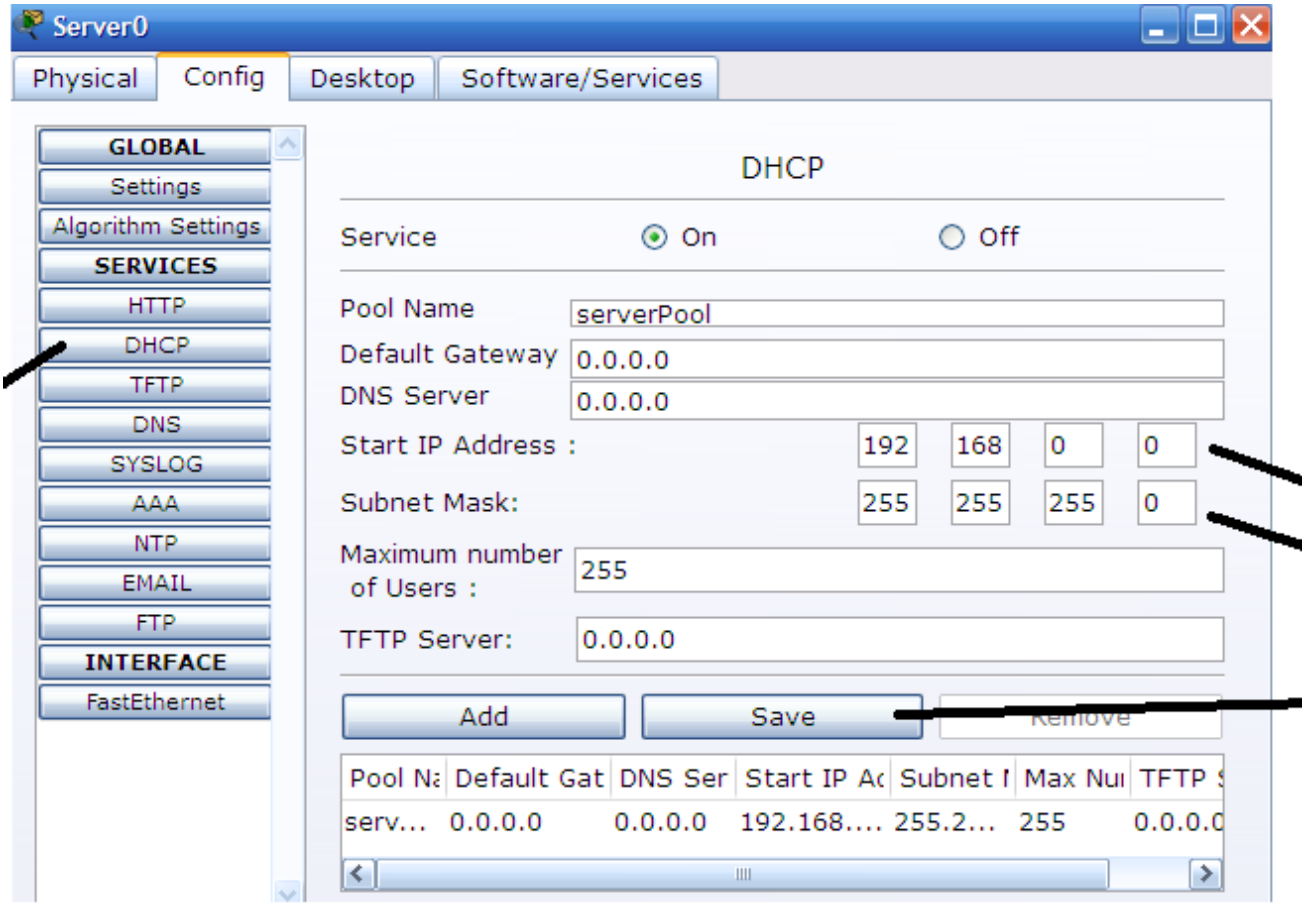
CISCO> enable

Password:

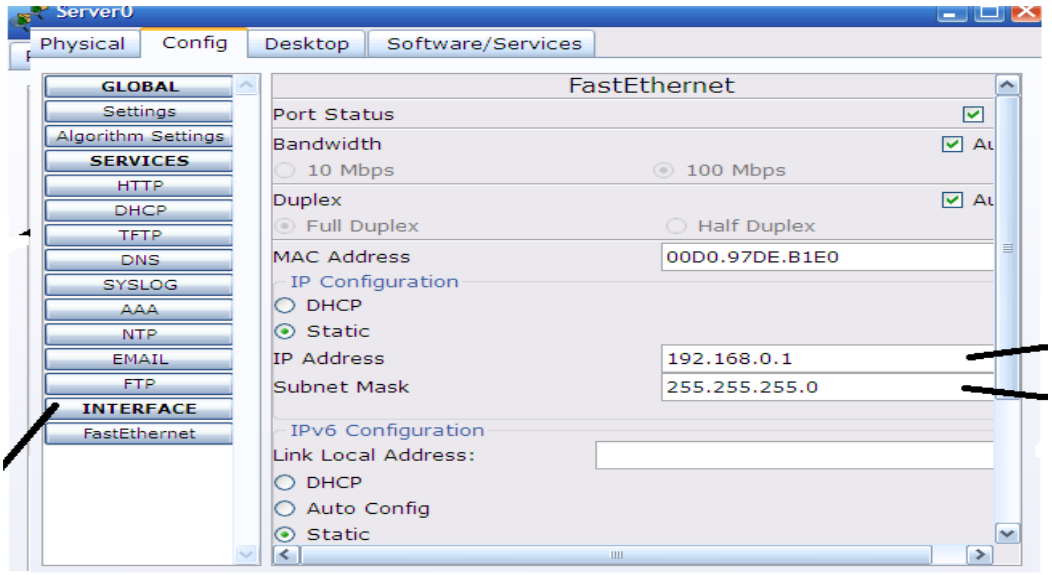
Exp 4: Configure the DHCP Server and wireless router and check the connectivity



Step1 :configure the DHCP server

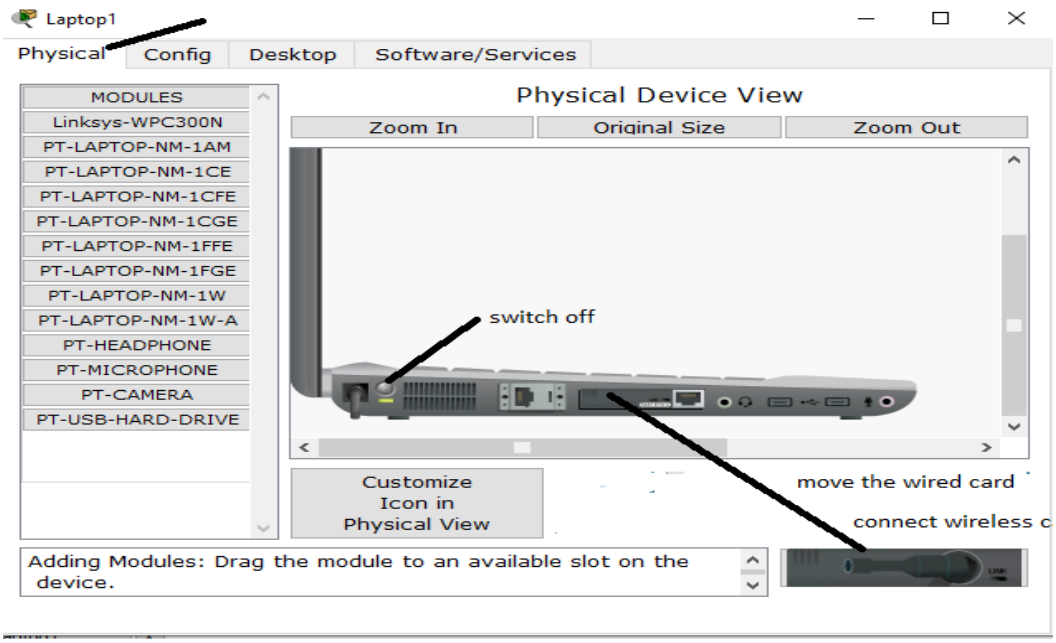


Step2: In server configure the fast Ethernet

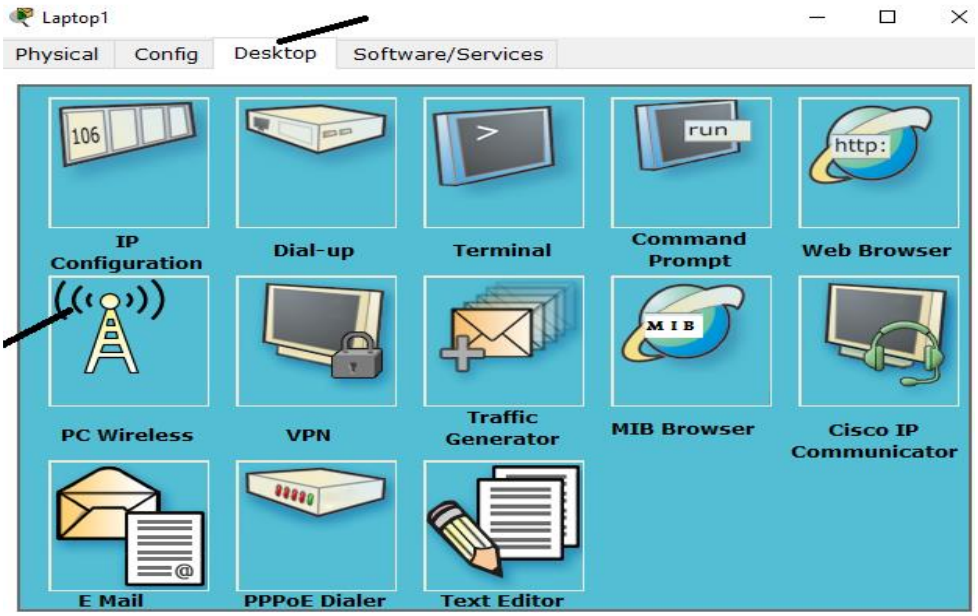


Step 3: configure the wireless router

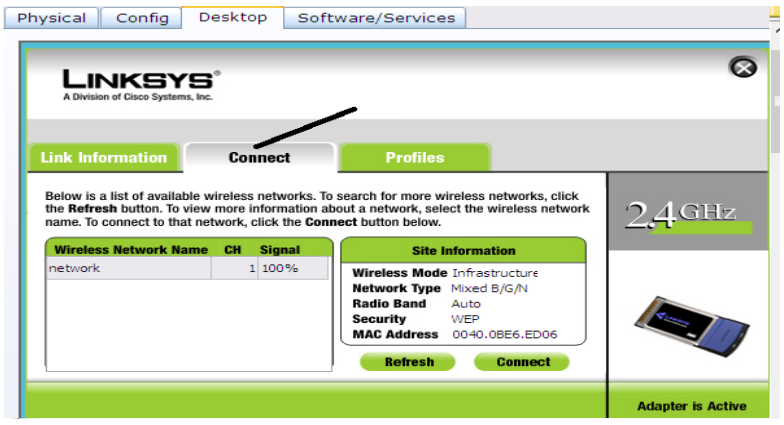
Step4 :Laptop remove the wired NIC card to wireless card



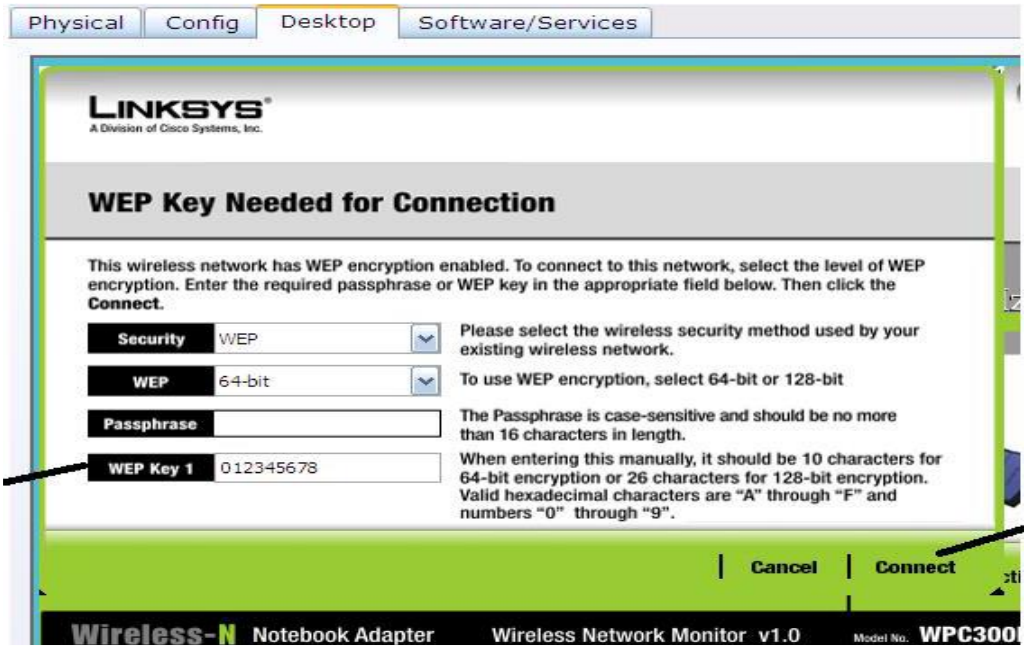
Step 5: Laptop connect the wireless



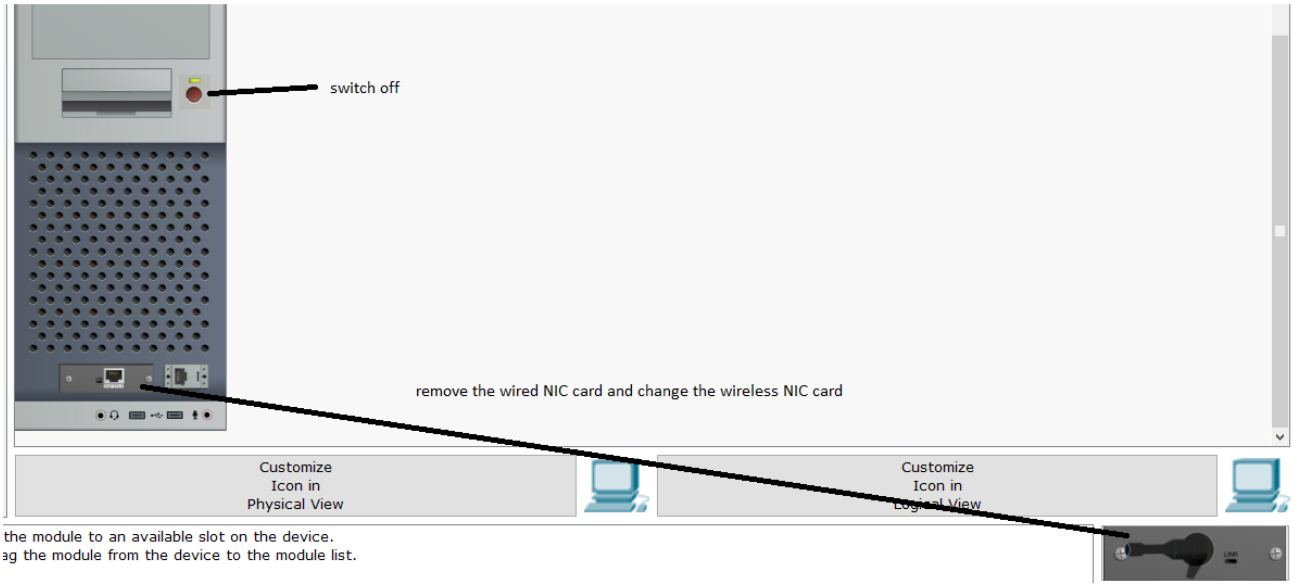
Step 6 : connect Laptop with wireless router



Step 7 :enter web key

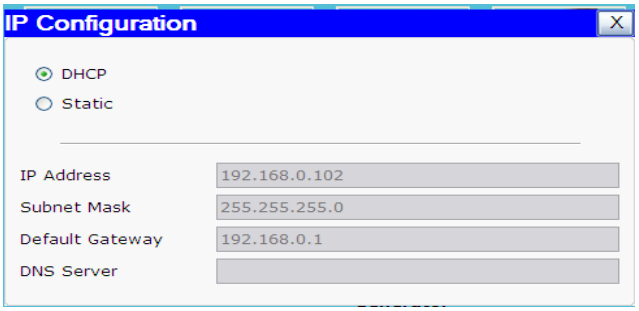


Step8 : PC remove the wired NIC card to wireless card

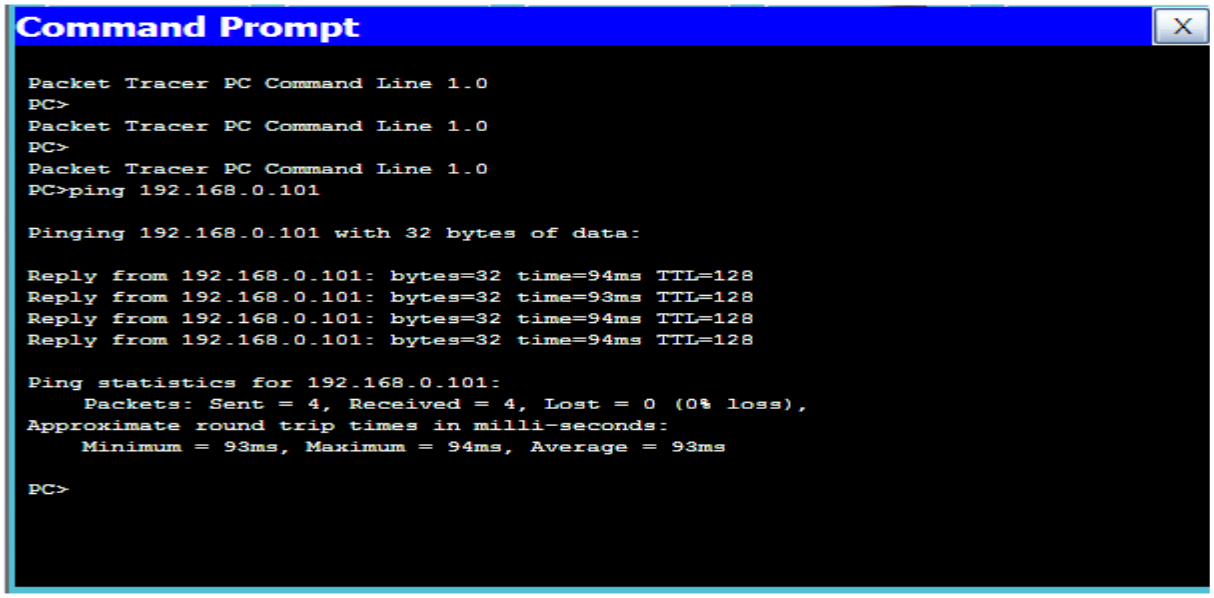


Note : Same steps 5,6,7

Step 9 : check the Pc and laptop for the DHCP ip address

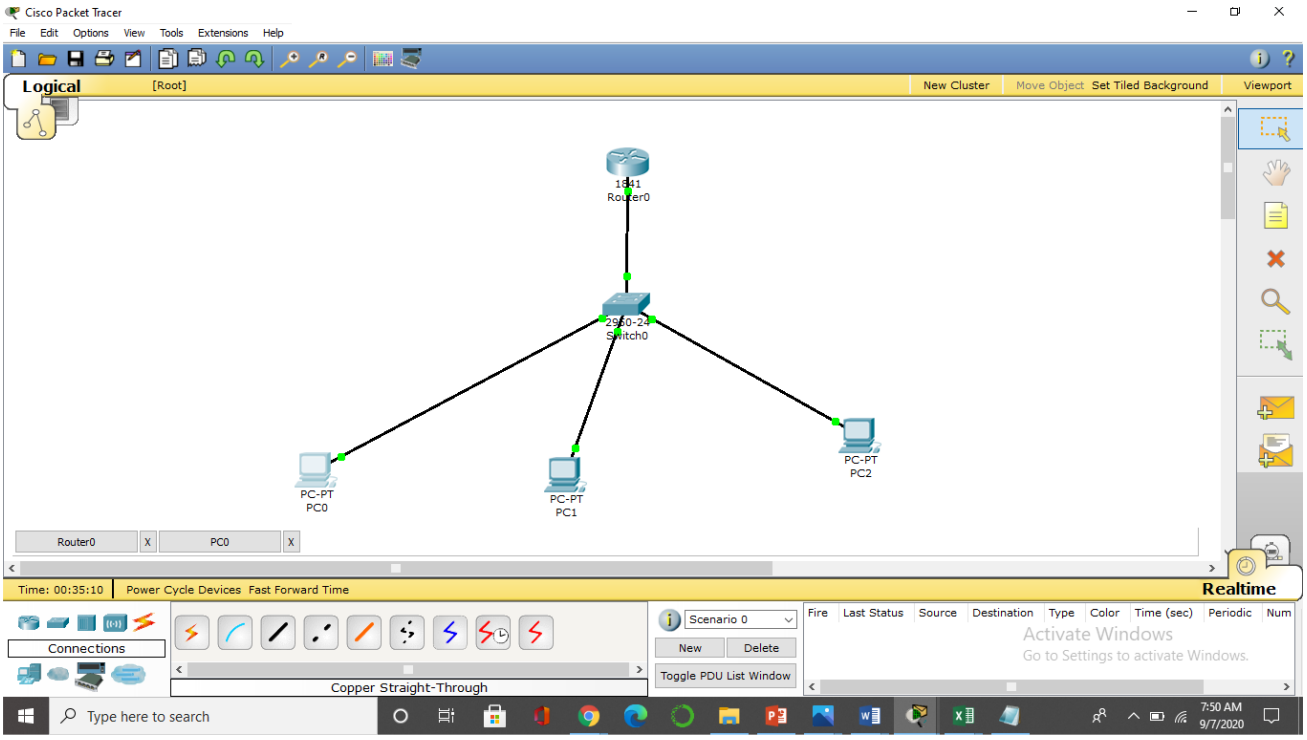


Step10 check the connectivity



Exp 5: Configure the telnet protocol using cisco packet tracer

Step 1: Draw a topology as shown below and assign IP address to all PC's.



Step 2:Configure IP address to router.

Router(config)#interface FastEthernet0/0

Router(config-if)#ip address 10.0.0.1 255.0.0.0

Router(config-if)#no shutdown

Router(config-if)#exit

Step 3:To set privilege mode password

Click on Router and go to CLI tab and type below.

Router(config)#enable password 1234

Router(config)#exit

Step 4:To configure telnet.

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#line vty 0 4

Router(config-line)#password cisco

Router(config-line)#login

Router(config-line)#exit

Step 5:To check telnet configuration.

Router#sh run

Building configuration...

Current configuration : 556 bytes

!

version 12.4

no service timestamps log datetime msec

no service timestamps debug datetime msec

no service password-encryption

!

hostname Router

!

!

!

enable password 1234

!

!

ODD SEM 2020-21 PRESIDENCY UNIVERSITY, BENGALURU

!

!

!

!

!

!

!

!

!

!

```
spanning-tree mode pvst
!
!
!
!
interface FastEthernet0/0
ip address 10.0.0.1 255.0.0.0
duplex auto
speed auto
!
interface FastEthernet0/1
no ip address
duplex auto
speed auto
shutdown
!
interface Vlan1
no ip address
shutdown
!
ip classless
!
!
!
!
!
!
!
line con 0
```

line vty 0 4

password cisco

login

line vty 5 6

password cisco

login

!

!

!

End

Step 6:To access cisco router via telnet connection from any PC.

Click on any PC>click on desktop>select command prompt and then type below commands

PC>ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32 time=13ms TTL=255

Reply from 10.0.0.1: bytes=32 time=16ms TTL=255

Reply from 10.0.0.1: bytes=32 time=16ms TTL=255

Reply from 10.0.0.1: bytes=32 time=16ms TTL=255

Ping statistics for 10.0.0.1:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 13ms, Maximum = 16ms, Average = 15ms

PC>telnet 10.0.0.1

Trying 10.0.0.1 ...Open

User Access Verification

Password:

Router>en

Password:

Router#

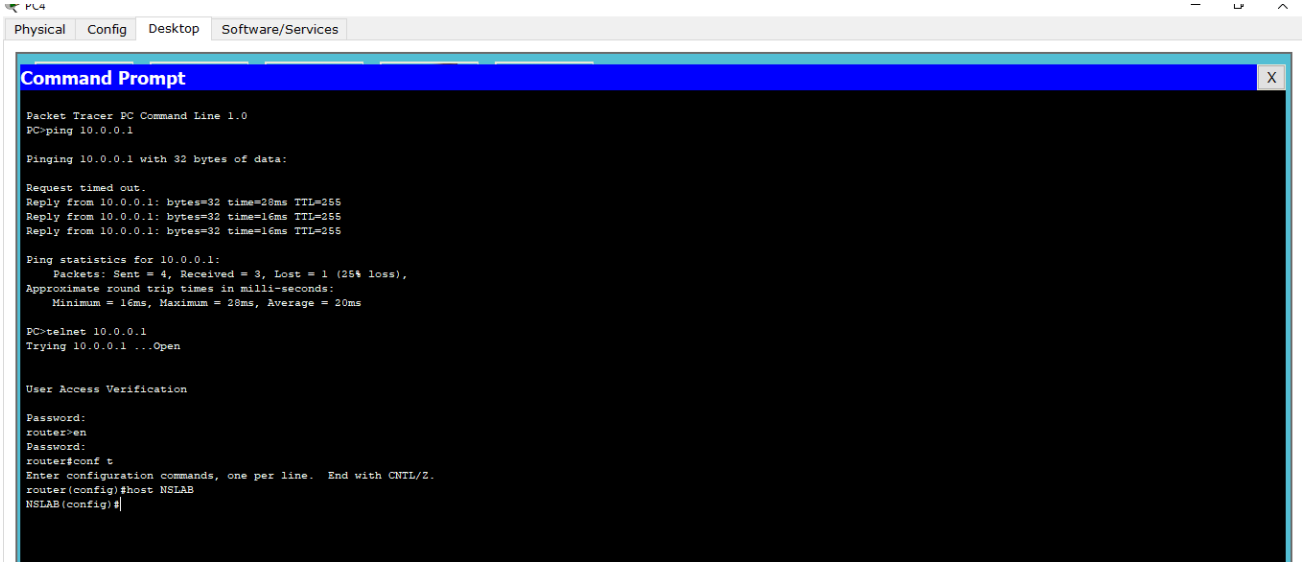
We can change hostname of router in PC command prompt.

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

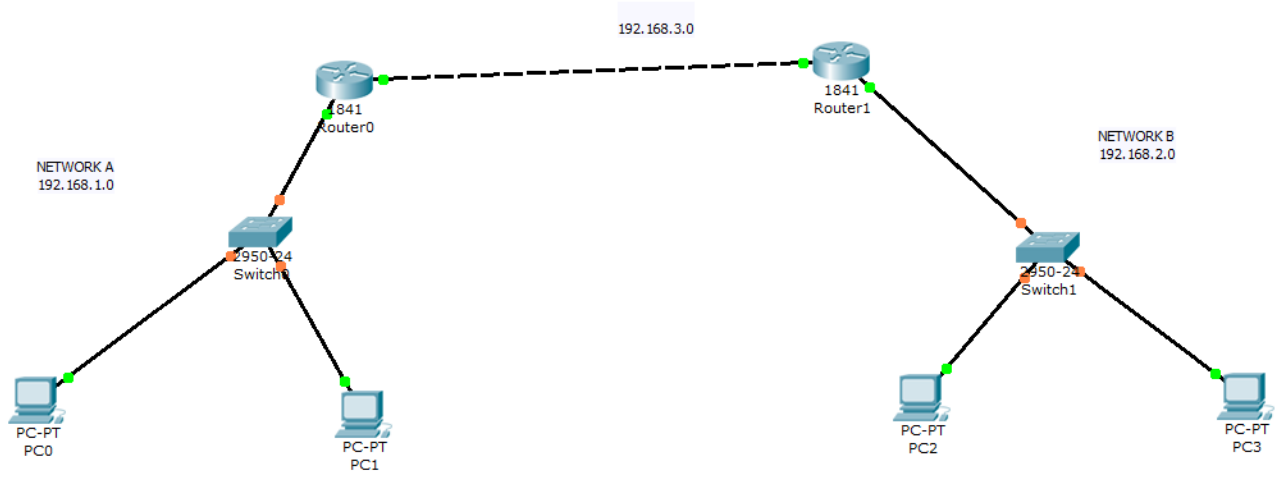
Router(config)#host nslab

nslab(config)#



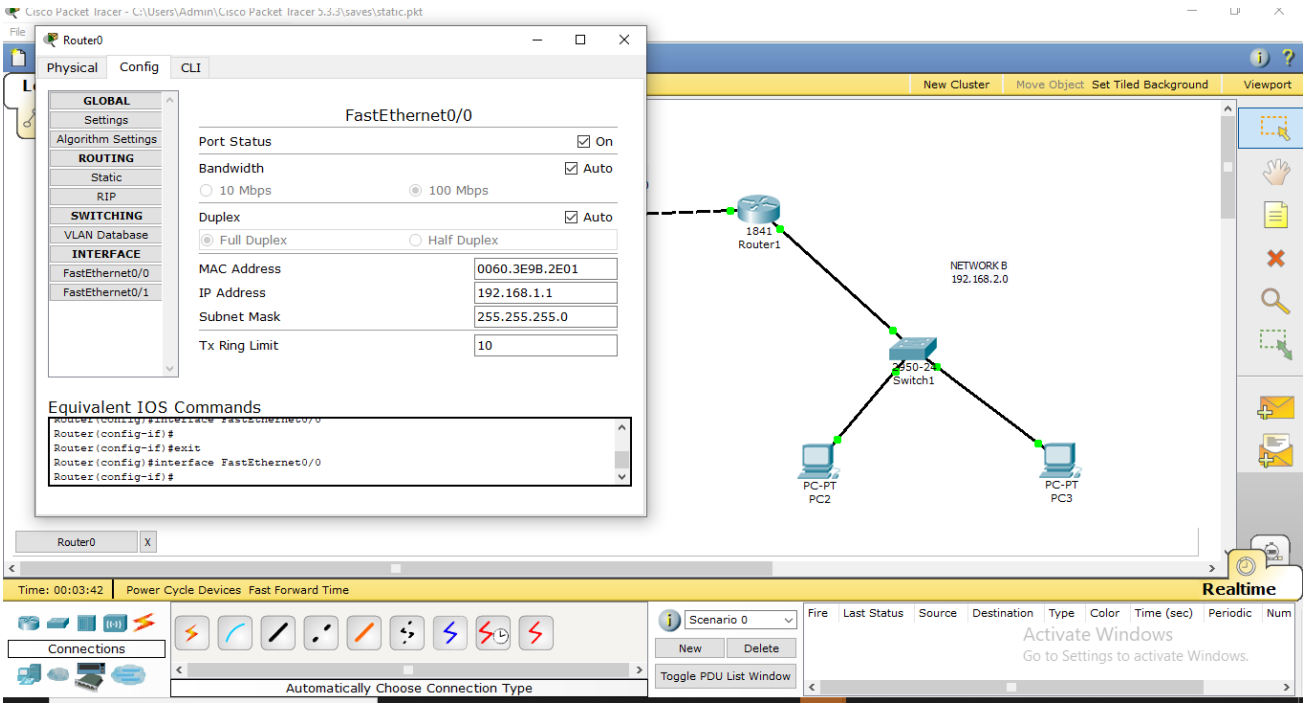
Exp 6: Configure the static routing using cisco packet tracer.

Step 1: Draw a topology as shown below and assign IP address to all PC's.

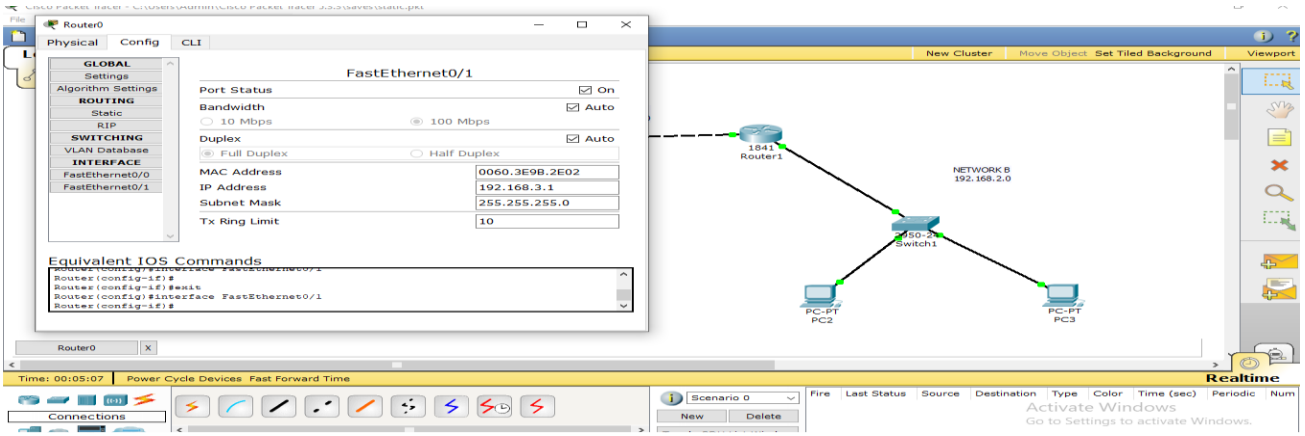


Step 2:Configure IP address to router1.

For Fastethernet 0/0

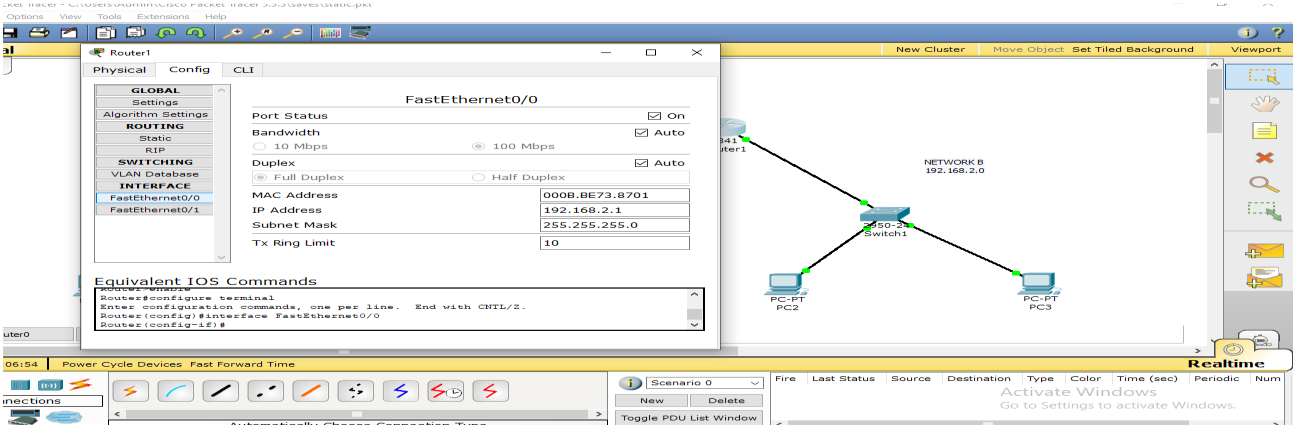


For Fastethernet 0/1

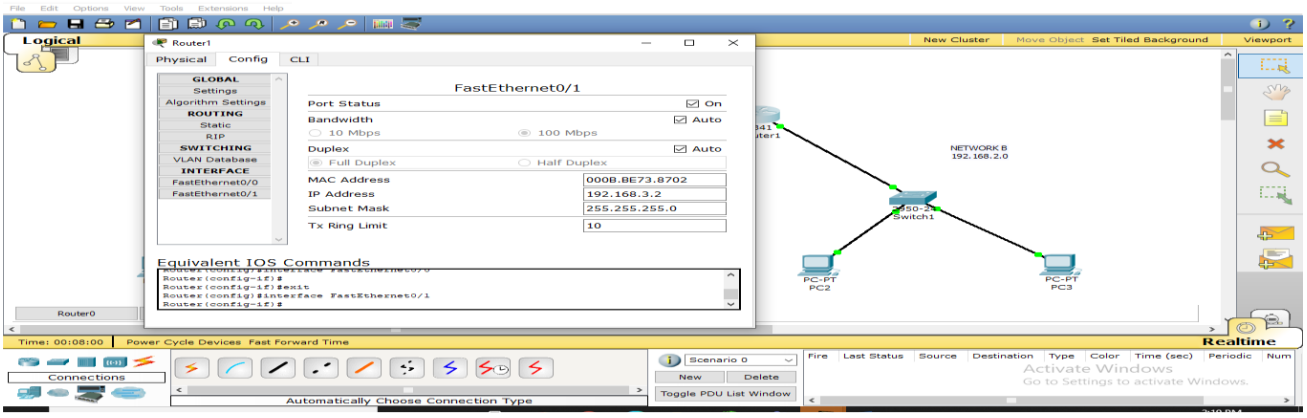


Step 3 :Configure IP address to router2

For Fastethernet 0/0



For FastEthernet 0/1



Step 4:To set up Static Routing

For Router 1:

In CLI:

Router(config)#

Router(config)#ip route 192.168.2.0 255.255.255.0 192.168.3.2

In Config Window:

Click on Static

Then Add Opposite Network Address 192.168.2.0 and Next Hop address 192.168.3.1 Along with Subnet Mask Address 255.255.255.0

For Router 2:

In CLI:

Router(config)#

Router(config)#ip route 192.168.1.0 255.255.255.0 192.168.3.1

In Config Window:

Click on Static

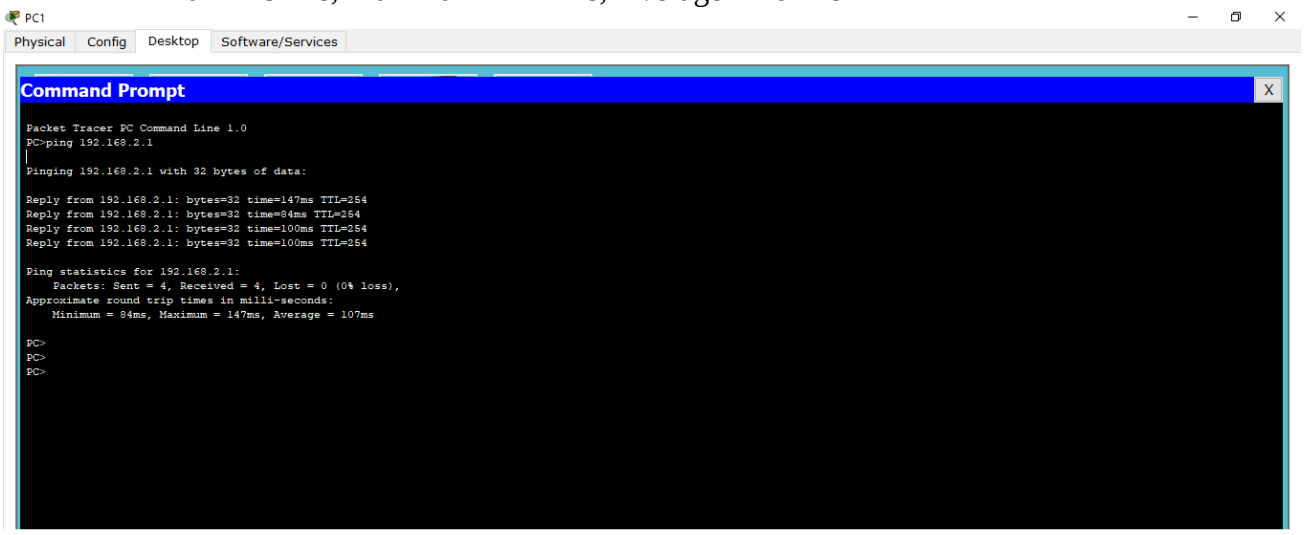
Then Add Opposite Network Address 192.168.1.0 and Next Hop address 192.168.3.2 Along with Subnet Mask Address 255.255.255.0

Step 5 :To Check Connectivity between two network using Static Routing

Click on any PC>click on desktop>select command prompt and then type below commands

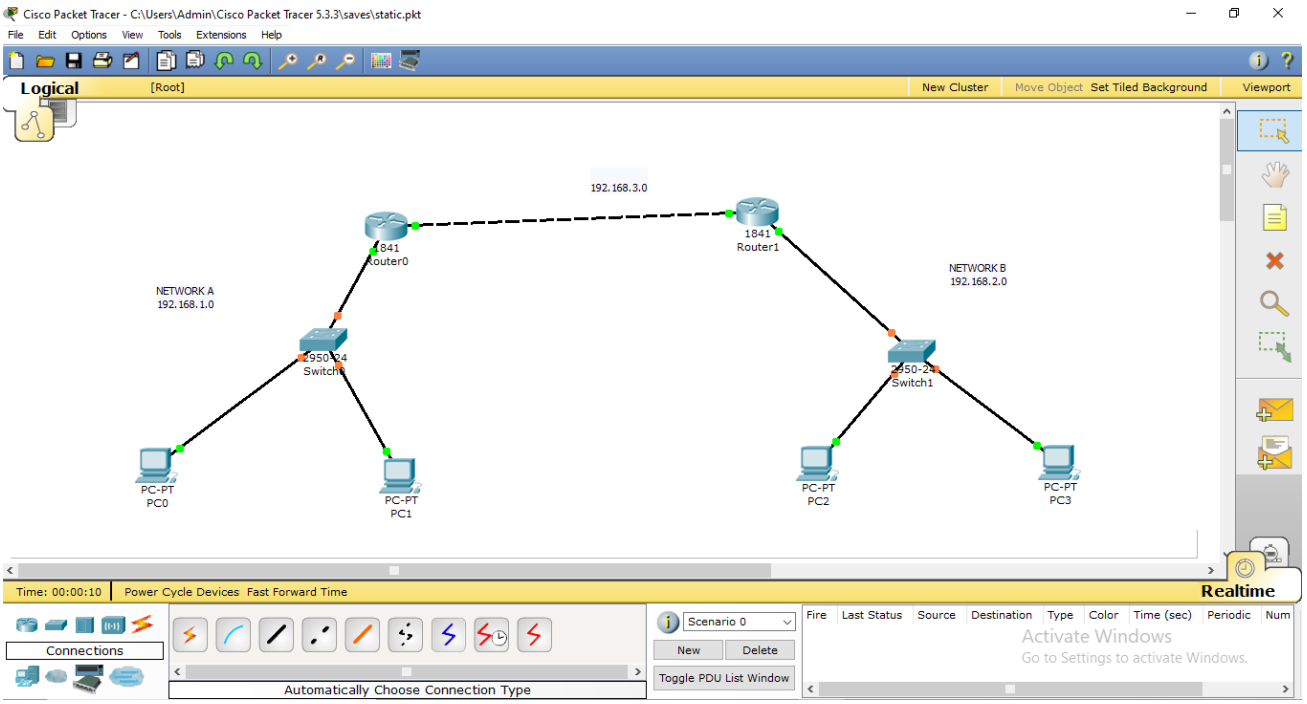
PC>ping 192.168.2.1

Pinging 192.168.2.1 with 32 bytes of data:
Reply from 192.168.2.1: bytes=32 time=147ms TTL=254
Reply from 192.168.2.1: bytes=32 time=84ms TTL=254
Reply from 192.168.2.1: bytes=32 time=100ms TTL=254
Reply from 192.168.2.1: bytes=32 time=100ms TTL=254
Ping statistics for 192.168.2.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 84ms, Maximum = 147ms, Average = 107ms



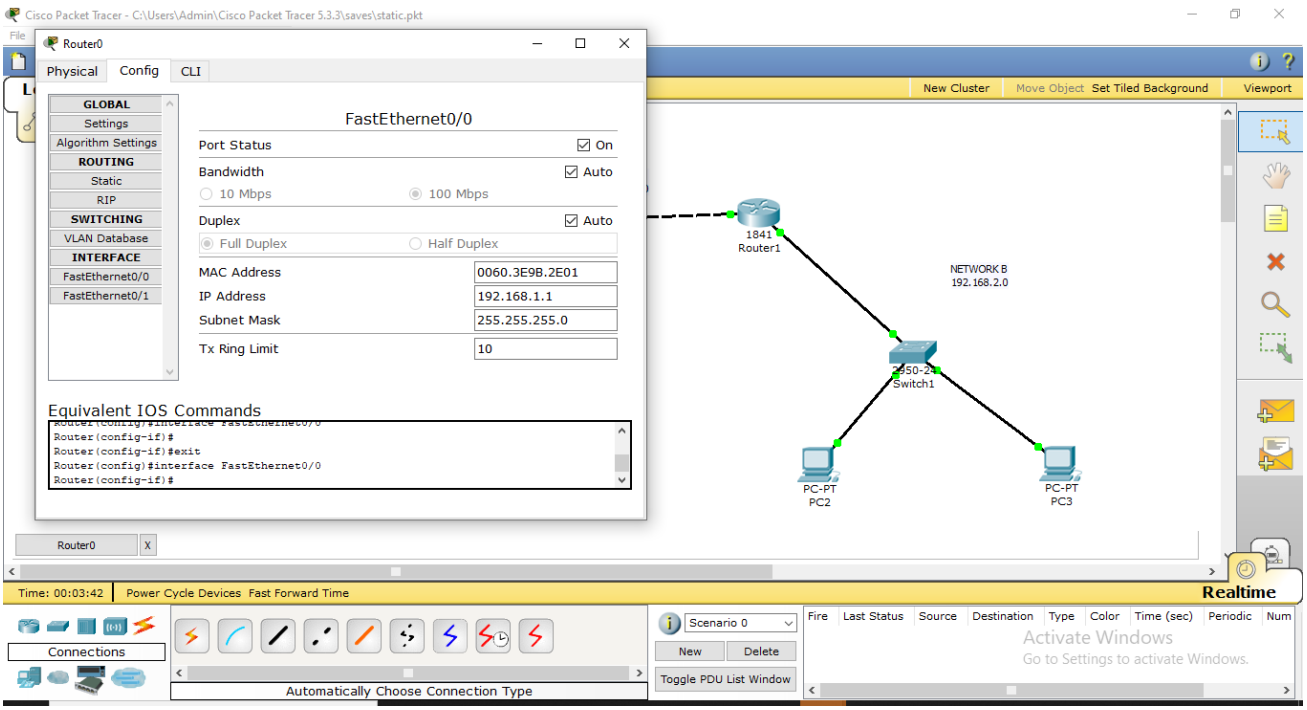
Exp 7: Configure the RIP routing using cisco packet tracer.

Step 1: Draw a topology as shown below and assign IP address to all PC's.

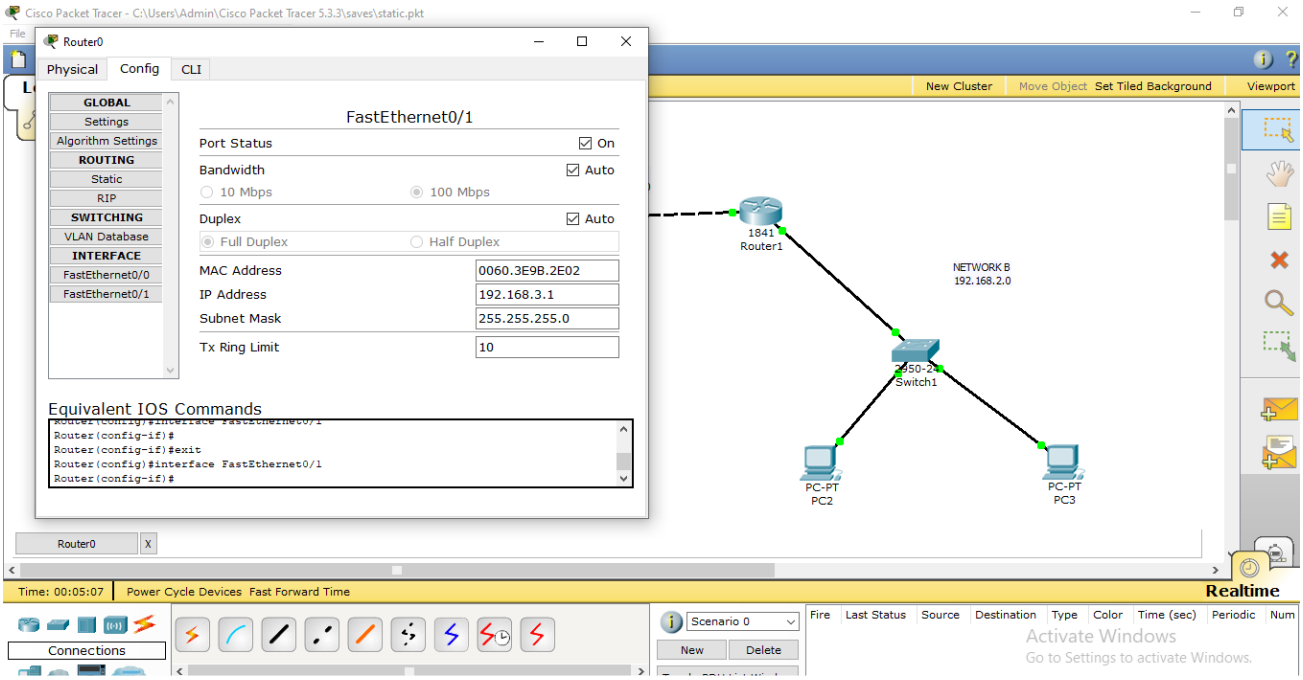


Step 2:Configure IP address to router1.

For Fastethernet 0/0

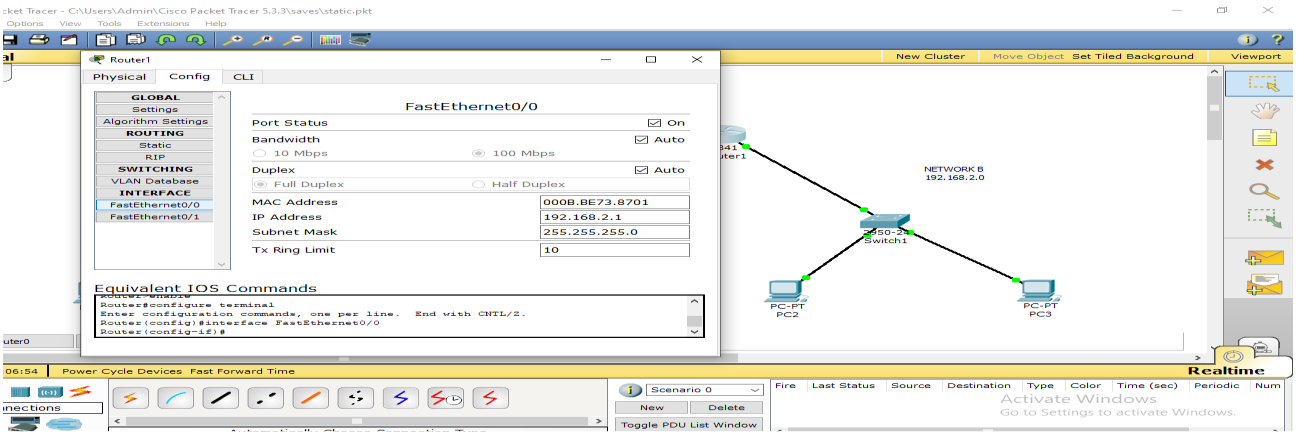


For Fastethernet 0/1

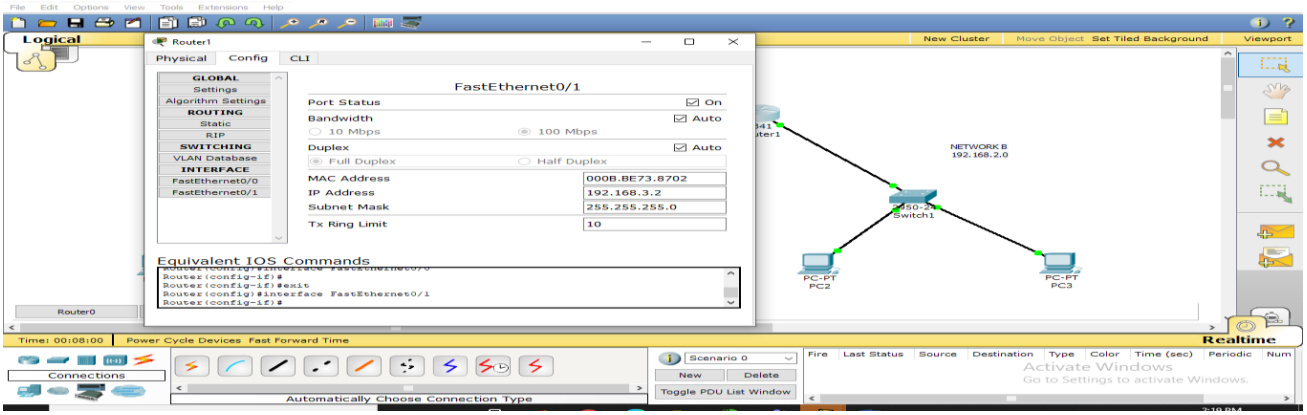


Step 3 :Configure IP address to router2

For Fastethernet 0/0



For FastEthernet 0/1



Step 4:To set up Dynamic Routing

For Router 1:

In CLI:

Router(config)#router rip

Router(config-router)#network 192.168.3.0

Router(config-router)#network 192.168.2.0

In Config Window:

Click on RIP

Then Add Opposite Network Address 192.168.2.0 and 192.168.3.0

For Router 2:

In CLI:

Router(config)#router rip

Router(config-router)#network 192.168.3.0

Router(config-router)#network 192.168.1.0

In Config Window:

Click on RIP

Then Add Opposite Network Address 192.168.1.0 and 192.168.3.0

Step 5 :To Check Connectivity between two network using RIP routing

Click on any PC>click on desktop>select command prompt and then type below commands

PC>ping 192.168.2.1

Pinging 192.168.2.1 with 32 bytes of data:

Reply from 192.168.2.1: bytes=32 time=147ms TTL=254

Reply from 192.168.2.1: bytes=32 time=84ms TTL=254

Reply from 192.168.2.1: bytes=32 time=100ms TTL=254

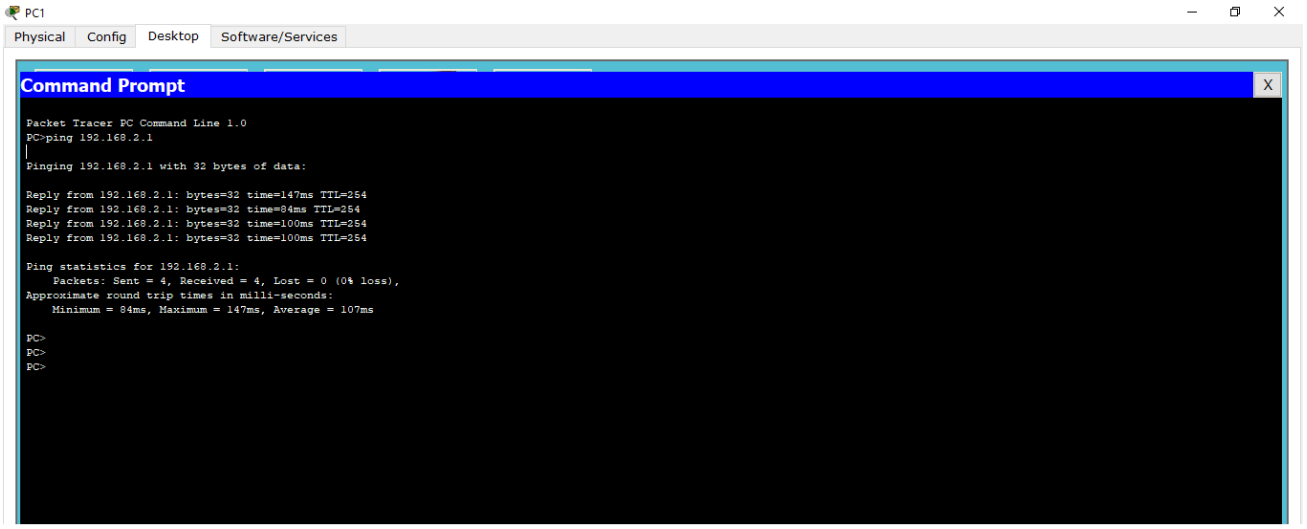
Reply from 192.168.2.1: bytes=32 time=100ms TTL=254

Ping statistics for 192.168.2.1:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 84ms, Maximum = 147ms, Average = 107ms

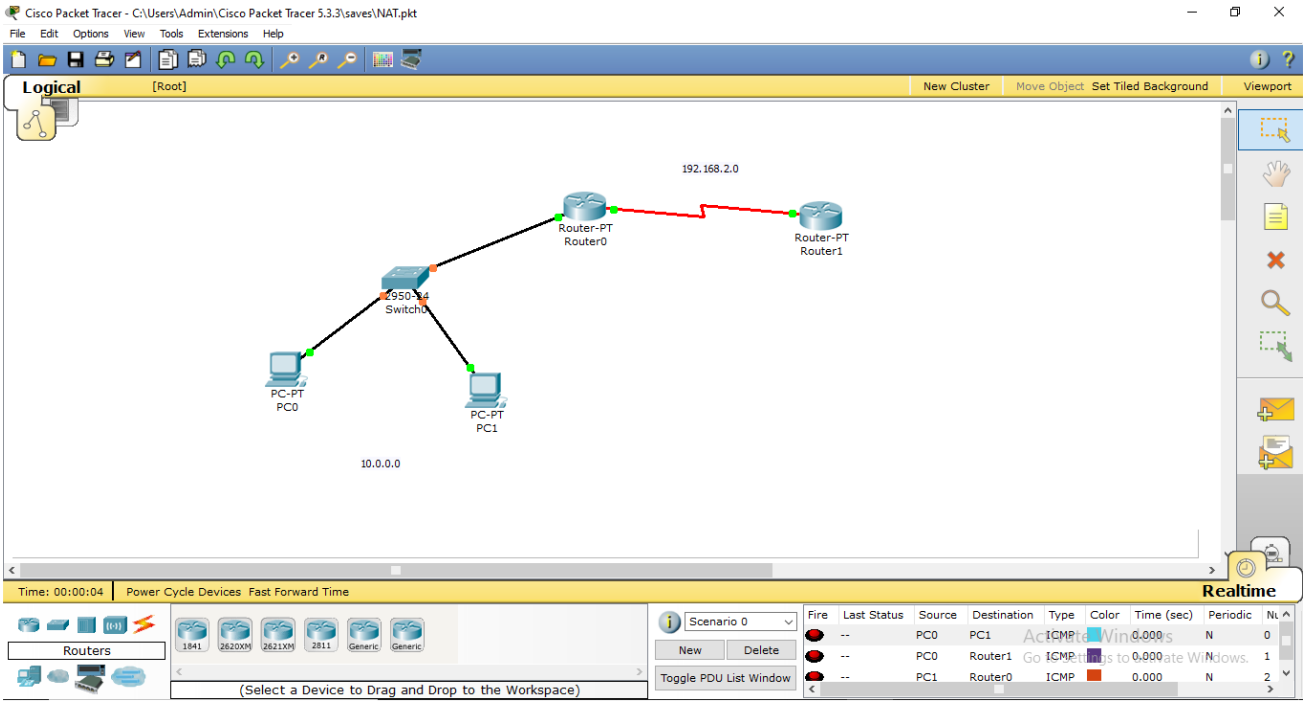

PC1
Physical
Config
Desktop
Software/Services
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 192.168.2.1
Pinging 192.168.2.1 with 32 bytes of data:
Reply from 192.168.2.1: bytes=32 time=147ms TTL=254
Reply from 192.168.2.1: bytes=32 time=84ms TTL=254
Reply from 192.168.2.1: bytes=32 time=100ms TTL=254
Reply from 192.168.2.1: bytes=32 time=100ms TTL=254
Ping statistics for 192.168.2.1:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 84ms, Maximum = 147ms, Average = 107ms
PC>
PC>
PC>

Exp 8: Configure the Static NAT using cisco packet tracer.

NAT(NETWORK ADDRESS TRANSLATION)

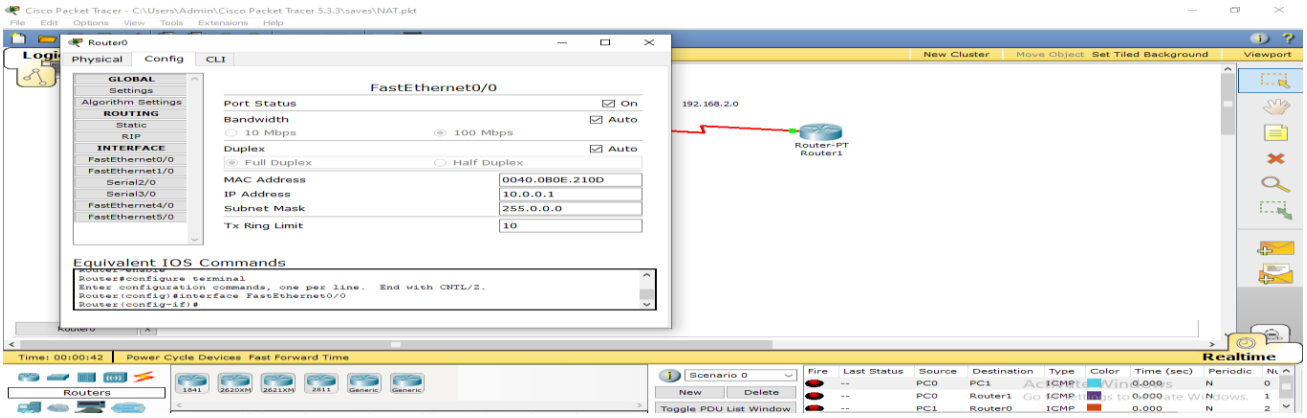
It is a Process in which one or more Local ip address is translated into global ip address or vice versa in order to provide internet access to the host. It Allows multiple devices to access internet through single public ip address.

Step 1: Draw a topology as shown below and assign IP address to all PC's.

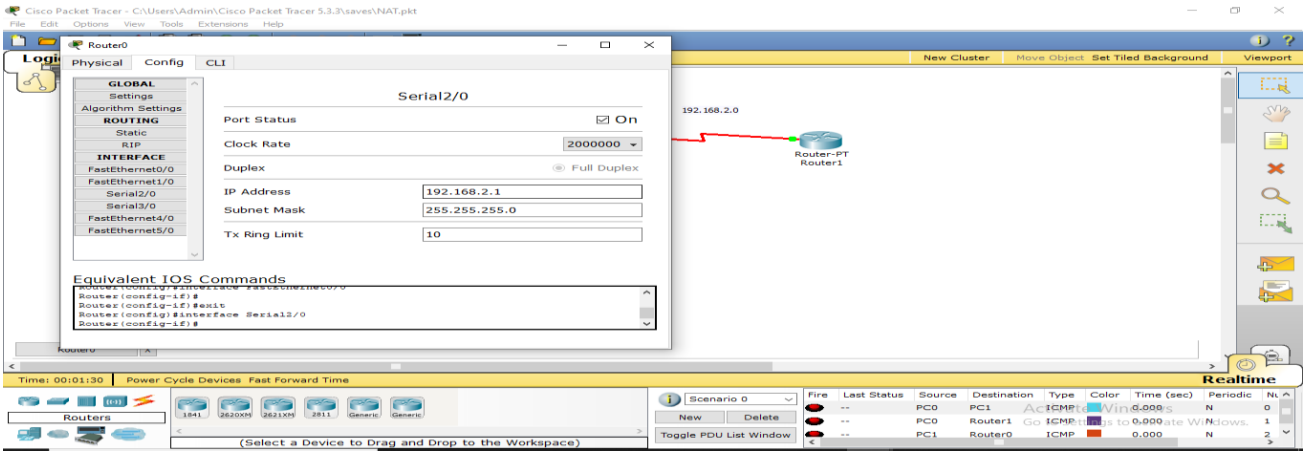


Step 2:Configure IP address to router1.

For Fastethernet 0/0

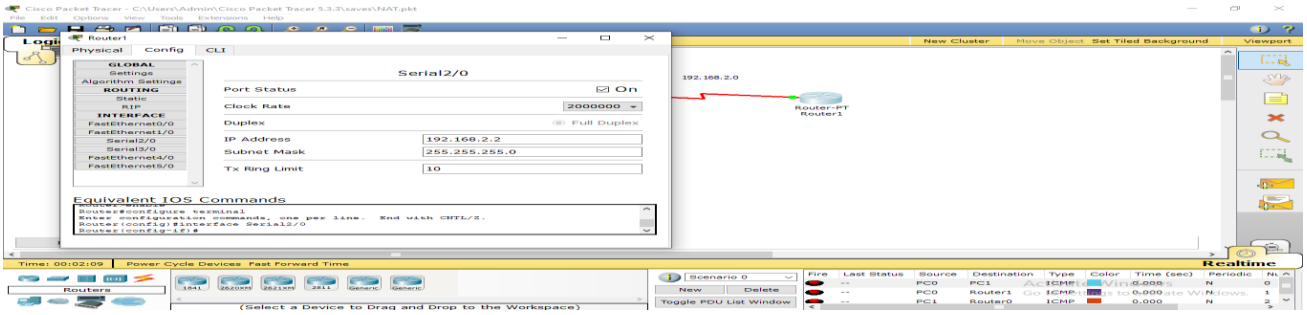


For Serial2/0



Step 3 :Configure IP address to router2

For Serial 2/0



Step 4:To set up Static NAT:

Router# sh ip nat translation

Router# config t

Router(config)#ip nat inside source static 10.0.0.2 192.168.1.3

Provide interface for NAT cable

Router(config)# int fa0/0

Router(config-if)# ip nat inside

exit

Router(config)# int serial2/0

Router(config-if)# ip nat outside

exit

exit

Router#sh ip nat translation

Step 5 :To Check Connectivity between two network

Click on any PC>click on desktop>select command prompt and then type below commands

PC>ping 192.168.2.1

Pinging 192.168.2.1 with 32 bytes of data:

Reply from 192.168.2.1: bytes=32 time=147ms TTL=254

Reply from 192.168.2.1: bytes=32 time=84ms TTL=254

Reply from 192.168.2.1: bytes=32 time=100ms TTL=254

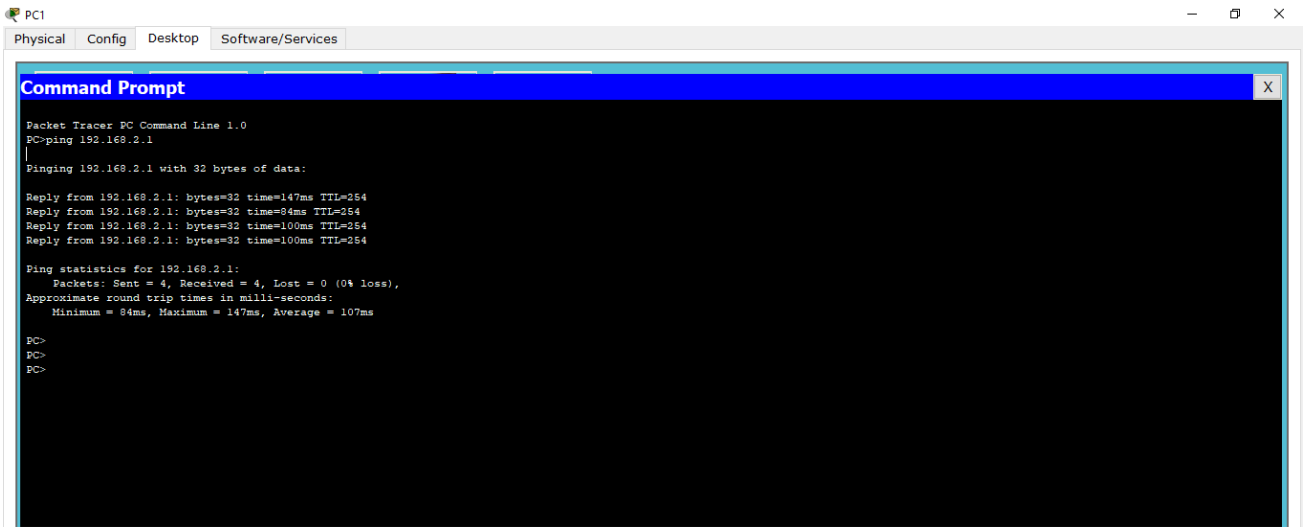
Reply from 192.168.2.1: bytes=32 time=100ms TTL=254

Ping statistics for 192.168.2.1:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 84ms, Maximum = 147ms, Average = 107ms



Exp 9: Configure the Dynamic NAT using cisco packet tracer.

NAT(NETWORK ADDRESS TRANSLATION)

It is a Process in which one or more Local ip address is translated into global ip address or vice versa in order to provide internet access to the host.

It Allows multiple devices to access internet through single public ip address.

Dynamic NAT configuration requires four steps: -

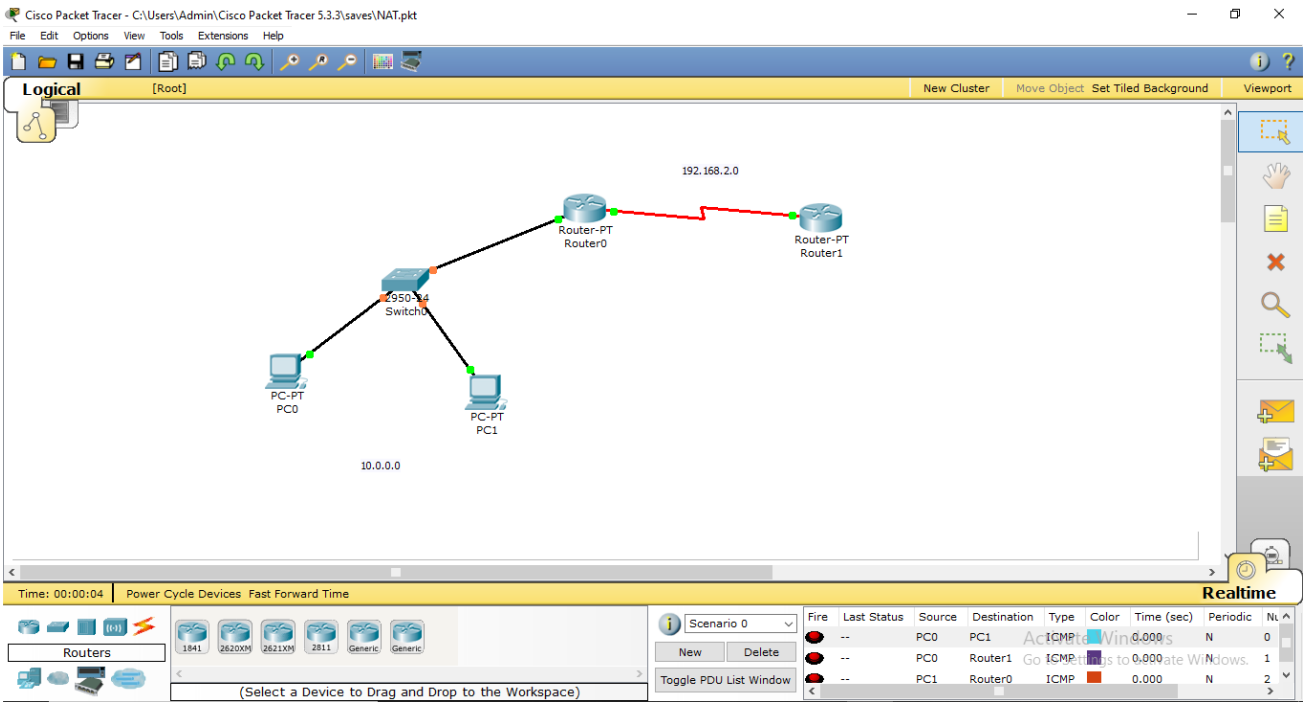
Step1: Create an access list of IP addresses which need translation(ACL range 1 to 99 and 1300 to 1999)

Step2 : Create a pool of all IP address which are available for translation

Step3: Map access list with pool

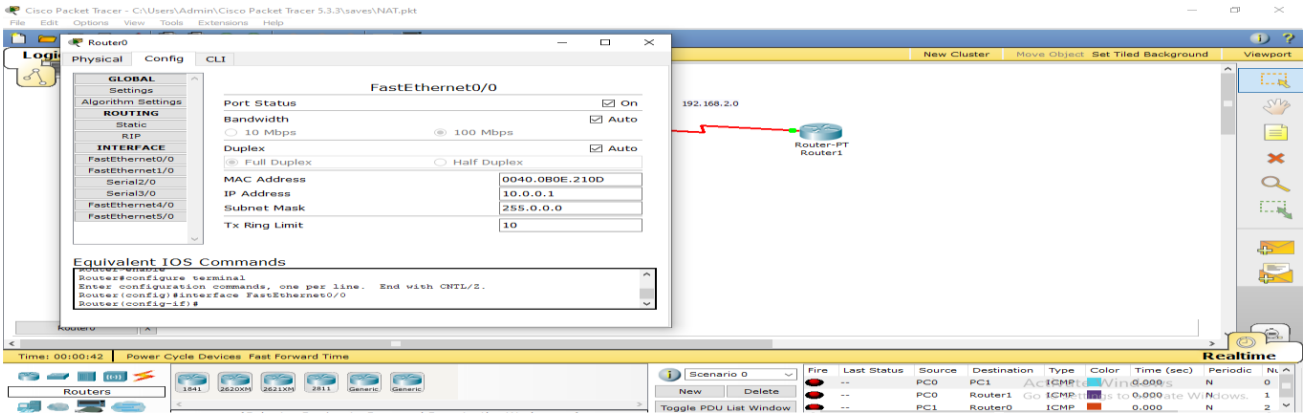
Stpe4: Define inside and outside interfaces

Step 1: Draw a topology as shown below and assign IP address to all PC's.

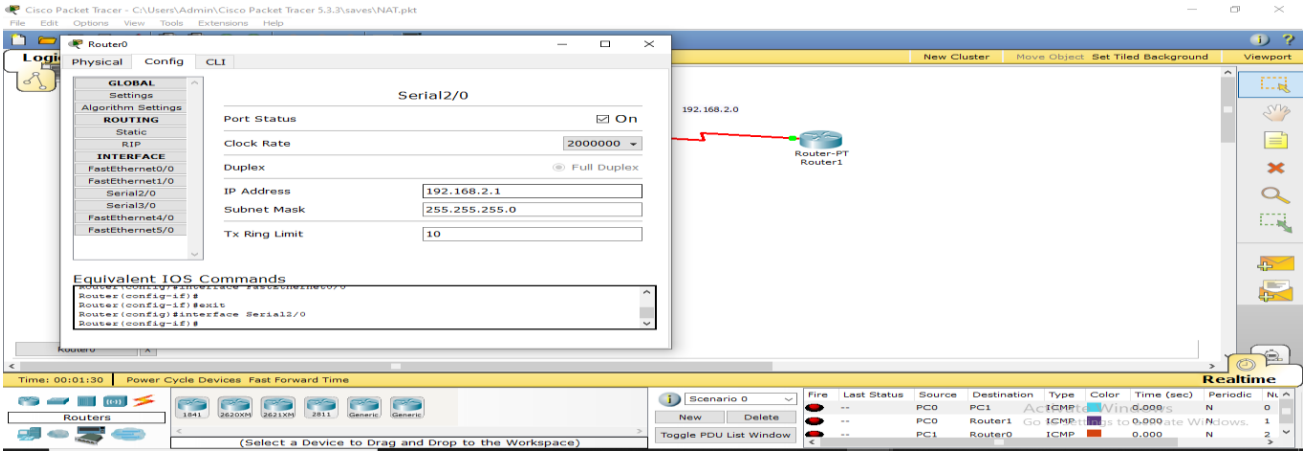


Step 2:Configure IP address to router1.

For Fastethernet 0/0

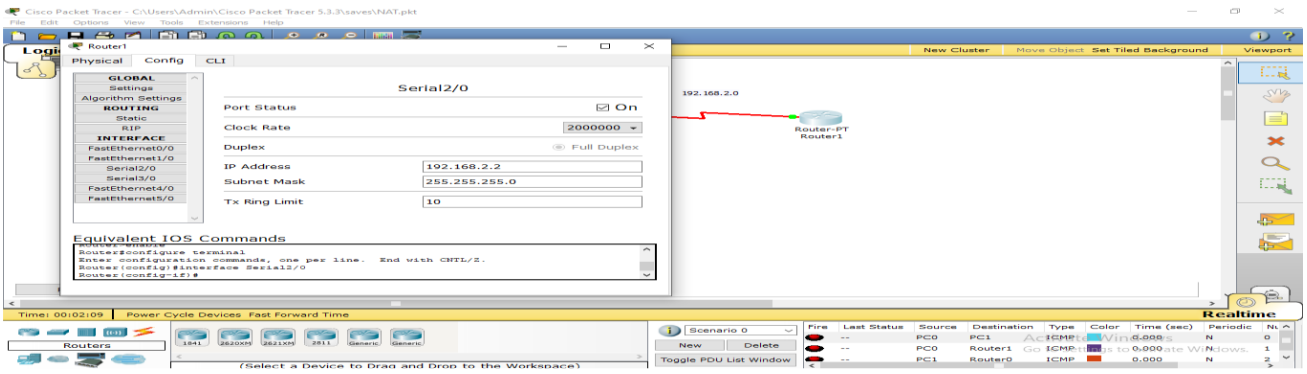


For Serial2/0



Step 3 :Configure IP address to router2

For Serial 2/0



Step 4:To set up Dynamic NAT:

router(config)#access-list 1 permit 10.0.0.2 0.0.0.0

router(config)#access-list 1 permit 10.0.0.3 0.0.0.0

router(config)#ip nat pool nslab 192.168.2.3 192.168.2.4 netmask 255.255.255.0

router(config)#ip nat inside source list 1 pool nslab

router(config)#int fa0/0

router(config-if)#ip nat inside

router(config-if)#exit

router(config)#int serial2/0

router(config-if)ip nat outside

router#sh ip nat translation

Step 5 :To Check Connectivity between two network

Click on any PC>click on desktop>select command prompt and then type below commands

PC>ping 192.168.2.1

Pinging 192.168.2.1 with 32 bytes of data:

Reply from 192.168.2.1: bytes=32 time=147ms TTL=254

Reply from 192.168.2.1: bytes=32 time=84ms TTL=254

Reply from 192.168.2.1: bytes=32 time=100ms TTL=254

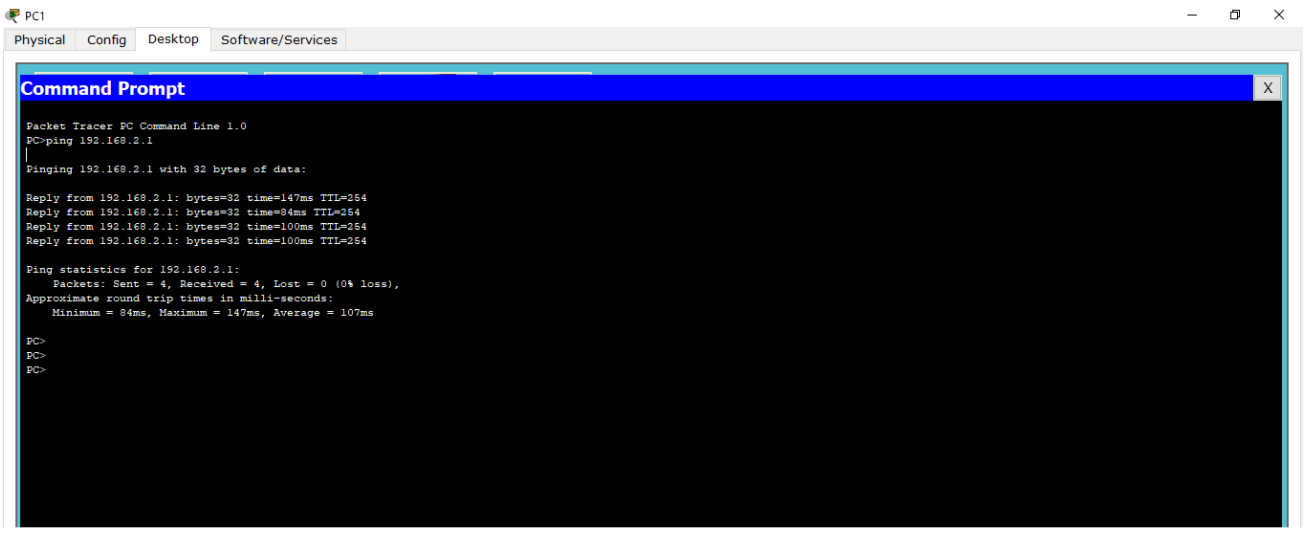
Reply from 192.168.2.1: bytes=32 time=100ms TTL=254

Ping statistics for 192.168.2.1:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

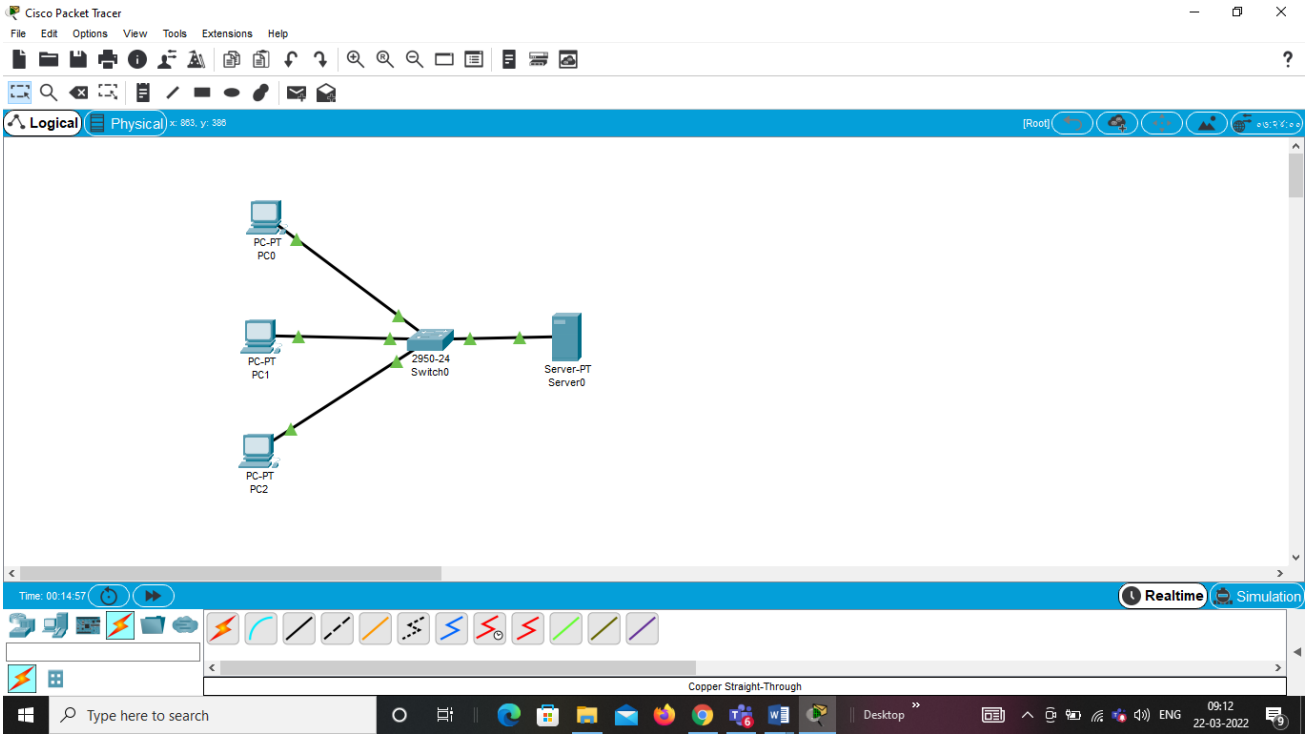
Approximate round trip times in milli-seconds:

Minimum = 84ms, Maximum = 147ms, Average = 107ms

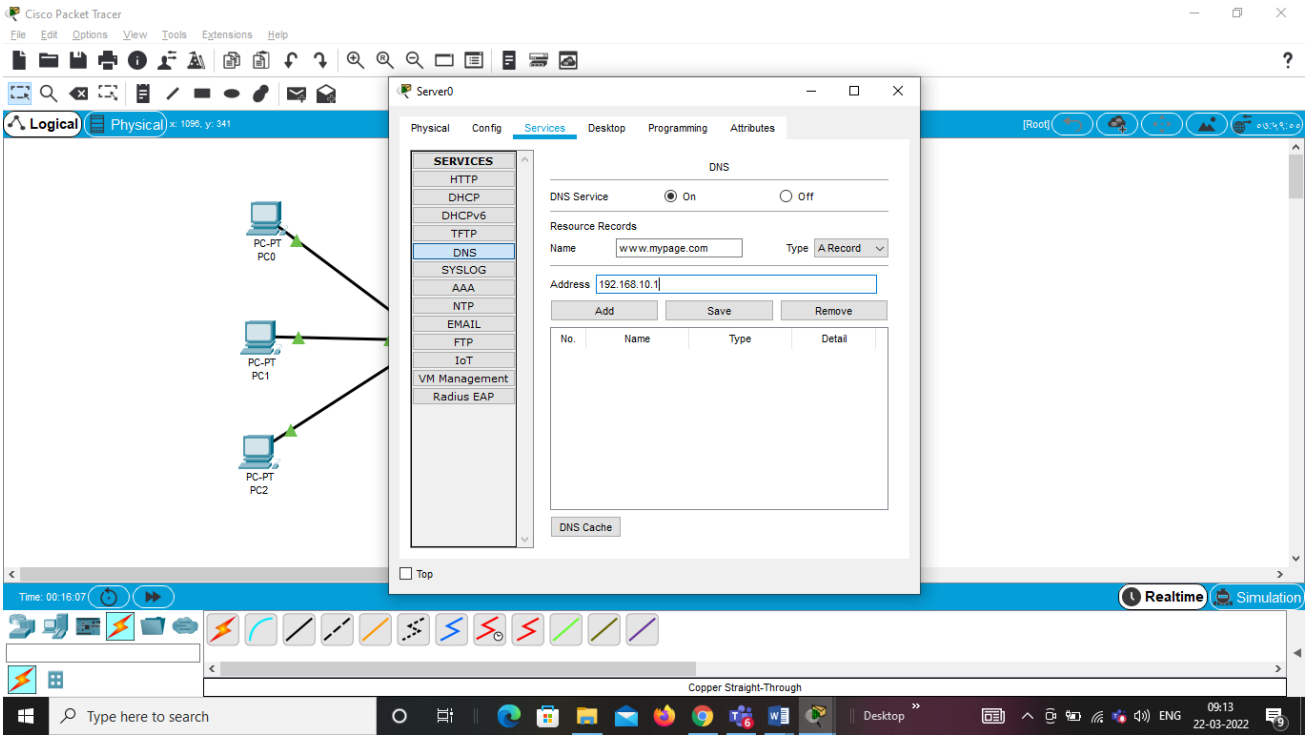


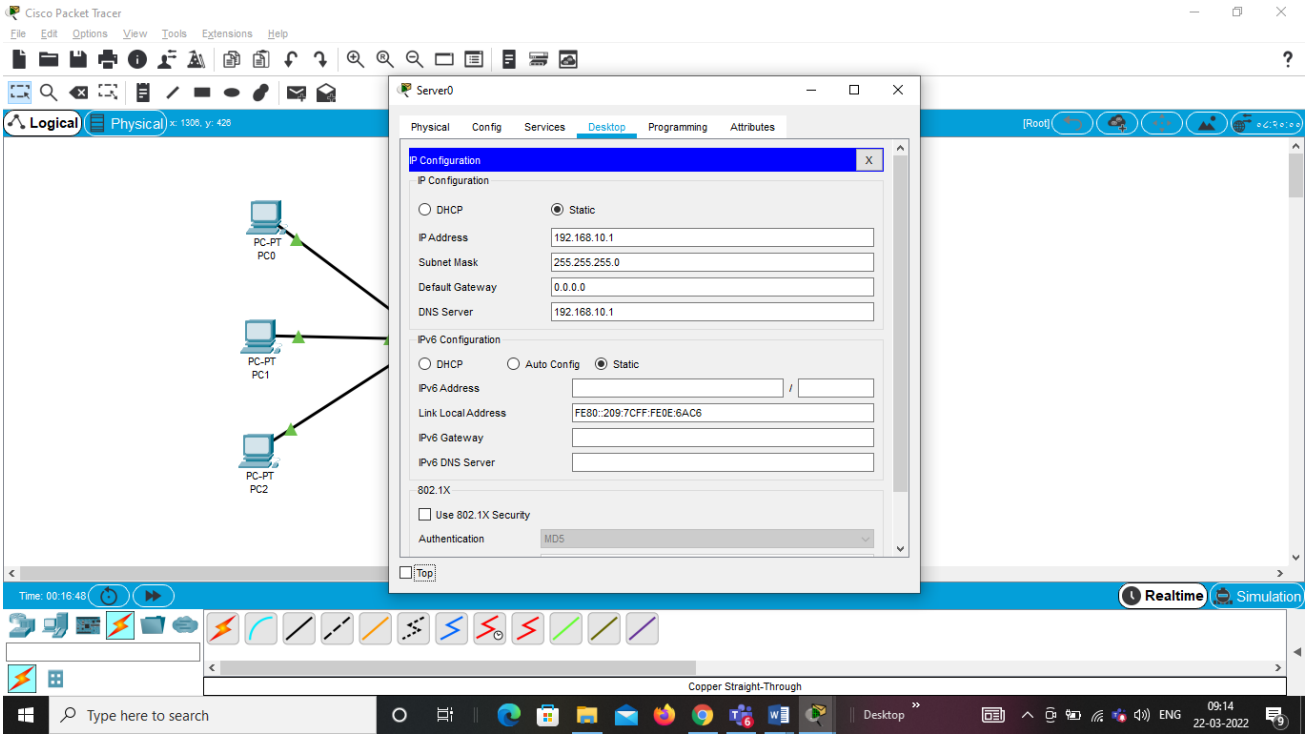
Exp 10: Configure the DNS Server using cisco packet tracer.

Step 1: Draw a topology as shown below and assign IP address to all PC's

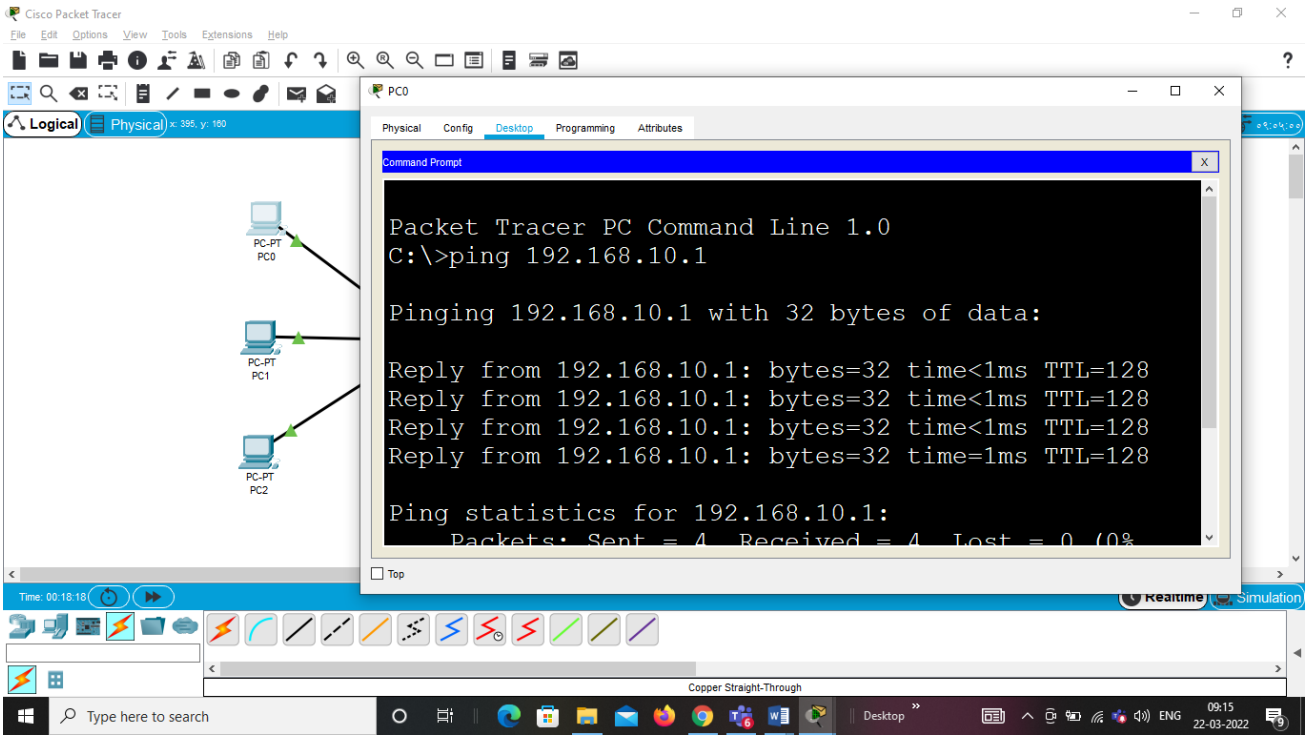


Step 2: Configure the server with DNS service and assign the server IP address.

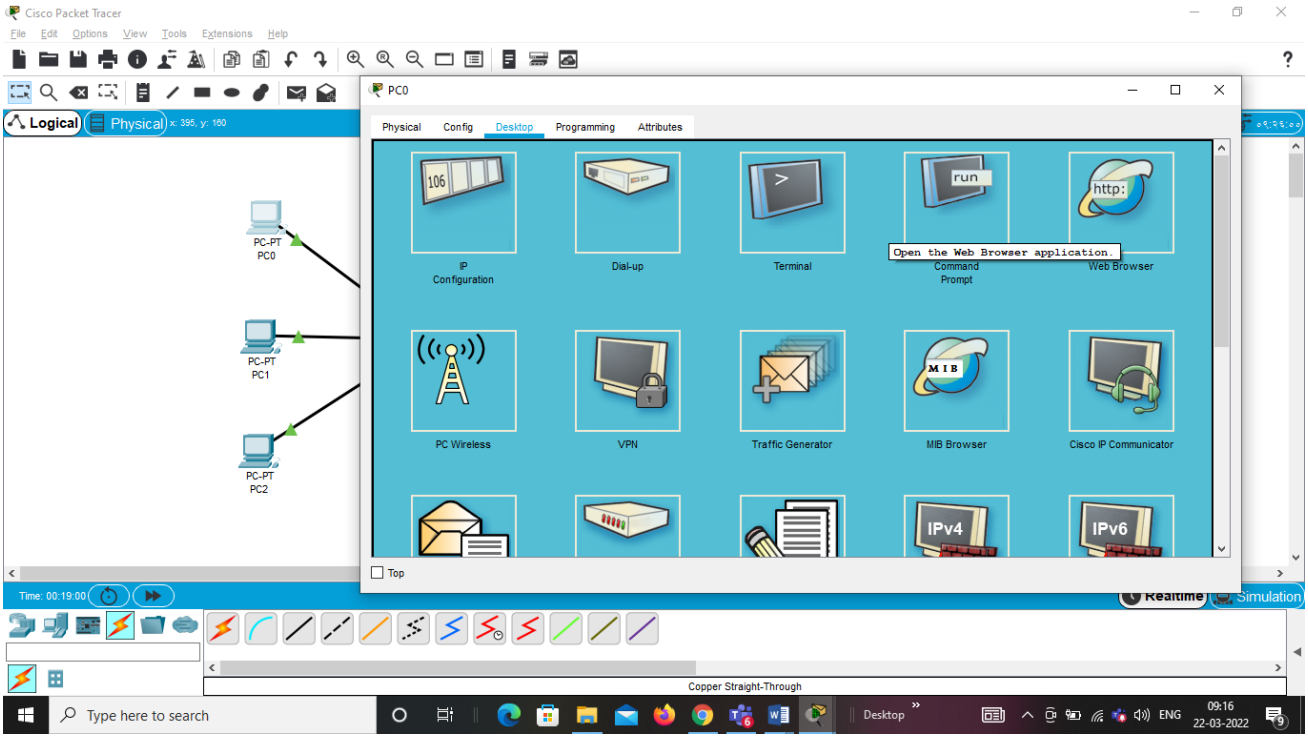




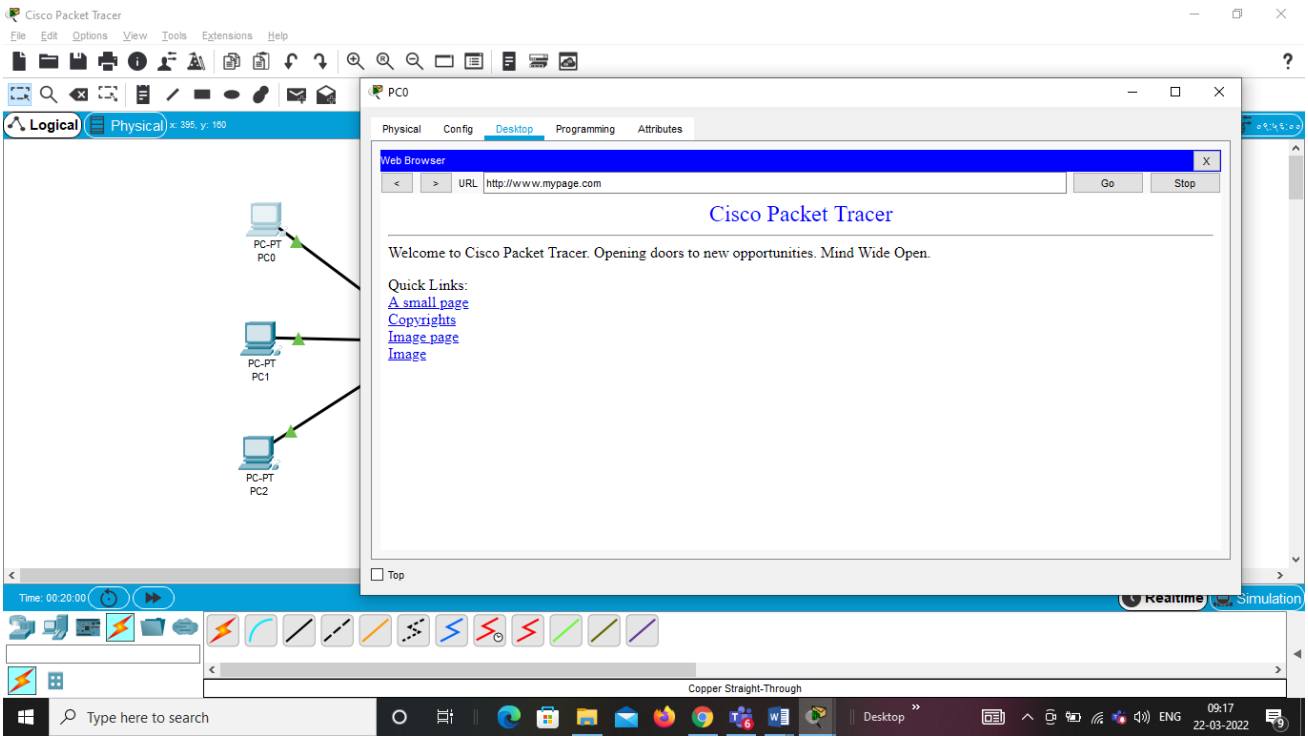
Step 3: Check the connectivity using ping



Step 3: Check the connectivity from web browser

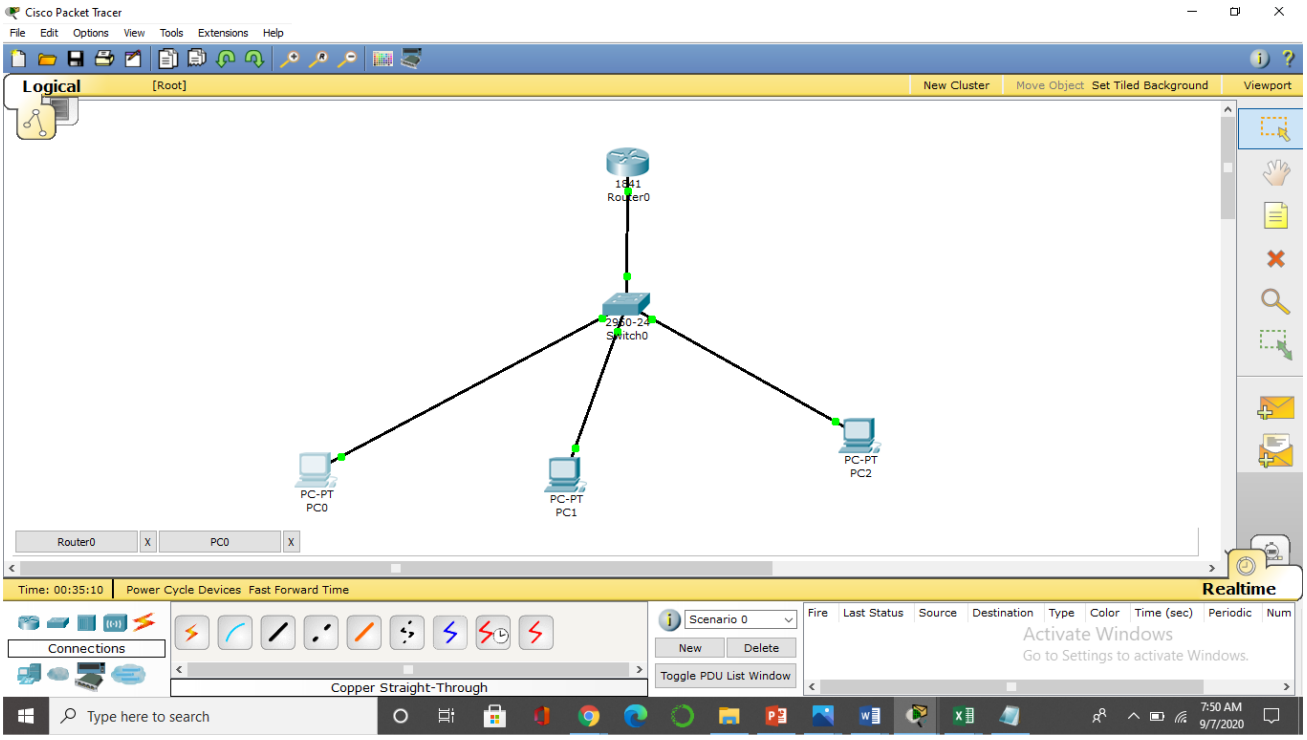


And give URL of the DNS server and click go.



Exp 8: Configure the telnet protocol using cisco packet tracer

Step 1: Draw a topology as shown below and assign IP address to all PC's.



Step 2:Configure IP address to router.

Router(config)#interface FastEthernet0/0

Router(config-if)#ip address 10.0.0.1 255.0.0.0

Router(config-if)#no shutdown

Router(config-if)#exit

Step 3:To set privilege mode password

Click on Router and go to CLI tab and type below.

Router(config)#enable password 1234

Router(config)#exit

Step 4:To configure telnet.

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#line vty 0 4

Router(config-line)#password cisco

Router(config-line)#login

Router(config-line)#exit

Step 5:To check telnet configuration.

Router#sh run

Building configuration...

Current configuration : 556 bytes

!

version 12.4

no service timestamps log datetime msec

no service timestamps debug datetime msec

no service password-encryption

!

hostname Router

!

!

!

enable password 1234

!

!

ODD SEM 2020-21 PRESIDENCY UNIVERSITY, BENGALURU

!

!

!

!

!

!

!

!

!

!

```
spanning-tree mode pvst
!
!
!
!
interface FastEthernet0/0
ip address 10.0.0.1 255.0.0.0
duplex auto
speed auto
!
interface FastEthernet0/1
no ip address
duplex auto
speed auto
shutdown
!
interface Vlan1
no ip address
shutdown
!
ip classless
!
!
!
!
!
!
!
line con 0
```

line vty 0 4

password cisco

login

line vty 5 6

password cisco

login

!

!

!

End

Step 6:To access cisco router via telnet connection from any PC.

Click on any PC>click on desktop>select command prompt and then type below commands

PC>ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32 time=13ms TTL=255

Reply from 10.0.0.1: bytes=32 time=16ms TTL=255

Reply from 10.0.0.1: bytes=32 time=16ms TTL=255

Reply from 10.0.0.1: bytes=32 time=16ms TTL=255

Ping statistics for 10.0.0.1:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

Minimum = 13ms, Maximum = 16ms, Average = 15ms

PC>telnet 10.0.0.1

Trying 10.0.0.1 ...Open

User Access Verification

Password:

Router>en

Password:

Router#

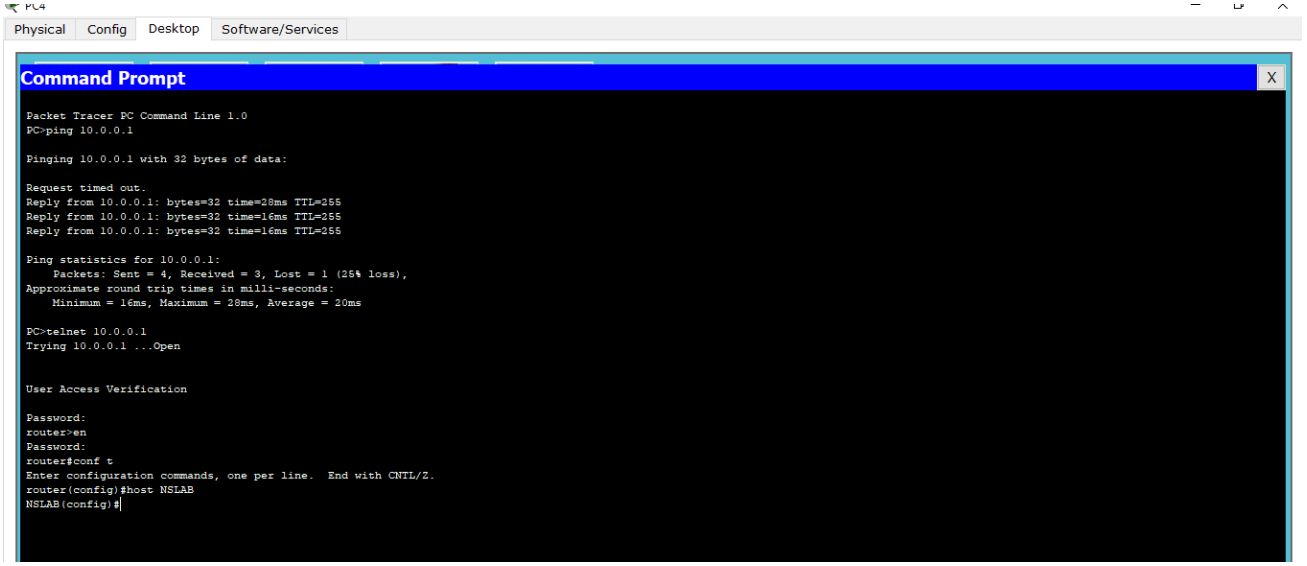
We can change hostname of router in PC command prompt.

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#host nslab

nslab(config)#



Module-3

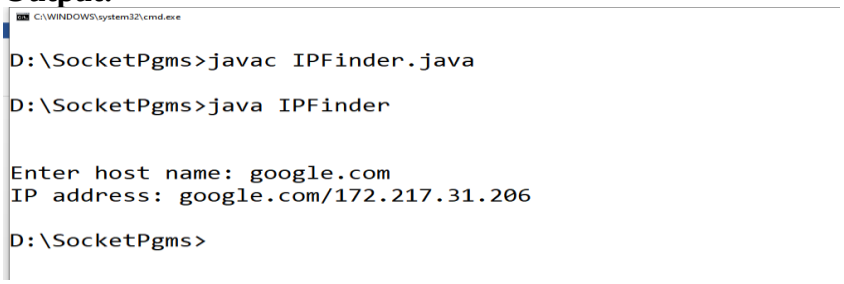
Socket Programming

Exp 1: To find the website address using socket programming

Solution:

```
import java.net.*;
import java.util.*;
public class IPFinder
{
	public static void main(String[] args)
	{
		String host;
		Scanner input = new Scanner(System.in);
		System.out.print("\n\nEnter host name: ");
		host = input.next();
		try
		{
			InetAddress address = InetAddress.getByName(host);
			System.out.println("IP address: " + address.toString());
		}
		catch (UnknownHostException uhEx)
		{
			System.out.println("Could not find " + host);
		}
	}
}
```

Output:

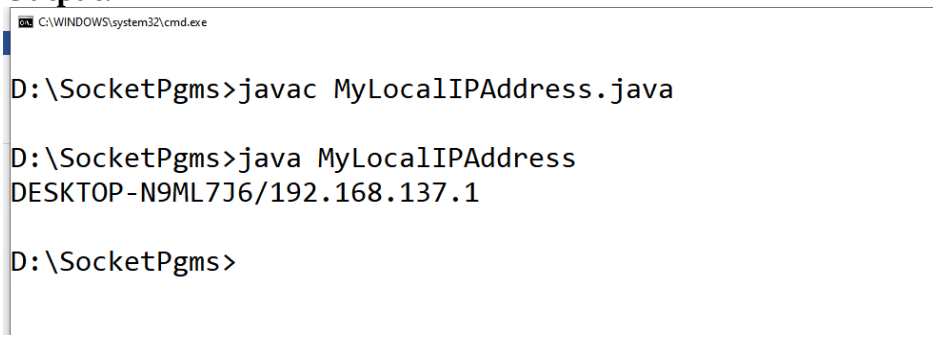


Exp 2: To find the local host IP address using socket programming

Solution:

```
import java.net.*;
public class MyLocalIPAddress
{
	public static void main(String[] args)
	{
		try
		{
			InetAddress address =InetAddress.getLocalHost();
			System.out.println(address);
		}
		catch (UnknownHostException uhEx)
		{
			System.out.println("Could not find local address!");
		}
	}
}
```

Output:



Exp 3: Write a program to communicate between client and server using UDP Protocol

Solution:

Client Side:

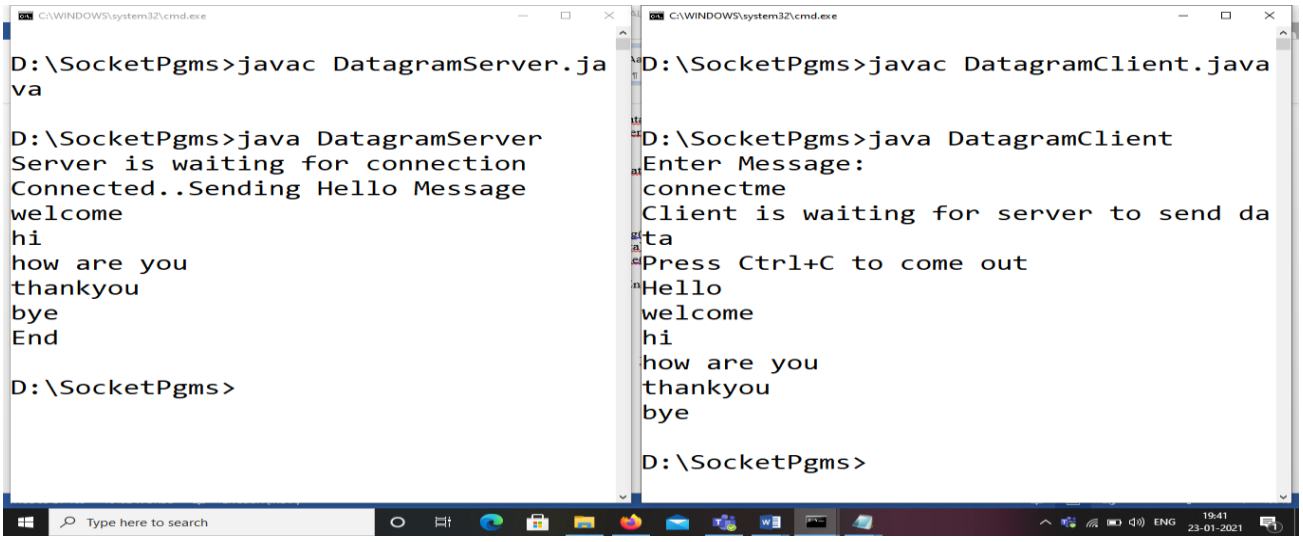
```
import java.net.*;
import java.io.*;
class DatagramClient
{
	public static DatagramSocket ds;
	public static byte buffer[]=new byte[1024];
	public static int clientport=1789, serverport=1790;
	public static void main(String args[]) throws Exception
	{
		byte buffer[]=new byte[1024];
		ds=new DatagramSocket(clientport);
		BufferedReader breader=new BufferedReader(new InputStreamReader(System.in));
		System.out.println("Enter Message:");
		String msg=breader.readLine();
		buffer=msg.getBytes();
		ds.send(new DatagramPacket(buffer, msg.length(),
InetAddress.getLocalHost(),serverport));
		System.out.println("Client is waiting for server to send data");
		System.out.println("Press Ctrl+C to come out");
		while(true)
		{
			DatagramPacket dp=new DatagramPacket(buffer, buffer.length);
			ds.receive(dp);
			String pdata=new String(dp.getData(),0, dp.getLength());
			if(pdata.equals("End"))
				break;
			System.out.println(pdata);
			String str=breader.readLine();
			buffer=str.getBytes();
			ds.send(new DatagramPacket(buffer, str.length(),
InetAddress.getLocalHost(),serverport));
		}
	}
}
```

Server Side:

```
import java.net.*;
import java.io.*;
class DatagramServer
{
	public static DatagramSocket ds;
	public static int clientport=1789, serverport=1790;
```

```
        public static void main(String args[]) throws Exception
        {
                byte buffer[]=new byte[1024];
                ds=new DatagramSocket(serverport);
                BufferedReader breader=new BufferedReader(new InputStreamReader(System.in));
                System.out.println("Server is waiting for connection");
                DatagramPacket dp=new DatagramPacket(buffer, buffer.length);
                ds.receive(dp);
                String pdata=new String(dp.getData(),0, dp.getLength());
                System.out.println("Connected..Sending Hello Message");
                String st="Hello";
                buffer=st.getBytes();
                ds.send(new DatagramPacket(buffer, st.length(),
InetAddress.getLocalHost(),clientport));
                while(true)
                {
                        ds.receive(dp);
                        String recvdata=new String(dp.getData(),0, dp.getLength());
                        System.out.println(recvdata);
                        String str=breader.readLine();
                        buffer=str.getBytes();
                        if(str==null || str.equals("End"))
                        {
                                ds.send(new DatagramPacket(buffer, str.length(),
InetAddress.getLocalHost(),clientport));
                                break;
                        }
                        ds.send(new DatagramPacket(buffer, str.length(),
InetAddress.getLocalHost(),clientport));
                }
        }
}
```

Output:



Exp 4: Write a program to communicate between client and server using TCP Protocol

Solution:

Client Side:

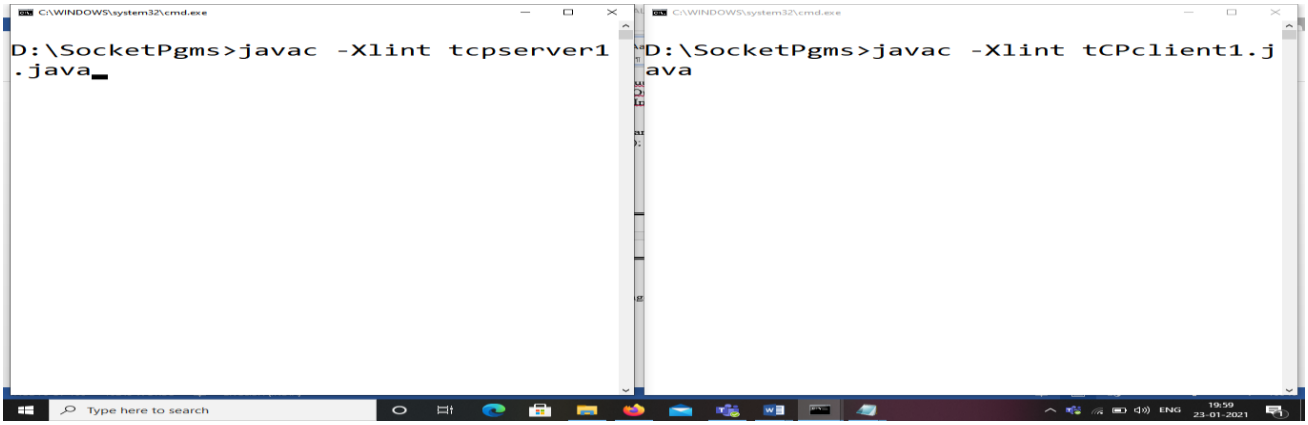
```
import java.net.*;
import java.io.*;
public class tCPclient1
{
	public static void main(String args[])throws IOException
	{
		Socket s=new Socket("localhost",55);
		DataInputStream in=new DataInputStream(s.getInputStream());
		DataOutputStream out=new DataOutputStream(s.getOutputStream());
		DataInputStream sysin=new DataInputStream(System.in);
		while(true)
		{
			String str=in.readLine();
			System.out.println("Message from server"+str);
			if(str.equals("End"))
				break;
			System.out.println("Enter reply message");
			String line=sysin.readLine();
			out.writeBytes(line+"\n");
		}
		s.close();
	}
}
```

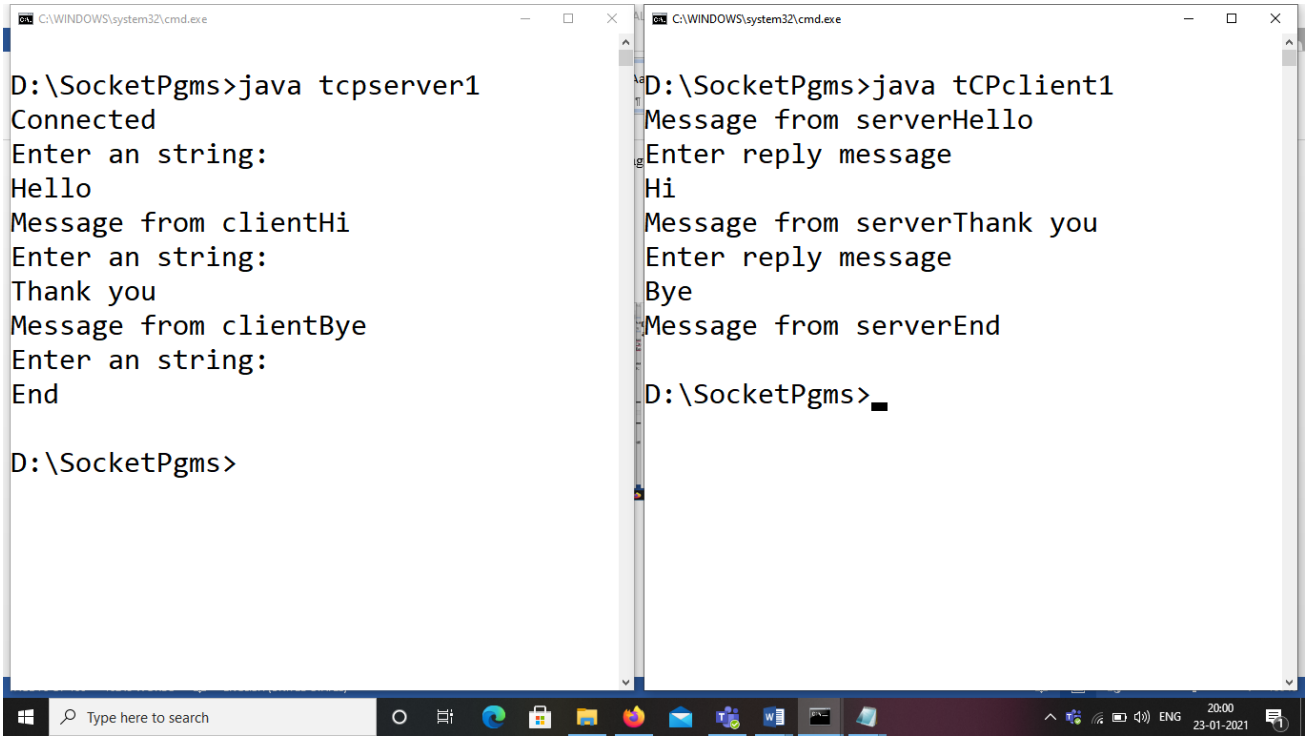
Server Side:

```
import java.io.*;
import java.net.*;
class tcpserver1
{
	public static void main(String args[ ])throws IOException
	{
		ServerSocket ss=new ServerSocket(55);
		Socket s=ss.accept();
		System.out.println("Connected");
		DataInputStream in=new DataInputStream(s.getInputStream());
		DataOutputStream out=new DataOutputStream(s.getOutputStream());
		DataInputStream sysin=new DataInputStream(System.in);
		while(true)
		{
			System.out.println("Enter an string:");
			String str=sysin.readLine();
			out.writeBytes(str+"\n");
			if(str.equals("End"))
				break;
```

```
                System.out.println("Message from client"+in.readLine());
            }
            ss.close();
        }
}
```

Output:





Exp 5: Write a program to check the connectivity for the given host name using socket programming.

Solution:

```
import java.net.*;
import java.io.*;
import java.util.*;
public class PingDemo
{
	public static void main(String[] args)throws IOException
	{
		String host="";
		Scanner input = new Scanner(System.in);
		System.out.print("\n\nEnter host name: ");
		host = input.next();
		try
		{
			InetAddress address = InetAddress.getByName(host);
			System.out.println("IP address: " + address.toString());
			System.out.println("Sending Ping Request to " + host);
			if (address.isReachable(5000))
				System.out.println(host + " is reachable.");
			else
				System.out.println(host + " NOT reachable.");
		}
		catch (UnknownHostException uhEx)
		{
			System.out.println("Could not find " + host);
		}
	}
}
```

Output:

Exp 6: Write a program to implement the ARP protocol using Socket programming

Solution:

Client Side:

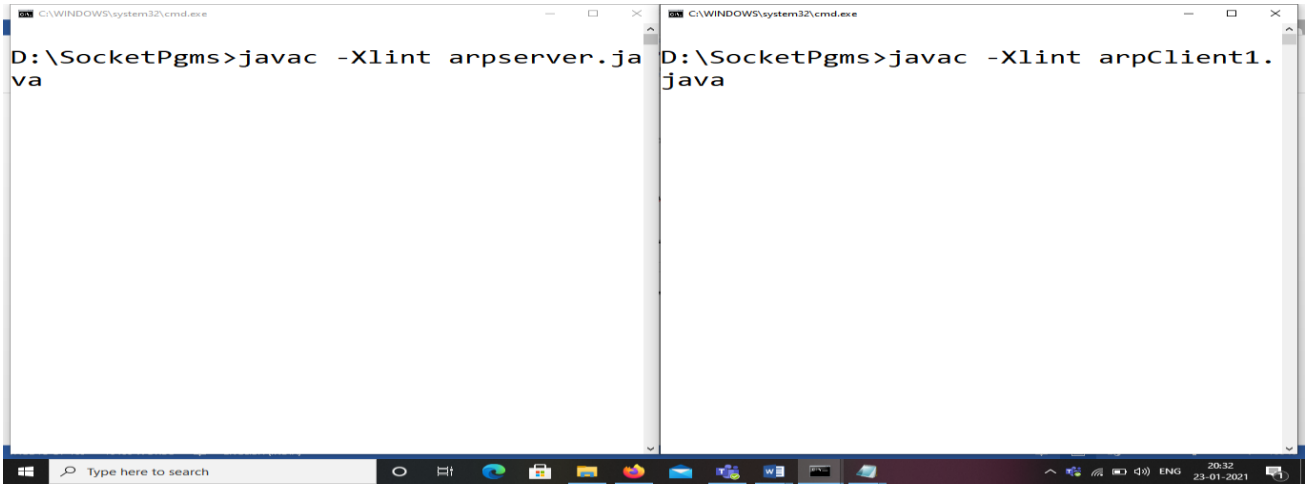
```
import java.io.*;
import java.net.*;
class arpClient1
{
	public static void main(String args[])throws IOException
	{
		Socket s=new Socket("localhost",55);
		DataInputStream in=new DataInputStream(s.getInputStream());
		DataOutputStream out=new DataOutputStream(s.getOutputStream());
		DataInputStream sysin=new DataInputStream(System.in);
		System.out.println("Enter an IP Address:");
		String str=sysin.readLine();
		out.writeBytes(str+"\n");
		System.out.println("The corresponding MAC address"+in.readLine());
	}
}
```

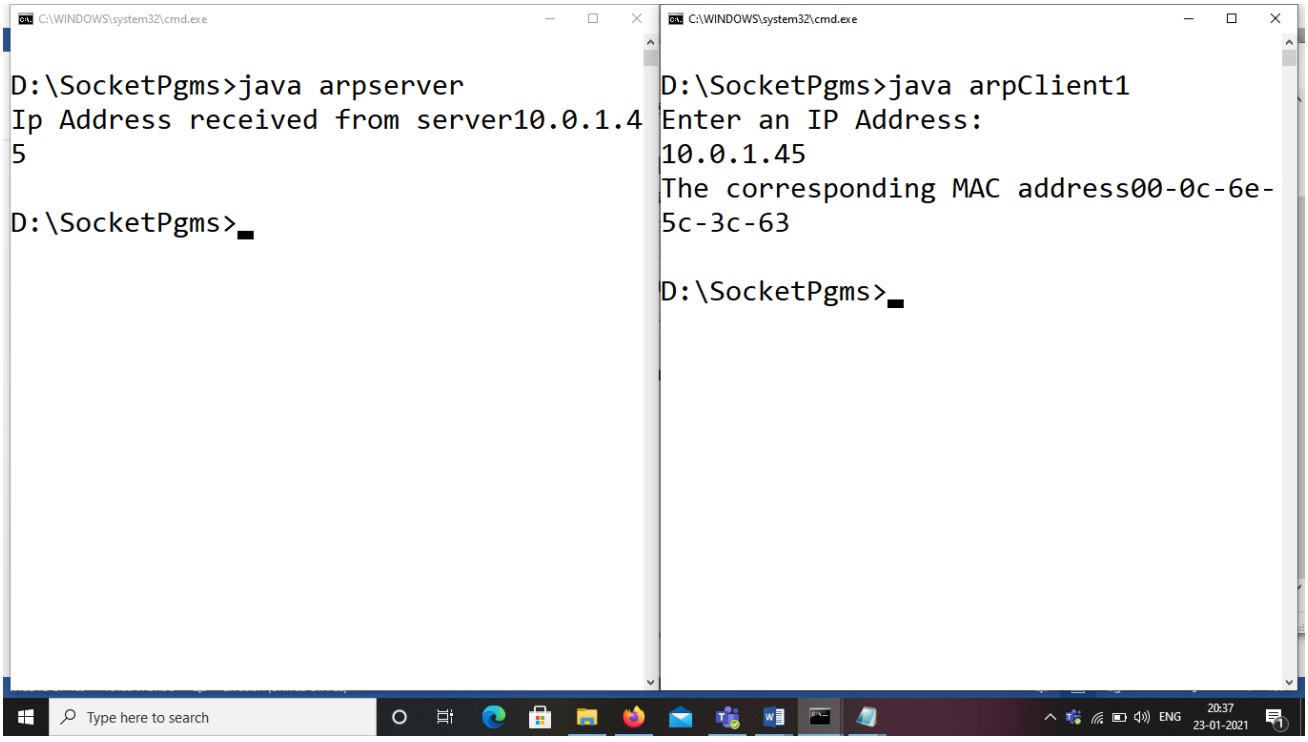
Server Side:

```
import java.net.*;
import java.io.*;
public class arpserver
{
	public static void main(String args[])throws IOException
	{
		ServerSocket ss=new ServerSocket(55);
		Socket s=ss.accept();
		DataInputStream in=new DataInputStream(s.getInputStream());
		DataOutputStream out=new DataOutputStream(s.getOutputStream());
		String iparr[]={"10.0.1.45","172.16.5.21","172.16..5.22"};
		String macarr[]={"00-0c-6e-5c-3c-63","02-11-B6-F3-EF-21","03-12-B3-F3-EF-18"};
		String str=in.readLine();
		System.out.println("Ip Address received from server"+str);
		int flag=0;
		for(int i=0;i<3;i++)
		{
			if(str.equals(iparr[i])==true)
			{
				flag=1;
				String str1=macarr[i];
				out.writeBytes(str1+"\n");
				break;
			}
		}
		if(flag==0)
```

```
                    System.out.println("Given IPAddress is not in the network");
                s.close();
        }
}
```

Output:





Exp 7: Write a program to implement the FTP protocol using Socket programming

Solution:

Client Side:

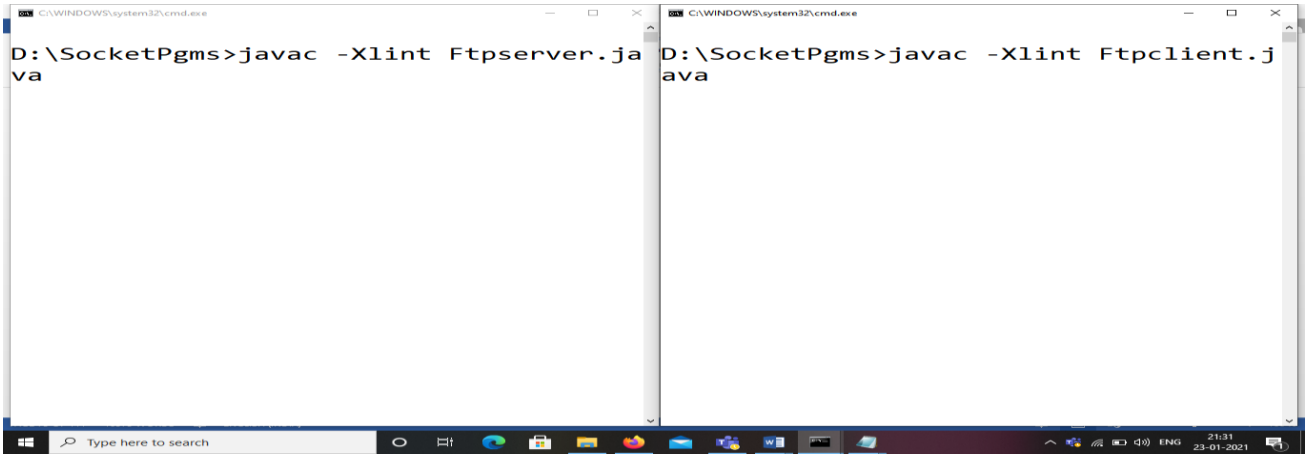
```
import java.io.*;
import java.net.*;
public class Ftpclient
{
	public static void main(String a[])throws IOException
	{
		Socket s=new Socket(InetAddress.getLocalHost(),5555);
		DataInputStream s1=new DataInputStream(s.getInputStream());
		DataInputStream inp=new DataInputStream(System.in);
		DataOutputStream so=new DataOutputStream(s.getOutputStream());
		System.out.println("\n enter the filename(path)");
		String str=inp.readLine();
		so.writeBytes(str+"\n");
		FileOutputStream fos=new FileOutputStream("output.txt");
		int str1;
		while((str1=s1.read())!=-1)
			fos.write((char)str1);
		System.out.println("\n file received successfully");
		s1.close();
		so.close();
		inp.close();
		s.close();
	}
}
```

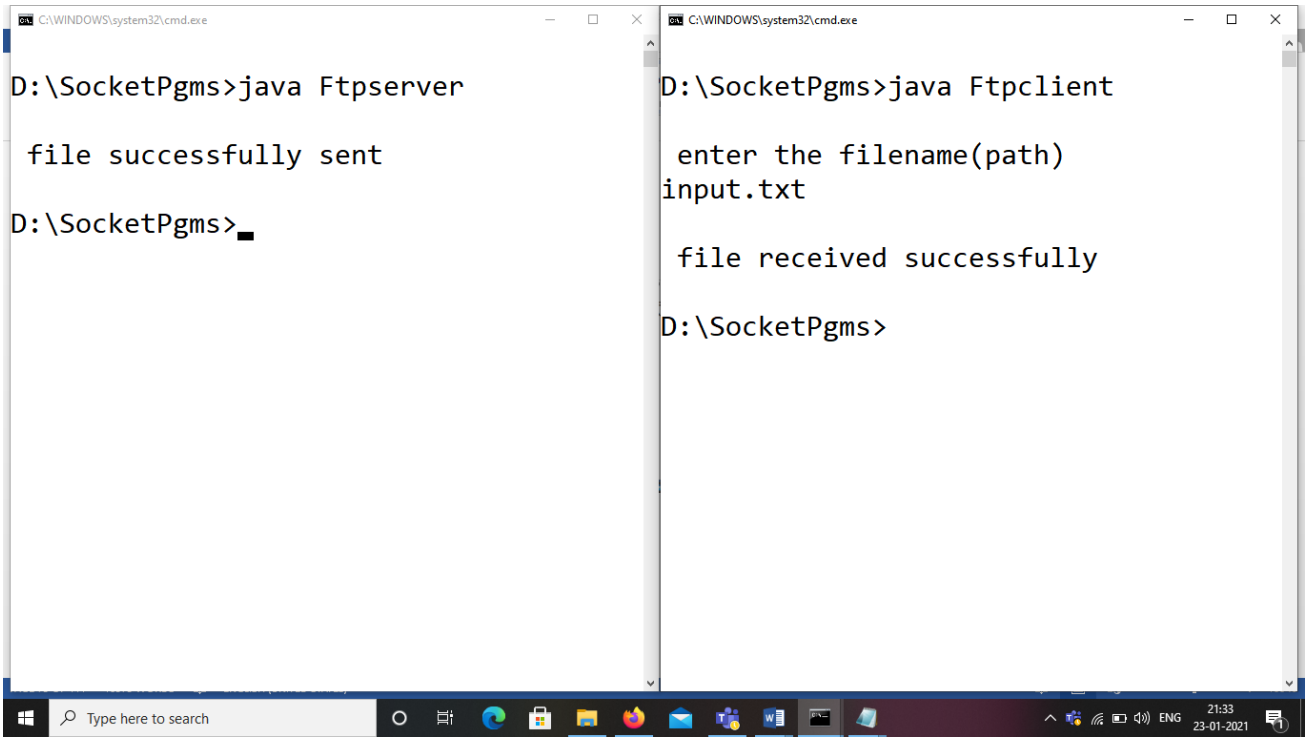
Server Side:

```
import java.io.*;
import java.net.*;
public class Ftpserver
{
	public static void main(String a[])throws IOException
	{
		ServerSocket ss=new ServerSocket(5555);
		Socket s=ss.accept();
		DataOutputStream dos=new DataOutputStream(s.getOutputStream());
		DataInputStream din=new DataInputStream(s.getInputStream());
		String s1;
		s1=din.readLine();
		FileInputStream fin=new FileInputStream(s1);
		int str1;
		while((str1=fin.read())!=-1)
			dos.writeBytes(""+(char)str1);
		System.out.println("\n file successfully sent");
		dos.close();
		din.close();
```

```
            s.close();
        }
}
```

Output:





Input file contents will be transferred into output file.

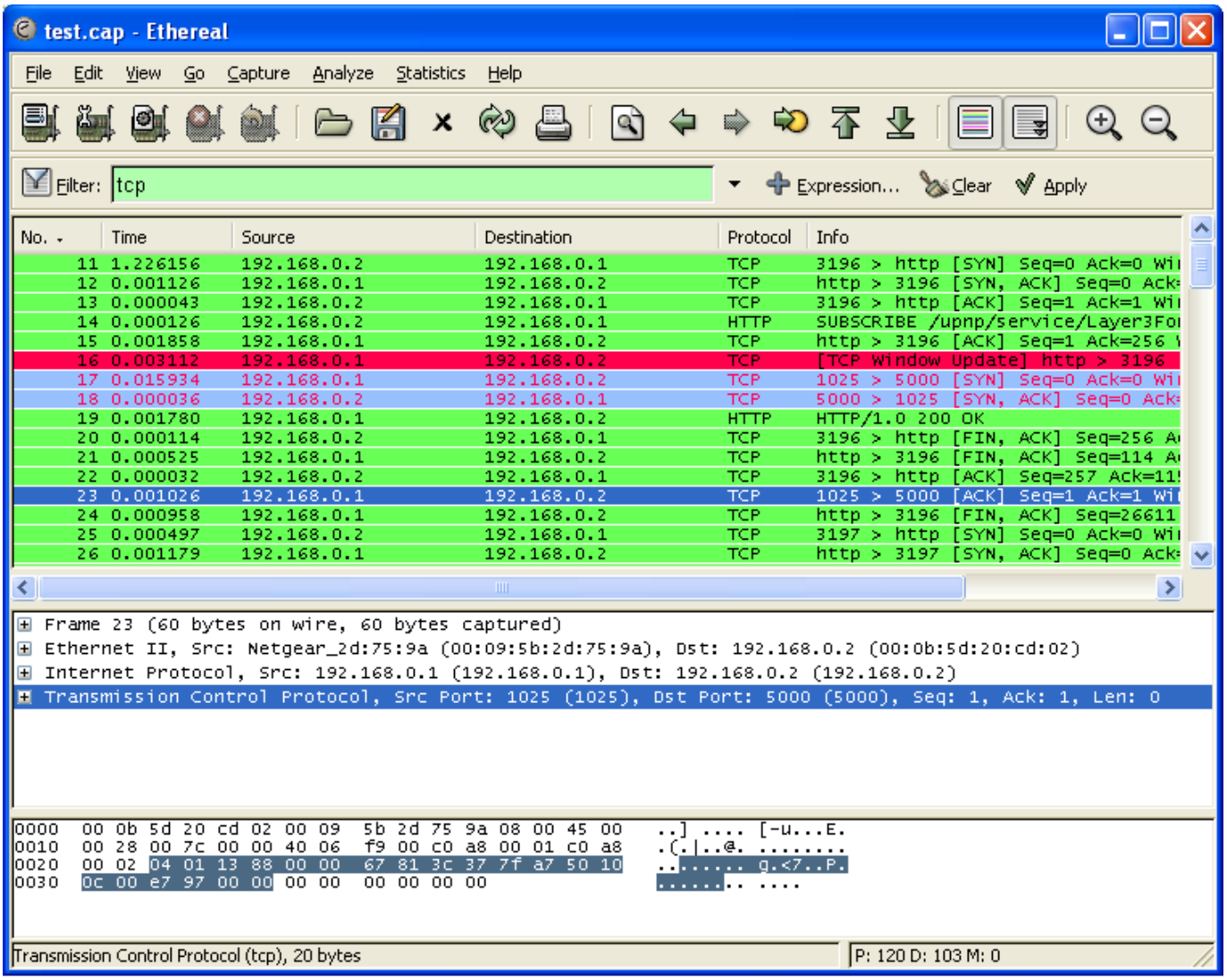
Wireshark tool

Introduction:

Wireshark, formerly known as Ethereal, is one of the most powerful tools in a network security analyst's toolkit. As a network packet analyzer, Wireshark can peer inside the network and examine the details of traffic at a variety of levels, ranging from connection-level information to the bits comprising a single packet. This flexibility and depth of inspection allows the valuable tool to analyze security events and troubleshoot network security device issues.

Installation:

You can find the source code at www.wireshark.org, download and install on your computer



Filter: field in the filter toolbar of the Wireshark window and press enter to initiate the filter.

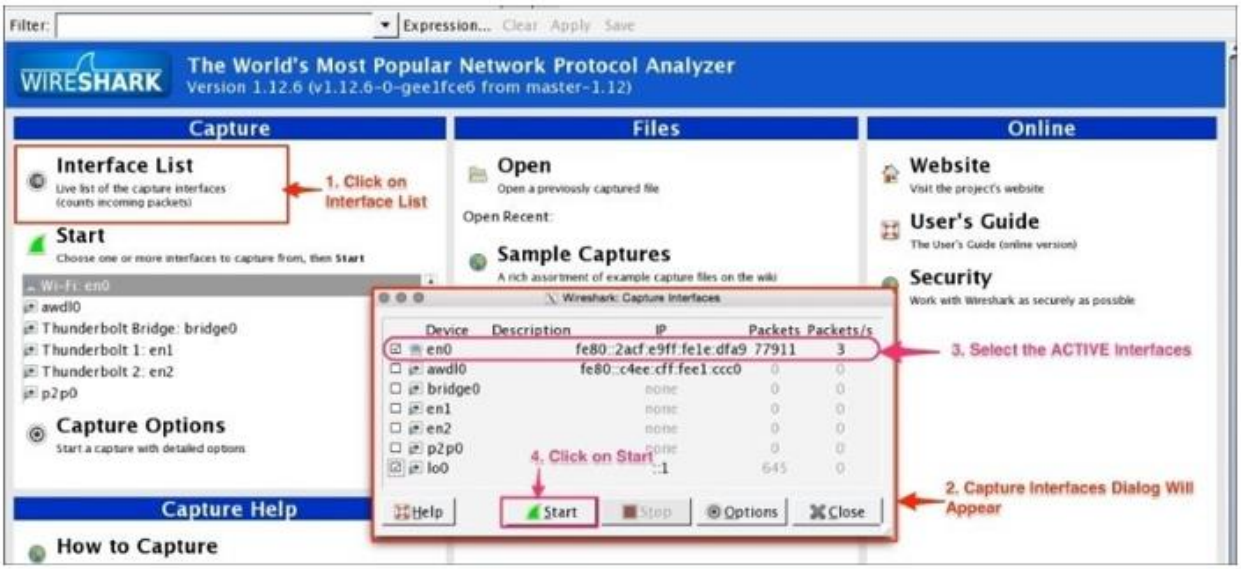
Wireshark has the following cool built-in features, few of them are listed as follows:

-

- Available in both UNIX and Windows
- Ability to capture live packets from various types of interface
- Filters packets with many criteria
- Ability to decode larger sets of protocols
- Can save and merge captured packets
- Can create various statistics

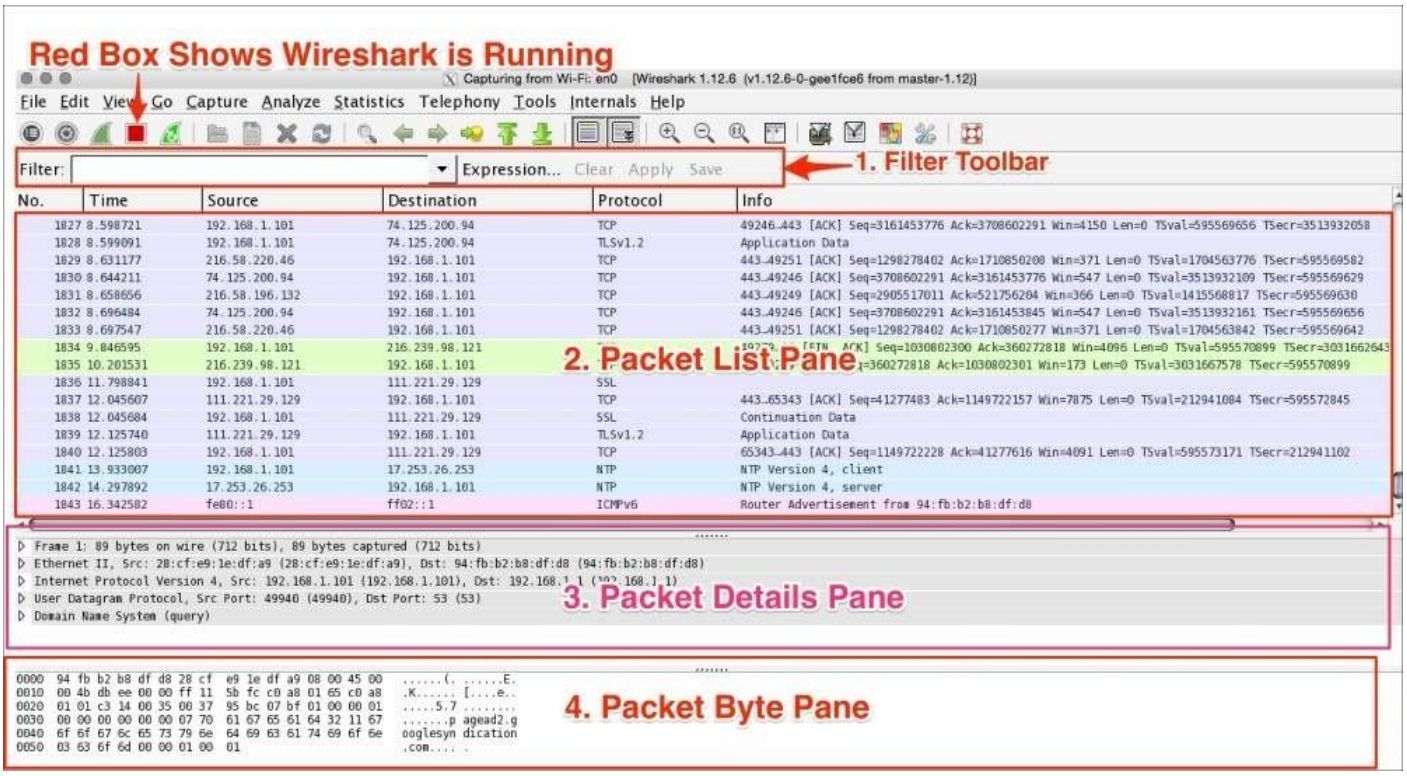
Capturing packets with Interface Lists

Click on **Interface List**; Wireshark will show a list of available network interfaces in the system and which one is active, by showing packets going in and out of the Interface, as shown in the following screenshot:



Wireshark user interface

The Wireshark main window appears when Wireshark starts capturing a packet, or when a .pcapfile is open for offline viewing. It looks similar to the following screenshot:



The Wireshark UI interface consists of different panes and provides various options to the user for customizing it. In this chapter, we will cover these panes indetail:

Item	**What is it?**
The red box	This shows that Wireshark is running and capturing a packet
1	This is the **Filter** toolbar, used for filtering packets based on the applied filter
2	This is the Packet List pane, which displays all captured packets
3	This is the Packet Details pane, which shows the selected packet in a verbose form
4	This is the Packet Byte pane, which shows the selected packet in a hex dump format

First, just observe pane **2** in the screen; the displayed packets appear with different colors. This is one of Wireshark's best features; it colors packets according to the set filter and helps you visualize the packet you are looking for.

To manage (view, edit, or create) a coloring rule, go to **View** | **Coloring Rules**. Wireshark will display the **Coloring Rules** dialog box, as shown in the screenshot:

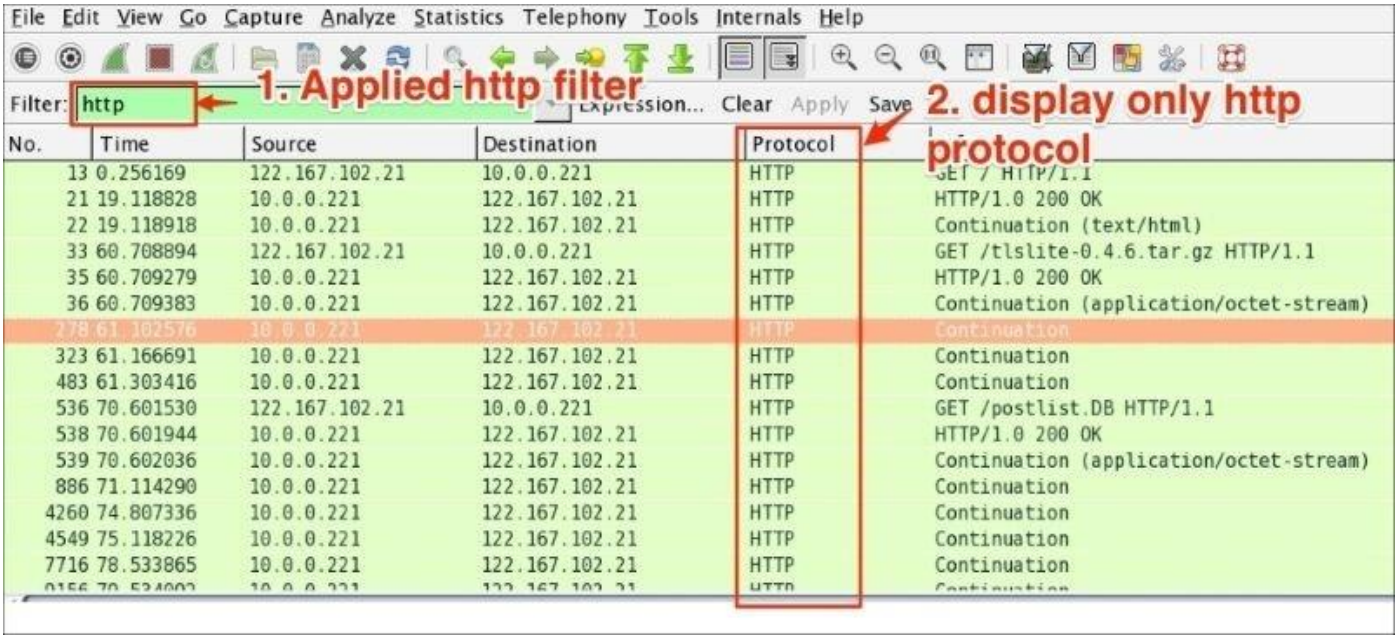
The Filter toolbar

The Wireshark display filter displays packets with its available coloring options. Wireshark display filters are used to change the view of a capture file by providing the full dissection of all packets, which helps analyzing a network tracefile efficiently. For example, if a user is interested in only HTTP packets, the user can set the display filter to http, as shown in the next screenshot.

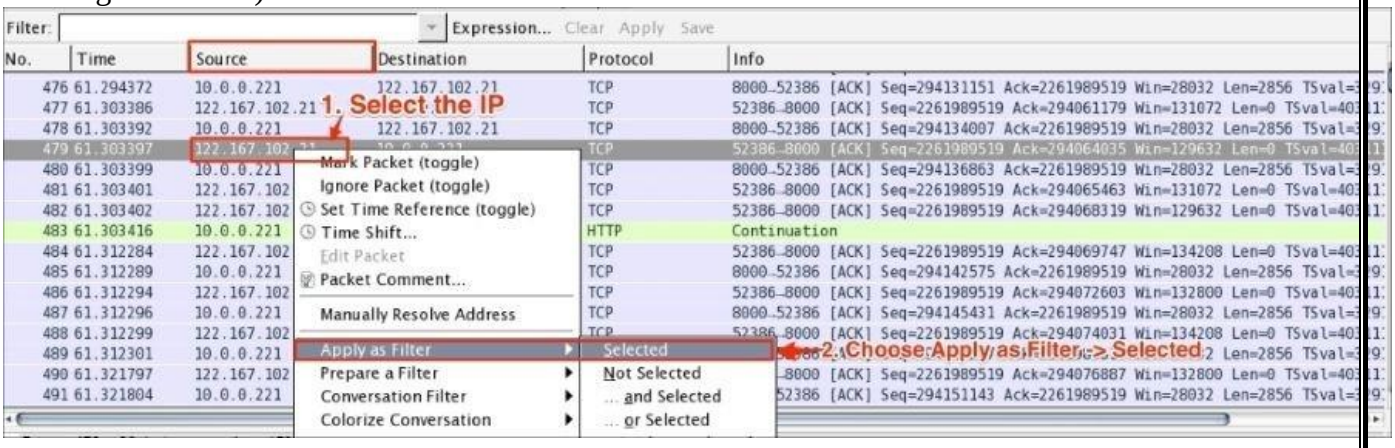
The steps to apply display filters are as follows:

1. Open the http_01.pcapfile.
2. Type the httpprotocol in the filter area and click on **Apply**.

Once the filter is applied, the Packet List pane will display only HTTP protocol-related packets:



Wireshark display filter can be applied or prepared from the column displayed in the Packet List pane by selecting the column, then right-clicking and going to **Apply as Filter** | **Selected** (as shown in the following screenshot) to create the filter from the source IP address 122.167.102.21:



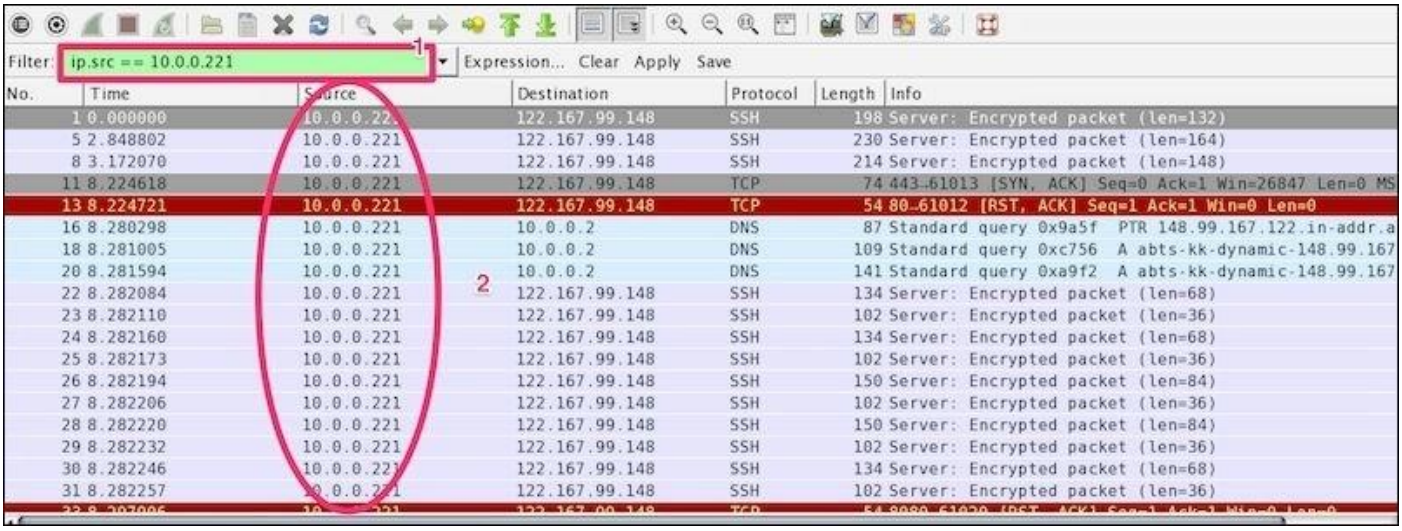
Wireshark provides the flexibility to apply filters from the Details pane; the steps remain the same.

Wireshark also provides the option to clear the filter. To do this click on **Clear** (available in the **Filter** toolbar) to display the entire captured packet.

Filtering techniques

Capturing and displaying packets properly will help you with packet captures. For example, to track a packet exchanged between two hosts: HOSTA(10.0.0.221) and HOSTB (122.167.99.148), open the SampleCapture01.pcapfile and apply the filter ip.src ==

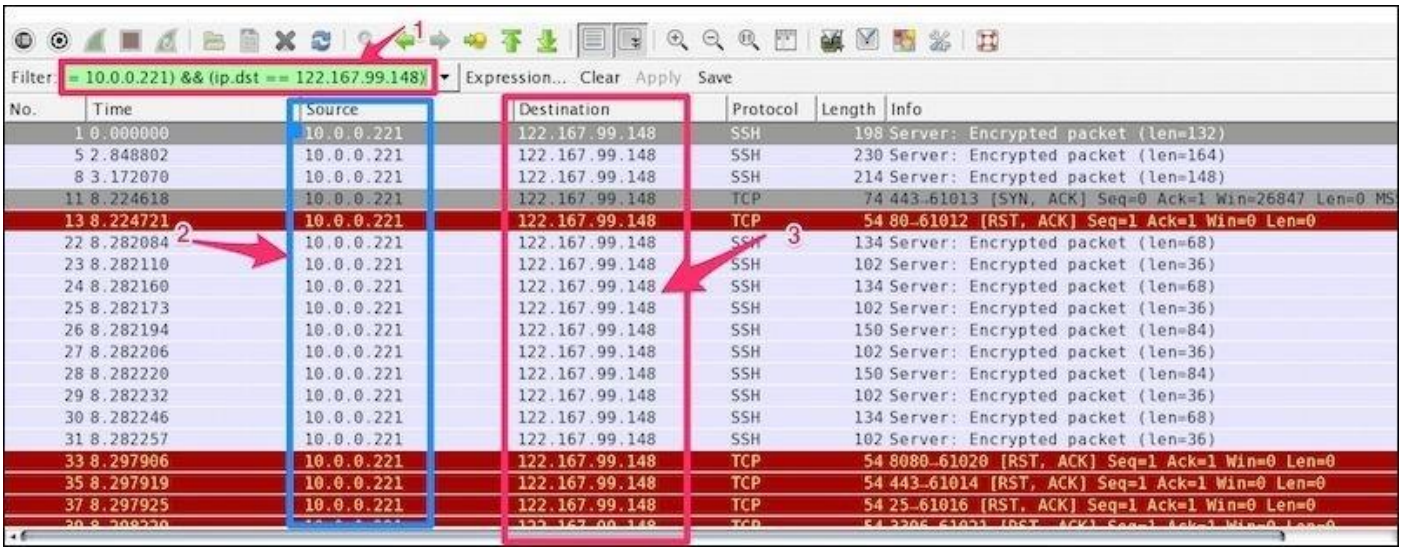
10.0.0.221as shown:



Let's see what the highlighted sections depict:

Item	Description
1	Apply filter ip.src == 10.0.0.221.
2	The Packet List pane displays the traffic from source to destination. The source shows the constant IP address 10.0.0.221. There is no evidence as to which packet is sent from host 122.167.99.148to host 10.0.0.221.

Now modify the filter (ip.src == 10.0.0.221) && (ip.dst == 122.167.99.148)to (ip.src == 10.0.0.221) or (ip.dst ==122.167.99.148). This will give the result shown in the following screenshot:

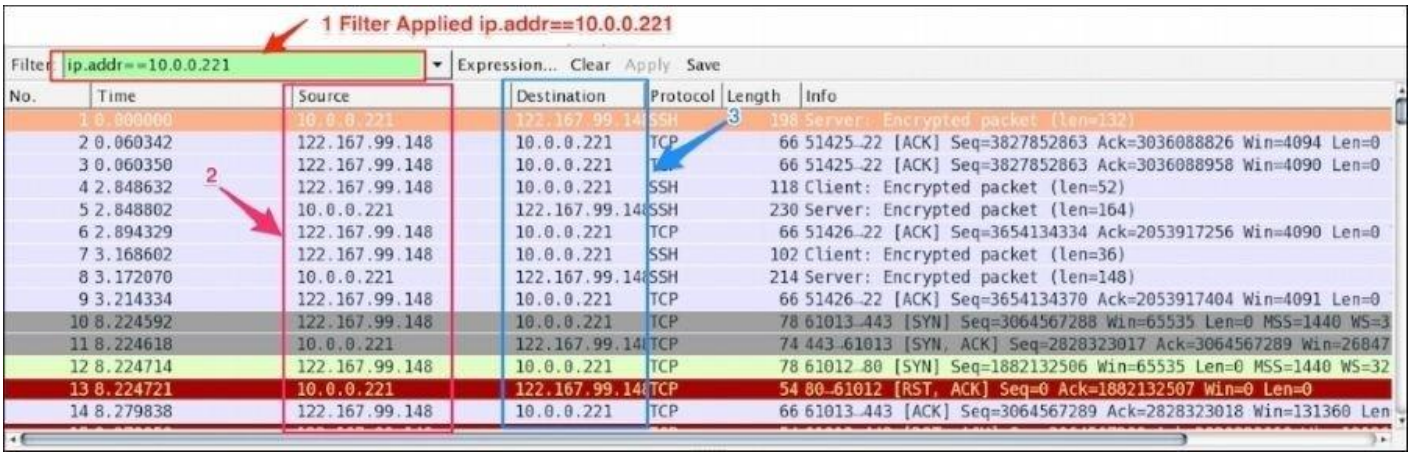


The highlighted sections in the preceding screenshot are explained as follows:

Item	Description
1	Applied filter (ip.src == 10.0.0.221) && (ip.dst == 122.167.99.148)
2	The source IP address (10.0.0.221) is not changed
3	The destination IP address (122.167.99.148) is not changed

Again the Packet List pane is not displaying the conversation between the two hosts.

Now modify the filter ip.addr == 122.167.99.148. The ip.addrfield will match the IP header for both the source and destination address and display the conversation between the hosts. Remember to choose the destination IP address as shown:



Let’s see what the highlighted sections depict:

Item	Description
1	Applied filter ip.addr == 122.167.99.148
2	The source IP is not constant; it shows the conversation between the two hosts
3	The destination IP is not constant; it shows the conversation between the two hosts

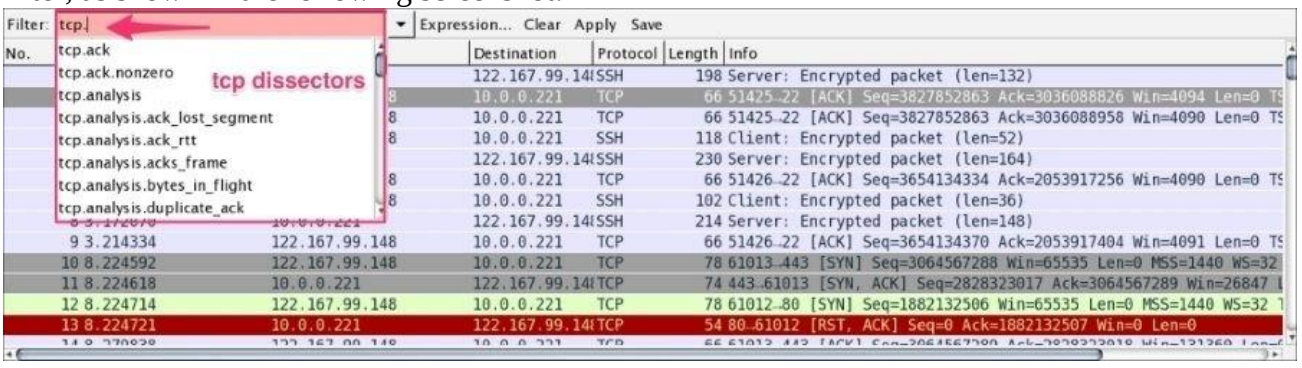
The same conversation is captured by choosing the destination MAC address using the display filter eth.addr == 06:73:7a:4c:2f:85.

Filter examples

Some common filter examples are as follows:

Filter/capture name	Filter value
Packet on a given port	tcp.port == 443
Packet on the source port	tcp.srcport=2222
SYN packet on port 443	(tcp.port == 443) && (tcp.flags == 0x0010)
The HTTP protocol	http
Based on the HTTP getmethod	http.request.method == "GET"
Using &&, tcp, and http	tcp && http
Checking the tcpwindow size	tcp.window_size <2000
No Arpused for normal traffic	!arp
The MAC address filter	eth.dst == 06:43:7b:4c:4f:85
Filter out TCP ACK	tcp.flags.ack==0
Check only RST and ACK packets	(tcp.flags.ack == 1) && (tcp.flags.reset == 1)
Filter all SNMP	Snmp
HTTP or DNS or SSL	http || dns | ssl

There is no need to memorize the filter; there is an easy way to apply it. The display filter Autocomplete feature lists all dissectors after the first period "." that have been added to the display filter, as shown in the following screenshot:



Display Filter comparison operators

eq	==	**Equal** ip.addr==10.0.0.5

ne	!=	**Not equal** ip.addr!=10.0.0.5
gt	>	**Greater than** frame.pkt_len > 10
lt	<	**Less than** frame.pkt_len < 128
ge	>=	**Greater than or equal to** frame.pkt_len ge 0x100
le	<=	**Less than or equal to** frame.pkt_len <= 0x20

Display Filter Logical Operations

English	C-like	Description and example
and	&&	**Logical AND** ip.addr==10.0.0.5 and tcp.flags.fin
or	||	**Logical OR** ip.addr==10.0.0.5 or ip.addr==192.1.1.1
xor	^^	**Logical XOR** tr.dst[0:3] == 0.6.29 xor tr.src[0:3] == 0.6.29

not	!	**Logical NOT** not llc

C-like syntax	Shortcut	Description	Example
==	eq	Equal	ip.addr == 192.168.1.1 or ip.addr eq 192.168.1.1
!=	ne	Not equal	!ip.addr==192.168.1.1 or ip.addr != 192.168.1.1 or ip.addr ne 192.168.1.1
>	gt	Greater than	frame.len > 64
<	lt	Less than	frame.len < 1500
>=	ge	Greater than or equal to	frame.len >= 64
<=	le	Less than or equal to	frame.len <= 1500
	Is present	A parameter is present	http.response
	contains	Contains a string	http.host contains cisco
	matches	A string matches the condition	http.host matches www.cisco.com

C-like syntax	Shortcut	Description	Example
&&	and	Logical AND	ip.src==10.0.0.1 and tcp.flags.syn==1 all SYN flags sent from IP address 10.0.0.1 practically—all connections opened (or tried to be opened) from 10.0.0.1
||	or	Logical OR	ip.addr==10.0.0.1 or ip.addr==10.0.02 all packets going in or out the two IP addresses
!	not	Logical NOT	not arp and not icmp all packets that are neither ARP nor ICMP packets

Address format	Syntax	Example
Ethernet (MAC) address	`eth.addr == xx:xx:xx:xx:xx:xx` where `x` is `0` to `f`	`eth.addr == 00:50:7f:cd:d5:38`
	`eth.addr == xx-xx-xx-xx-xx-xx` where `x` is `0` to `f`	`eth.addr == 00-50-7f-cd-d5-38`
	`eth.addr == xxxx.xxxx.xxxx` where `x` is `0` to `f`	`eth.addr == 0050.7fcd.d538`
Broadcast MAC address	`Eth.addr == ffff.ffff.ffff`	
IPv4 host address	`ip.addr == x.x.x.x` where `x` is `0` to `255`	`Ip.addr == 192.168.1.1`
IPv4 network address	`ip.addr == x.x.x.x/y` where `x` is `0` to `255`, `y` is `0` to `32`	`ip.addr == 192.168.200.0/24` (all addresses in the network `192.168.200.0` mask `255.255.255.0`)
IPv6 host address	`ipv6.addr == x:x:x:x:x:x:x:x` `ipv6.addr == x::x:x:x:x` where in the format of `nnnn`, `n` is `0` to `f` (hex)	`ipv6.addr == fe80::85ab:dc2e:ab12:e6c7`

ARP filters:

- Display only ARP requests:
  - `arp.opcode == 1`
- Display only ARP responses:
  - `arp.opcode == 2`

IP and ICMP filters:

- Display only packets from specific IP addresses:
  - `ip.src == 10.1.1.254`
- Display only packets that are not from a specific address:
  - `!ip.src == 64.23.1.1`
- Display only packets between two hosts:
  - `ip.addr == 192.168.1.1 and ip.addr == 200.1.1.1`
- Display only packets that are sent to multicast IP addresses:
  - `ip.dst == 224.0.0.0/4`
- Display only packets coming from network `192.168.1.0/24` (mask `255.255.255.0`):
  - `ip.src==192.168.1.0/24`
- Display only IPv6 packets to/from specific addresses:
  - `ipv6.addr == ::1`
  - `ipv6.addr == 2008:0:130F:0:0:09d0:666A:13ab`

Complex filters:

- Packets from network `10.0.0.0/24` to a website that contains the word `packt`:
  - `ip.src == 10.0.0.0/24 and http.host contains "packt"`
- Packets from networks `10.0.0.0/24` to websites that end with `.com`:
  - `ip.addr == 10.0.0.0/24 and http.host matches ".com$"`
- All broadcasts from source IP address `10.0.0.3`:
  - `ip.src == 10.0.0.0/24 and eth.dst == ffff.ffff.ffff`
- All broadcasts that are not ARP requests:
  - `not arp and eth.dst == ffff.ffff.ffff`
- All packets that are not ICMP and not ARP:
  - `!arp || !icmp or not arp&&not icmp`

TCP and UDP port number display filters

For TCP or UDP port numbers, use the following display filters:

- `tcp.port == <value>` or `udp.port == <value>` for specific TCP or UDP ports (source or destination)
- `tcp.dstport == <value>` or `udp.dstport == <value>` for specific TCP or UDP destination ports
- `tcp.srcport == <value>` **or** `udp.srcport == <value>` for specific TCP or UDP destination ports

- `tcp.flags`: These filters are used for finding out if flags are set or not:
  - `tcp.flags.syn == 1` to check if the SYN flag is set
  - `tcp.flags.reset == 1` to check if the RST flag is set
  - `tcp.flags.fin == 1` to check if the FIN flag is set
  - `tcp.window_size_value < <value>` to look for small TCP window sizes that are, in some cases, an indication of slow devices

---

Some examples of filters in TCP/UDP filters are as follows:

- All packets to the HTTP server:
  - `tcp.dstport == 80`
- All packets from network `10.0.0.0/24` to HTTP server:
  - `ip.src==10.0.0.0/24 and tcp.dstport == 80`

- All packets from `10.0.0.5` to the DNS server:
  - `ip.src == 10.0.0.5 && udp.port == 53`
- All packets in TCP or protocols in TCP (for example HTTP) that contain the string `cacti` (case-sensitive):
  - `tcp contains "cacti"`
- All packets from `10.0.0.3` that are retransmitted:
  - `ip.src == 10.0.0.3 and tcp.analysis.retransmission`
- All packets to any HTTP server:
  - `tcp.dstport == 80`

Some display Filter Comparison Operators

ip.dst == computer IP

ip.src == computer IP

tcp.port == 80

type http.request in the filter box you will get HTTP GET

type http.response in the filter box you will get HTTP OK

By default the value of the time column in the packet listing window is the amount of time in seconds

To display the time field in Time-of-day format

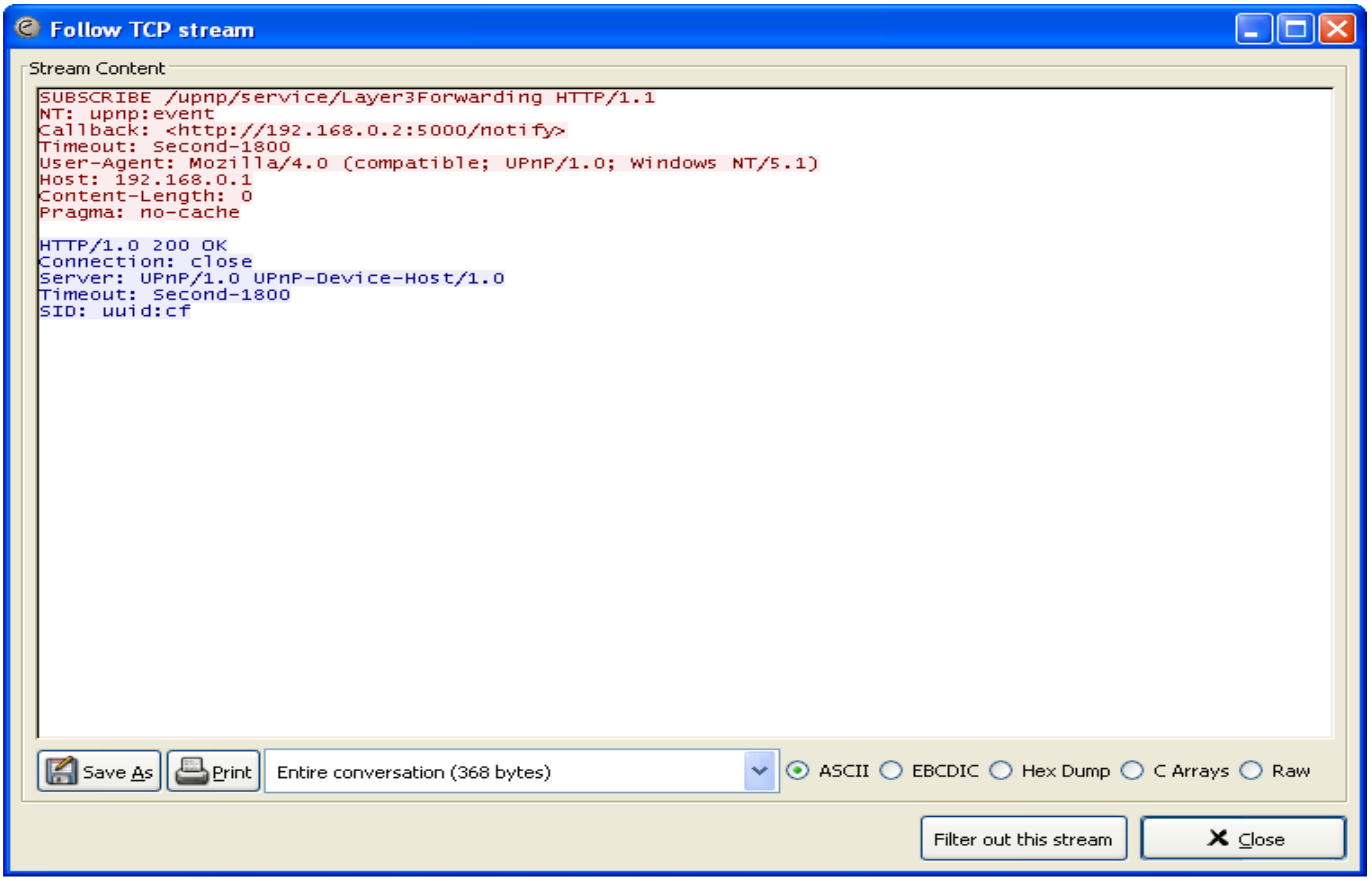
Select View ---→ Time display format ---→ Time –of-day

Refer to the following table for sample capture filters:

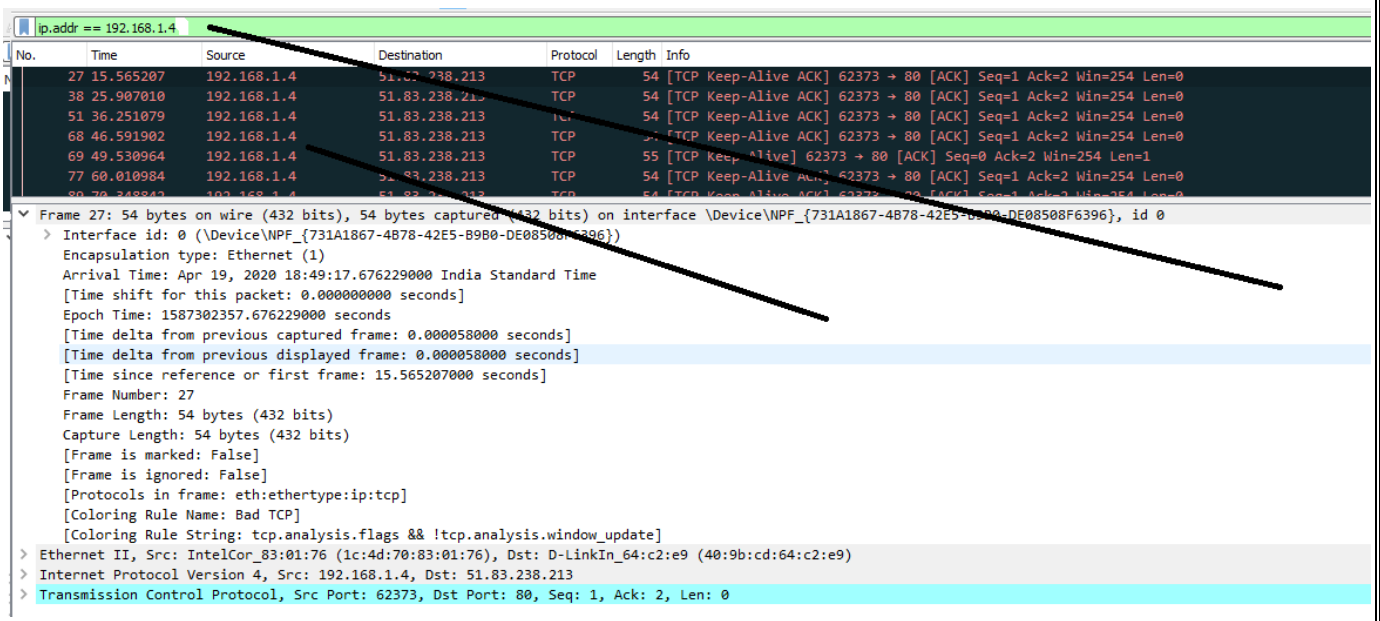
Filters	Description
host192.168.1.1	All traffic associated with host 192.168.1.1
port 8080	All traffic associated with port 8080
src host192.168.1.1	All traffic originating from host 192.168.1.1
dst host 192.168.1.1	All traffic destined to host 192.168.1.1
src port 53	All traffic originating from port 53
dst port 21	All traffic destined to port 21
src192.168.1.1and tcp port 21	All traffic originating from 192.168.1.1and associated with port 21
dst192.168.1.1or dst 192.168.1.2	All traffic destined to 192.168.1.1or destined to host 192.168.1.2
not port 80	All traffic not associated with port 80
not src host 192.168.1.1	All traffic not originating from host 192.168.1.1
not port 21 and not port 22	All traffic not associated with port 21 or port 22
tcp	All tcp traffic
Ipv6	All ipv6 traffic
tcp or udp	All TCP or UDP traffic
host www.google.com	All traffic to and from Google's IP address
ether host 07:34:AA:B6:78:89	All traffic associated with the specified MAC address

Following TCP streams

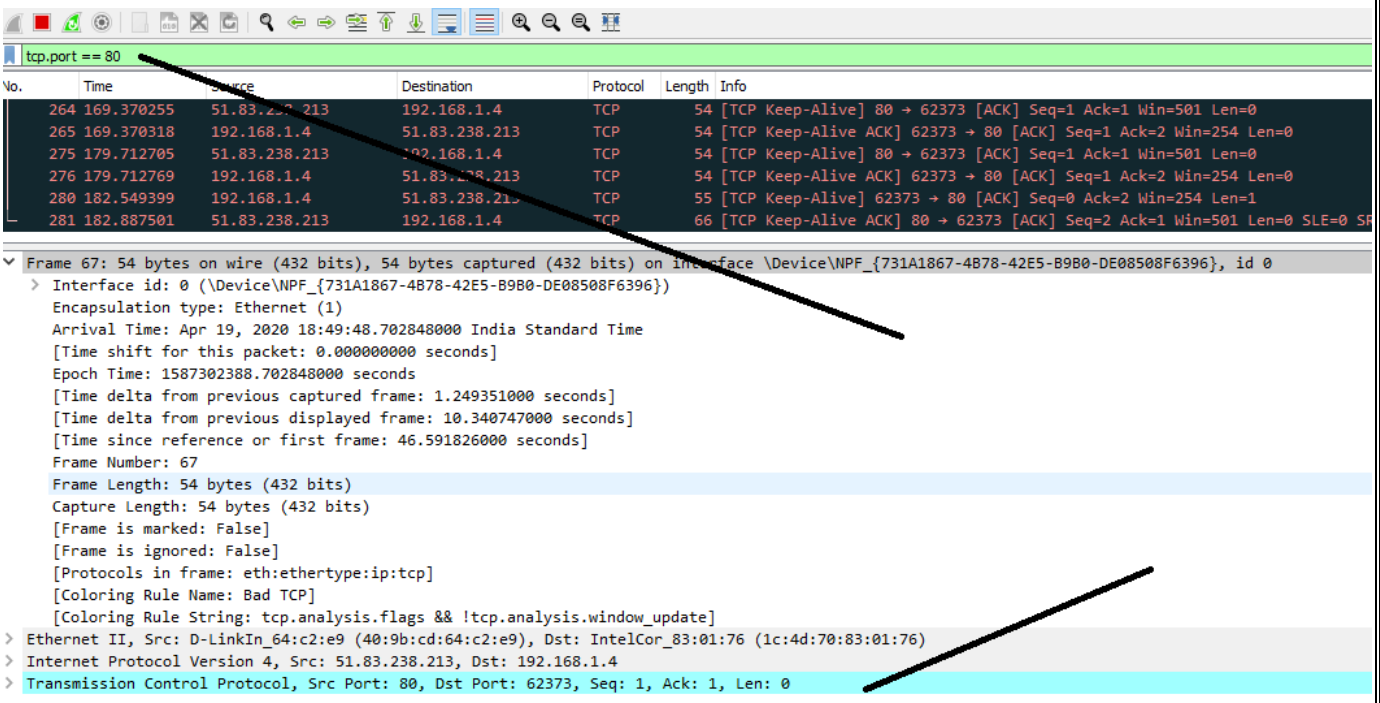
If you are working with TCP based protocols it can be very helpful to see the data from a TCP stream in the way that the application layer sees it. Perhaps you are looking for passwords in a Telnet stream, or you are trying to make sense of a data stream. Maybe you just need a display filter to show only the packets of that TCP stream. If so, Wireshark's ability to follow a TCP stream will be useful to you. It is worthwhile noting that Follow TCP Stream installs a display filter to select all the packets in the TCP stream you have selected. **The "Follow TCP Stream" dialog box**



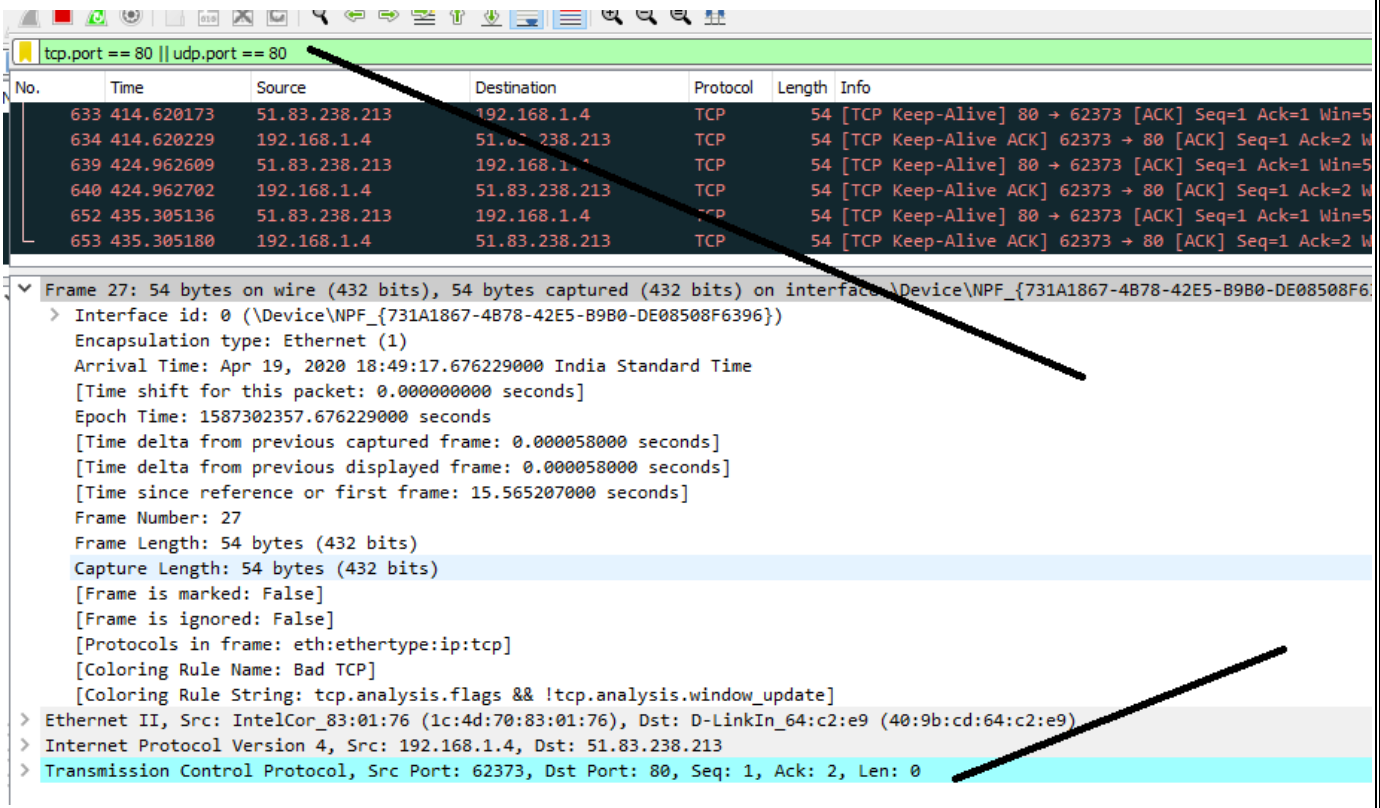
Filter the particular IP address (192.168.1.4)



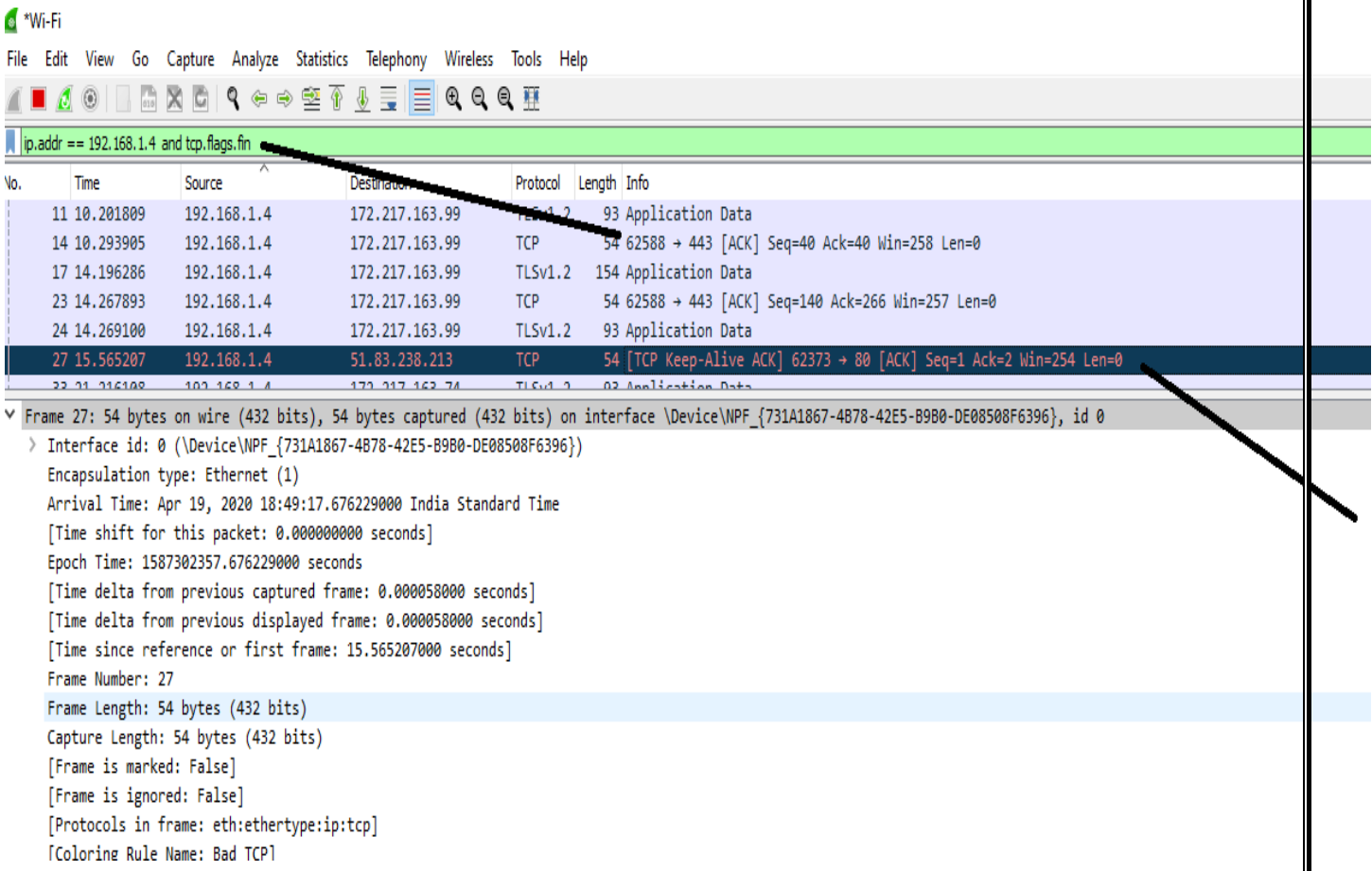
Filter and Display tcp port= 80



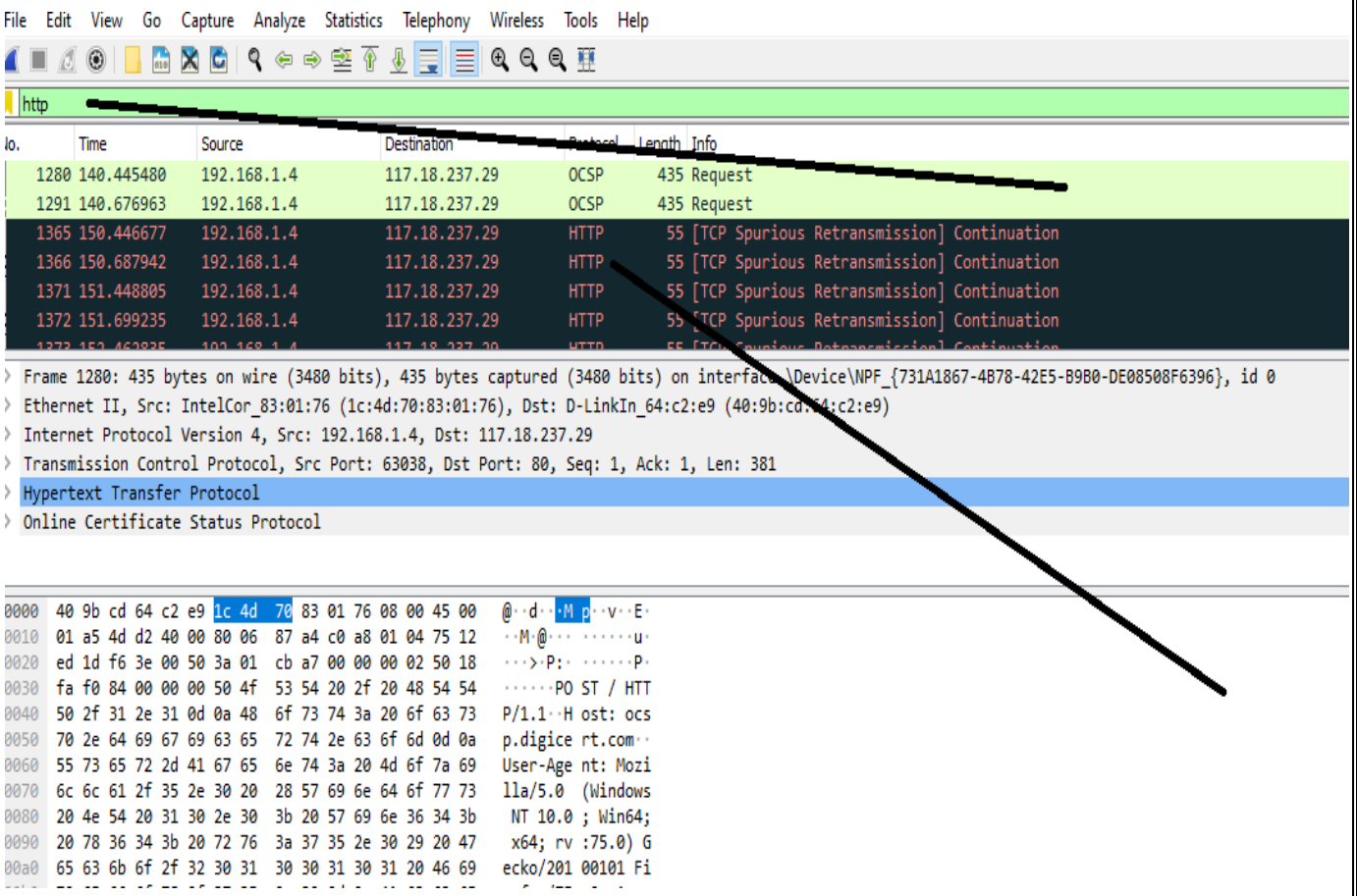
Display tcp port =80 or UDP port =80



Filter the Packet of IP address 192.168.1.4 and tcp.flags.fin



Filter the Http protocol


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
lo. Time Source Destination Protocol Length Info
1280 140.445480 192.168.1.4 117.18.237.29 OCSP 435 Request
1291 140.676963 192.168.1.4 117.18.237.29 OCSP 435 Request
1365 150.446677 192.168.1.4 117.18.237.29 HTTP 55 [TCP Spurious Retransmission] Continuation
1366 150.687942 192.168.1.4 117.18.237.29 HTTP 55 [TCP Spurious Retransmission] Continuation
1371 151.448805 192.168.1.4 117.18.237.29 HTTP 55 [TCP Spurious Retransmission] Continuation
1372 151.699235 192.168.1.4 117.18.237.29 HTTP 55 [TCP Spurious Retransmission] Continuation
Frame 1280: 435 bytes on wire (3480 bits), 435 bytes captured (3480 bits) on interface \Device\NPF_{731A1867-4B78-42E5-B9B0-DE08508F6396}, id 0
Ethernet II, Src: IntelCor_83:01:76 (1c:4d:70:83:01:76), Dst: D-LinkIn_64:c2:e9 (40:9b:cd:64:c2:e9)
Internet Protocol Version 4, Src: 192.168.1.4, Dst: 117.18.237.29
Transmission Control Protocol, Src Port: 63038, Dst Port: 80, Seq: 1, Ack: 1, Len: 381
Hypertext Transfer Protocol
Online Certificate Status Protocol
0000 40 9b cd 64 c2 e9 1c 4d 70 83 01 76 08 00 45 00 @··d···M p··v··E·
0010 01 a5 4d d2 40 00 80 06 87 a4 c0 a8 01 04 75 12 ··M·@··· ·····u·
0020 ed 1d f6 3e 00 50 3a 01 cb a7 00 00 00 02 50 18 ···>·P:· ·····P·
0030 fa f0 84 00 00 00 50 4f 53 54 20 2f 20 48 54 54 ·····PO ST / HTT
0040 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 6f 63 73 P/1.1··H ost: ocs
0050 70 2e 64 69 67 69 63 65 72 74 2e 63 6f 6d 0d 0a p.digice rt.com··
0060 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 User-Age nt: Mozi
0070 6c 6c 61 2f 35 2e 30 20 28 57 69 6e 64 6f 77 73 lla/5.0 (Windows
0080 20 4e 54 20 31 30 2e 30 3b 20 57 69 6e 36 34 3b NT 10.0 ; Win64;
0090 20 78 36 34 3b 20 72 76 3a 37 35 2e 30 29 20 47 x64; rv :75.0) G
00a0 65 63 6b 6f 2f 32 30 31 30 30 31 30 31 20 46 69 ecko/201 00101 Fi

Module-5

NS2 Simulator

Exp 1: THREE NODE POINT TO POINT NETWORK

Aim: Simulate a three node point to point network with duplex links between them. Set queue size and vary the bandwidth and find number of packets dropped.

```
set ns [new Simulator]          # Letter S is capital
set nf [open PA1.nam w]         # open a nam trace file in write mode
$ns namtrace-all $nf    # nf nam filename
set tf [open PA1.tr w] # tf trace filename
$ns trace-all $tf
proc finish { } {
global ns nf tf
$ns flush-trace# clears trace file contents
close $nf
close $tf
exec nam PA1.nam &
exit 0
}
set n0 [$ns node]               # creates 3 nodes
set n2 [$ns node]
set n3 [$ns node]
$ns duplex-link $n0    $n2 200Mb 10ms DropTail    # establishing links
$ns duplex-link $n2    $n3 1Mb 1000ms DropTail
$ns queue-limit $n0    $n2 10
set udp0 [new Agent/UDP]    # attaching transport layer protocols
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]       # attaching application layer protocols
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
set null0 [new Agent/Null]
$ns attach-agent $n3 $null0
$ns connect $udp0 $null0

$ns at 0.1 "$cbr0 start"
$ns at 1.0 "finish"
$ns run
```

AWK file: (Open a new editor using “vi command” and write awk file and save with “.awk” extension)

```
#immediately after BEGIN should open braces {

BEGIN{ c=0;}
{
if($1= ="d")
```

```
{c++;
printf("%s\t%s\n",$5,$11);
}
}
END{ printf("The number of packets dropped =%d\n",c); }
```

Steps for execution

- Open vi editor and type program. Program name should have the extension " .tcl "

[root@localhost ~]# vi lab1.tcl

- Save the program by pressing "ESC key" first, followed by "Shift and :" keys simultaneously and type "wq" and press Enter key.

- Open vi editor and type awk program. Program name should have the extension ".awk "

  [root@localhost ~]# vi lab1.awk

- Save the program by pressing "ESC key" first, followed by "Shift and :" keys simultaneously and type "wq" and press Enter key.

- Run the simulation program

  [root@localhost~]# ns lab1.tcl

- Here "ns" indicates network simulator. We get the topology shown in the snapshot.

- Now press the play button in the simulation window and the simulation will begins.

- After simulation is completed run awk file to see the output ,

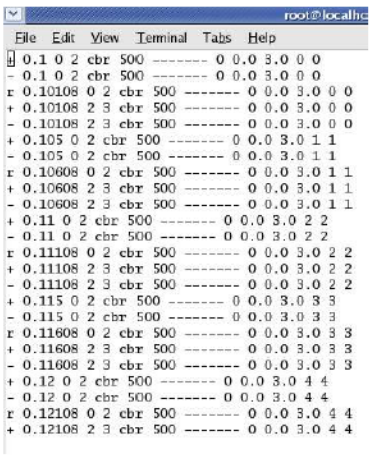
  [root@localhost~]# awk –f lab1.awk lab1.tr

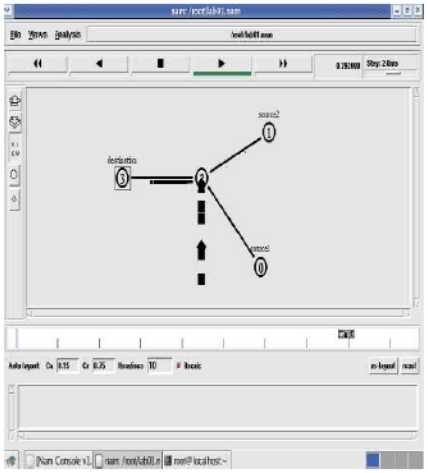
- To see the trace file contents open the file as ,

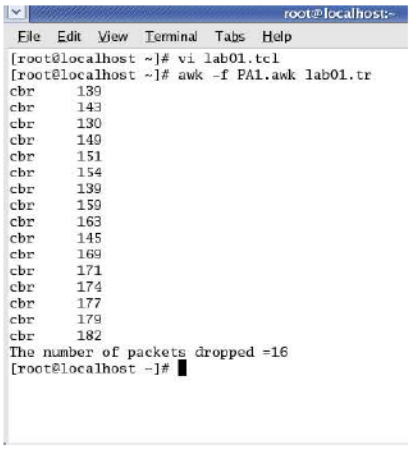
  [root@localhost~]# vi lab1.tr

Trace file contains 12 columns:

Event type, Event time, From Node, To Node, Packet Type, Packet Size, Flags (indicated by --------), Flow ID, Source address, Destination address, Sequence ID, Packet ID







Contents of Trace File Topology Output

Exp 2: TRANSMISSION OF PING MESSAGE

Aim: Simulate the transmission of ping messages over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

```
set ns [ new Simulator ]
set nf [ open lab4.nam w ]
$ns namtrace-all $nf
set tf [ open lab4.tr w ]
$ns trace-all $tf
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
$ns duplex-link $n0 $n4 1005Mb 1ms DropTail
$ns duplex-link $n1 $n4 50Mb 1ms DropTail
$ns duplex-link $n2 $n4 2000Mb 1ms DropTail
$ns duplex-link $n3 $n4 200Mb 1ms DropTail
$ns duplex-link $n4 $n5 1Mb 1ms DropTail
set p1 [new Agent/Ping] # letters A and P should be capital
$ns attach-agent $n0 $p1
$p1 set packetSize_ 50000
$p1 set interval_ 0.0001
set p2 [new Agent/Ping]         # letters A and P should be capital
$ns attach-agent $n1 $p2
set p3 [new Agent/Ping]         # letters A and P should be capital
$ns attach-agent $n2 $p3
$p3 set packetSize_ 30000
$p3 set interval_ 0.00001
set p4 [new Agent/Ping]         # letters A and P should be capital
$ns attach-agent $n3 $p4
set p5 [new Agent/Ping]         # letters A and P should be capital
$ns attach-agent $n5 $p5
$ns queue-limit $n0 $n4 5
$ns queue-limit $n2 $n4 3
$ns queue-limit $n4 $n5 2
Agent/Ping instproc recv {from rtt} {
$self instvar node_
puts "node [$node_ id]received answer from $from with round trip time $rtt msec" }

#         please provide space between $node_ and id. No space between $ and from. No space between and $ and rtt */

$ns connect $p1 $p5
$ns connect $p3 $p4
proc finish { } {
global ns nf tf
$ns flush-trace
```

```
close $nf
close $tf
exec nam lab4.nam &
exit 0
}
$ns at 0.1 "$p1 send"
$ns at 0.2 "$p1 send"
$ns at 0.3 "$p1 send"
$ns at 0.4 "$p1 send"
$ns at 0.5 "$p1 send"
$ns at 0.6 "$p1 send"
$ns at 0.7 "$p1 send"
$ns at 0.8 "$p1 send"
$ns at 0.9 "$p1 send"
$ns at 1.0 "$p1 send"
$ns at 1.1 "$p1 send"
$ns at 1.2 "$p1 send"
$ns at 1.3 "$p1 send"
$ns at 1.4 "$p1 send"
$ns at 1.5 "$p1 send"
$ns at 1.6 "$p1 send"
$ns at 1.7 "$p1 send"
$ns at 1.8 "$p1 send"
$ns at 1.9 "$p1 send"
$ns at 2.0 "$p1 send"
$ns at 2.1 "$p1 send"
$ns at 2.2 "$p1 send"
$ns at 2.3 "$p1 send"
$ns at 2.4 "$p1 send"
$ns at 2.5 "$p1 send"
$ns at 2.6 "$p1 send"
$ns at 2.7 "$p1 send"
$ns at 2.8 "$p1 send"
$ns at 2.9 "$p1 send"

$ns at 0.1 "$p3 send"
$ns at 0.2 "$p3 send"
$ns at 0.3 "$p3 send"
$ns at 0.4 "$p3 send"
$ns at 0.5 "$p3 send"
$ns at 0.6 "$p3 send"
$ns at 0.7 "$p3 send"
$ns at 0.8 "$p3 send"
$ns at 0.9 "$p3 send"
$ns at 1.0 "$p3 send"
$ns at 1.1 "$p3 send"
$ns at 1.2 "$p3 send"
$ns at 1.3 "$p3 send"
$ns at 1.4 "$p3 send"
$ns at 1.5 "$p3 send"
```

```
$ns at 1.6 "$p3 send"
$ns at 1.7 "$p3 send"
$ns at 1.8 "$p3 send"
$ns at 1.9 "$p3 send"
$ns at 2.0 "$p3 send"
$ns at 2.1 "$p3 send"
$ns at 2.2 "$p3 send"
$ns at 2.3 "$p3 send"
$ns at 2.4 "$p3 send"
$ns at 2.5 "$p3 send"
$ns at 2.6 "$p3 send"
$ns at 2.7 "$p3 send"
$ns at 2.8 "$p3 send"
$ns at 2.9 "$p3 send"

$ns at 3.0 "finish"
$ns run
```

AWK file: (Open a new editor using “vi command” and write awk file and save with “.awk” extension)

```
BEGIN{
drop=0;
}
{
if($1= ="d" )
{
drop++;
}
}
END{
printf("Total number of %s packets dropped due to congestion =%d\n",$5,drop);
}
```

Steps for execution:

1) Open vi editor and type program. Program name should have the extension “ .tcl ”

   [root@localhost ~]# vi lab4.tcl

2) Save the program by pressing “ESC key” first, followed by “Shift and :” keys simultaneously and type “wq” and press Enter key.

3) Open vi editor and type awk program. Program name should have the extension “.awk ”

   [root@localhost ~]# vi lab4.awk

4) Save the program by pressing “ESC key” first, followed by “Shift and :” keys simultaneously and type “wq” and press Enter key.

5) Run the simulation program

[root@localhost~]# ns lab4.tcl

i) Here “ns” indicates network simulator. We get the topology shown in the snapshot.

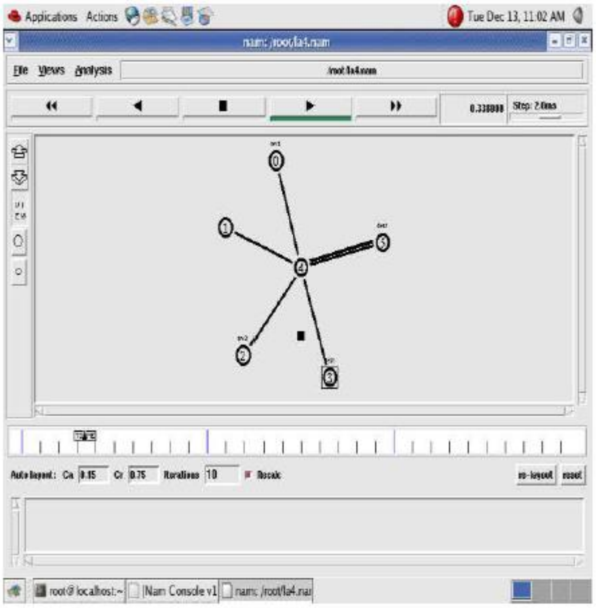
ii) Now press the play button in the simulation window and the simulation will begins.

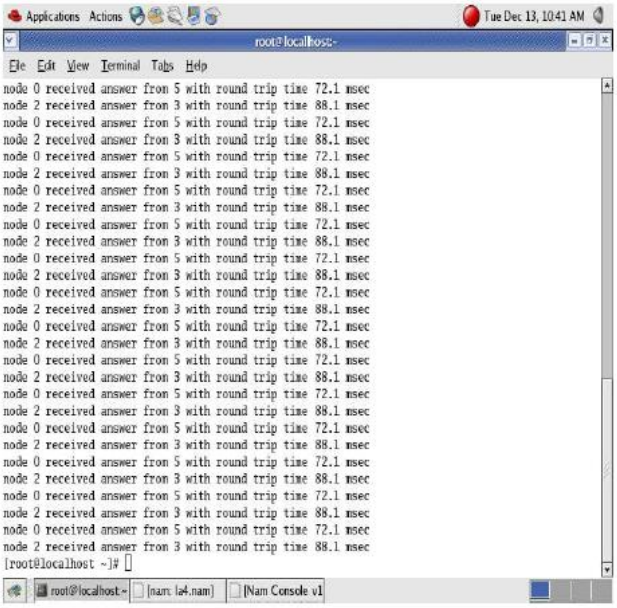
6) After simulation is completed run awk file to see the output,

[root@localhost~]# awk –f lab4.awk lab4.tr

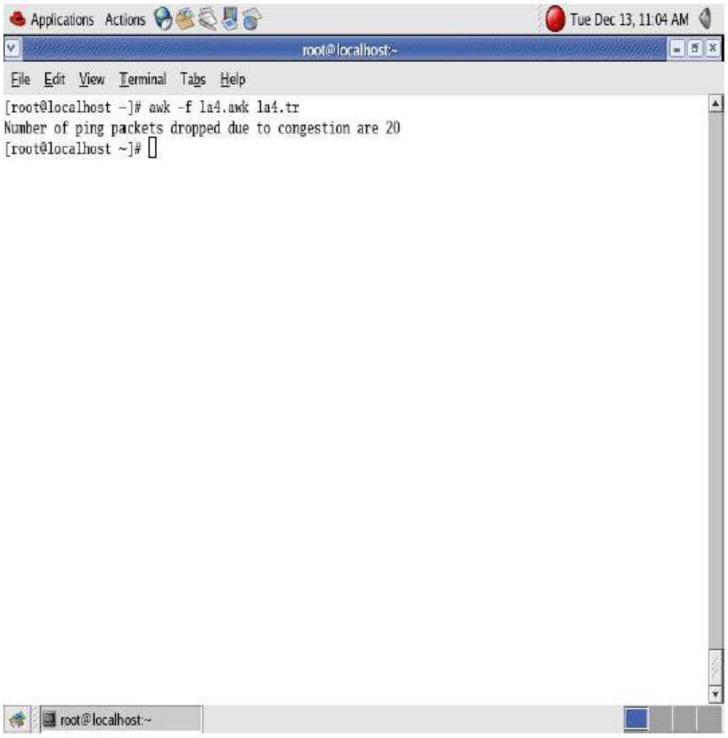
7) To see the trace file contents open the file as ,

[root@localhost~]# vi lab4.tr





Topology **Output**



Output

Exp 3: ETHERNET LAN USING N-NODES

Aim: Simulate an Ethernet LAN using n nodes, change error rate and data rate and compare throughput.

```
set ns [new Simulator]
set tf [open lab5.tr w]
$ns trace-all $tf
set nf [open lab5.nam w]
$ns namtrace-all $nf
$ns color 0 blue
set n0 [$ns node]
$n0 color "red"
set n1 [$ns node]
$n1 color "red"
set n2 [$ns node]
$n2 color "red"
set n3 [$ns node]
$n3 color "red"
set n4 [$ns node]
$n4 color "magenta"
set n5 [$ns node]
$n5 color "magenta"
set n6 [$ns node]
$n6 color "magenta"
set n7 [$ns node]
$n7 color "magenta"
$ns make-lan "$n0 $n1 $n2 $n3" 100Mb 300ms LL Queue/ DropTail Mac/802_3 $ns make-lan "$n4 $n5 $n6 $n7" 100Mb 300ms LL Queue/ DropTail Mac/802_3
$ns duplex-link $n3 $n4 100Mb 300ms DropTail $ns duplex-link-op $n3 $n4 color "green"

#       set error rate. Here ErrorModel is a class and it is single word and space should not be given between Error and Model

#       lossmodel is a command and it is single word. Space should not be given between loss and model

set err [new ErrorModel]
$ns lossmodel $err $n3 $n4
$err set rate_ 0.1

# error rate should be changed for each output like 0.1,0.3,0.5…. */

set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set fid_ 0
$cbr set packetSize_ 1000
$cbr set interval_ 0.0001
```

```
set null [new Agent/Null]
$ns attach-agent $n7 $null
$ns connect $udp $null
proc finish { } {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam lab5.nam &
exit 0
}
$ns at 0.1 "$cbr start"
$ns at 3.0 "finish"
$ns run
```

AWK file: (Open a new editor using “vi command” and write awk file and save with “.awk” extension)

```
BEGIN{
pkt=0;
time=0;
}
{
if($1= ="r" && $3= ="9" && $4= ="7"){
pkt = pkt + $6;
time =$2;
}
}
END {
printf("throughput:%fMbps",(( pkt / time) * (8 / 1000000)));
}
```

Steps for execution

- Open vi editor and type program. Program name should have the extension “ .tcl ”

[root@localhost ~]# vi lab5.tcl

- Save the program by pressing “ESC key” first, followed by “Shift and :” keys simultaneously and type “wq” and press Enter key.
- Open vi editor and type awk program. Program name should have the extension “.awk ”

[root@localhost ~]# vi lab5.awk

- Save the program by pressing “ESC key” first, followed by “Shift and :” keys simultaneously and type “wq” and press Enter key.
- Run the simulation program

[root@localhost~]# ns lab5.tcl

o Here “ns” indicates network simulator. We get the topology shown in the snapshot.

o Now press the play button in the simulation window and the simulation will begins.

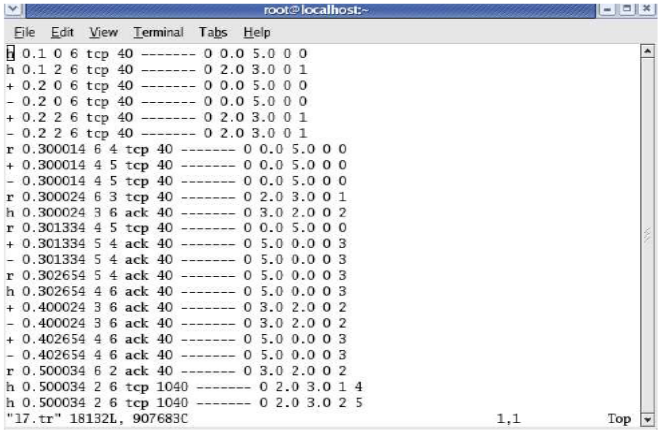
 After simulation is completed run awk file to see the output ,

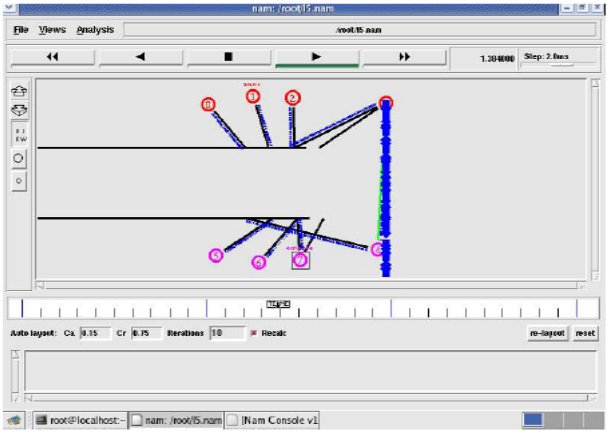
[root@localhost~]# awk –f lab5.awk lab5.tr

 To see the trace file contents open the file as ,

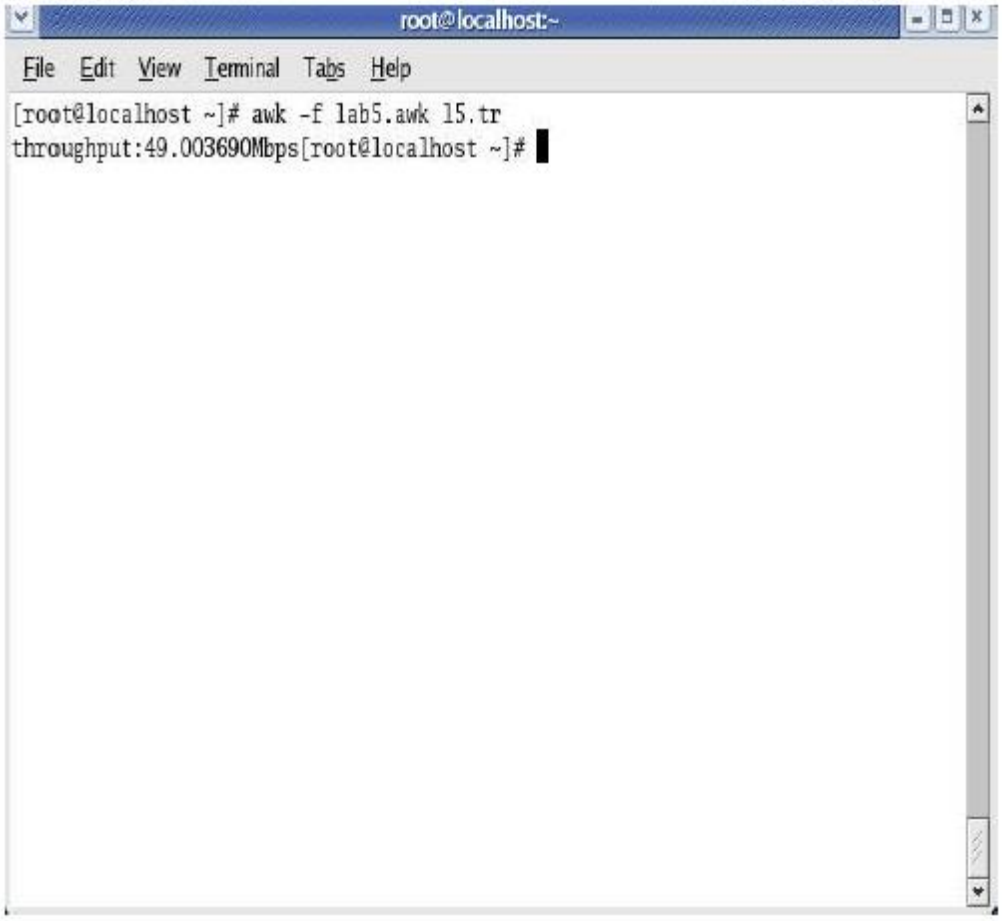
[root@localhost~]# vi lab5.tr

Here “h” indicates host.





Topology **Output**



This above output is for error rate 0.1. During next execution of simulation change error rate to 0.3, 0.5….., and check its effect on throughput.

Exp 4: ETHERNET LAN USING N-NODES WITH MULTIPLE TRAFFIC

Aim: Simulate an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination

```
set ns [new Simulator]
set tf [open pgm7.tr w]
$ns trace-all $tf
set nf [open pgm7.nam w]
$ns namtrace-all $nf
set n0 [$ns node]
$n0 color "magenta"
$n0 label "src1"
set n1 [$ns node]
set n2 [$ns node]
$n2 color "magenta"
$n2 label "src2"
set n3 [$ns node]
$n3 color "blue"
$n3 label "dest2"
set n4 [$ns node]
set n5 [$ns node]
$n5 color "blue"
$n5 label "dest1"
$ns make-lan "$n0 $n1 $n2 $n3 $n4" 100Mb 100ms LL Queue/ DropTail Mac/802_3
```

```
#should come in single line

$ns duplex-link $n4 $n5 1Mb 1ms DropTail
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ftp0 set packetSize_ 500
$ftp0 set interval_ 0.0001
set sink5 [new Agent/TCPSink]
$ns attach-agent $n5 $sink5
$ns connect $tcp0 $sink5
set tcp2 [new Agent/TCP]
$ns attach-agent $n2 $tcp2
set ftp2 [new Application/FTP]
$ftp2 attach-agent $tcp2
$ftp2 set packetSize_ 600
$ftp2 set interval_ 0.001
set sink3 [new Agent/TCPSink]
$ns attach-agent $n3 $sink3
$ns connect $tcp2 $sink3
set file1 [open file1.tr w]
$tcp0 attach $file1
set file2 [open file2.tr w]
$tcp2 attach $file2
$tcp0 trace cwnd_      # must put underscore ( _ ) after cwnd and no space between them $tcp2 trace
cwnd_
proc finish { } {
global ns nf tf
$ns flush-trace
close $tf
close $nf
exec nam pgm7.nam &
exit 0
}
$ns at 0.1 "$ftp0 start"
$ns at 5 "$ftp0 stop"
$ns at 7 "$ftp0 start"
$ns at 0.2 "$ftp2 start"
$ns at 8 "$ftp2 stop"
$ns at 14 "$ftp0 stop"
$ns at 10 "$ftp2 start"
$ns at 15 "$ftp2 stop"
$ns at 16 "finish"
$ns run
```

AWK file: (Open a new editor using “vi command” and write awk file and save with “.awk” extension)

cwnd:- means congestion window

```
BEGIN {
}
{
if($6= ="cwnd_")                # don't leave space after writing cwnd_
printf("%f\t%f\t\n",$1,$7);      # you must put \n in printf
}
END {
}
```

Steps for execution

- Open vi editor and type program. Program name should have the extension ".tcl "

[root@localhost ~]# vi lab7.tcl

- Save the program by pressing "ESC key" first, followed by "Shift and :" keys simultaneously and type "wq" and press Enter key.
- Open vi editor and type awk program. Program name should have the extension ".awk "

[root@localhost ~]# vi lab7.awk

- Save the program by pressing "ESC key" first, followed by "Shift and :" keys simultaneously and type "wq" and press Enter key.
- Run the simulation program

[root@localhost~]# ns lab7.tcl

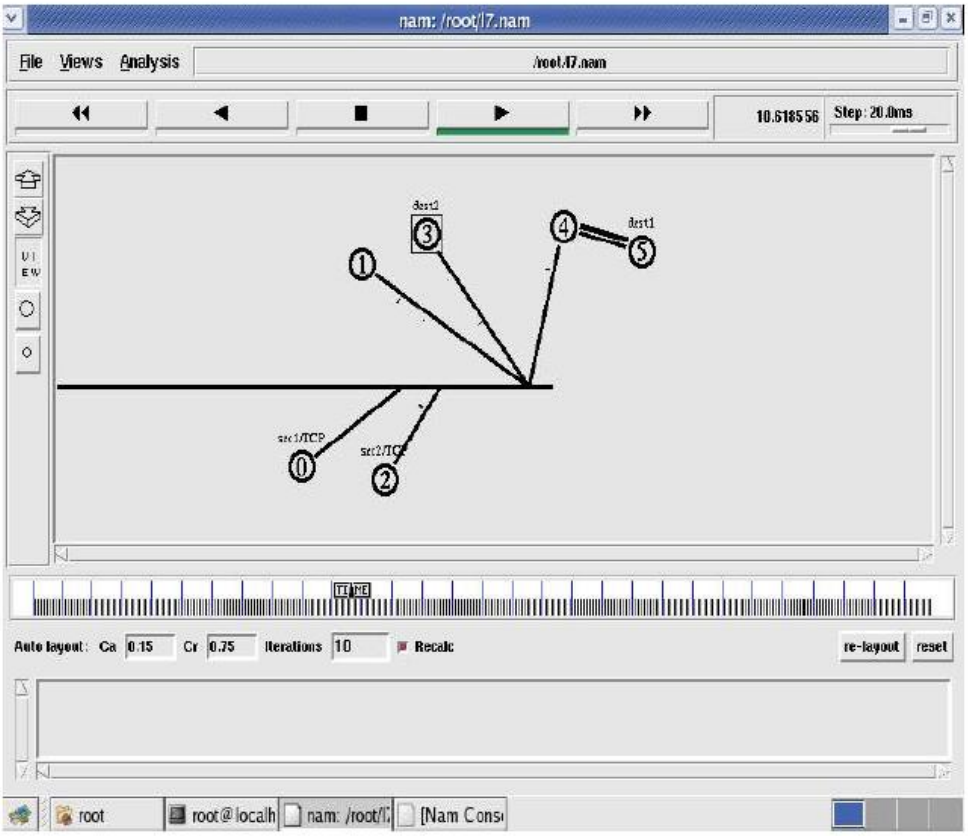
- After simulation is completed run awk file to see the output , [root@localhost~]# awk –f lab7.awk file1.tr > a1 [root@localhost~]# awk –f lab7.awk file2.tr > a2 [root@localhost~]# xgraph a1 a2

- Here we are using the congestion window trace files i.e. file1.tr and file2.tr and we are redirecting the contents of those files to new files say a1 and a2 using output redirection operator (>).

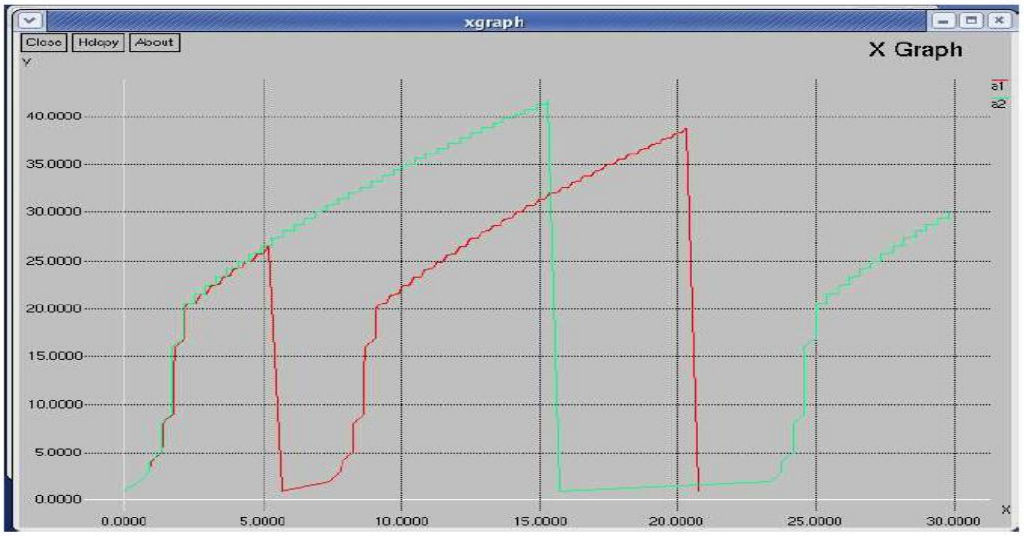
- To see the trace file contents open the file as ,

[root@localhost~]# vi lab7.tr

Topology:


nam: /root/l7.nam
File Views Analysis
/root/l7.nam
10.618556
Step: 20.0ms
dest2
dest1
src1/TCP
src2/TCP
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc
re-layout
reset
root
root@localh
nam: /root/l7
[Nam Cons

Output:


xgraph
Close
Hdcpy
About
X Graph
Y
40.0000
35.0000
30.0000
25.0000
20.0000
15.0000
10.0000
5.0000
0.0000
0.0000
5.0000
10.0000
15.0000
20.0000
25.0000
30.0000
X

Module-4

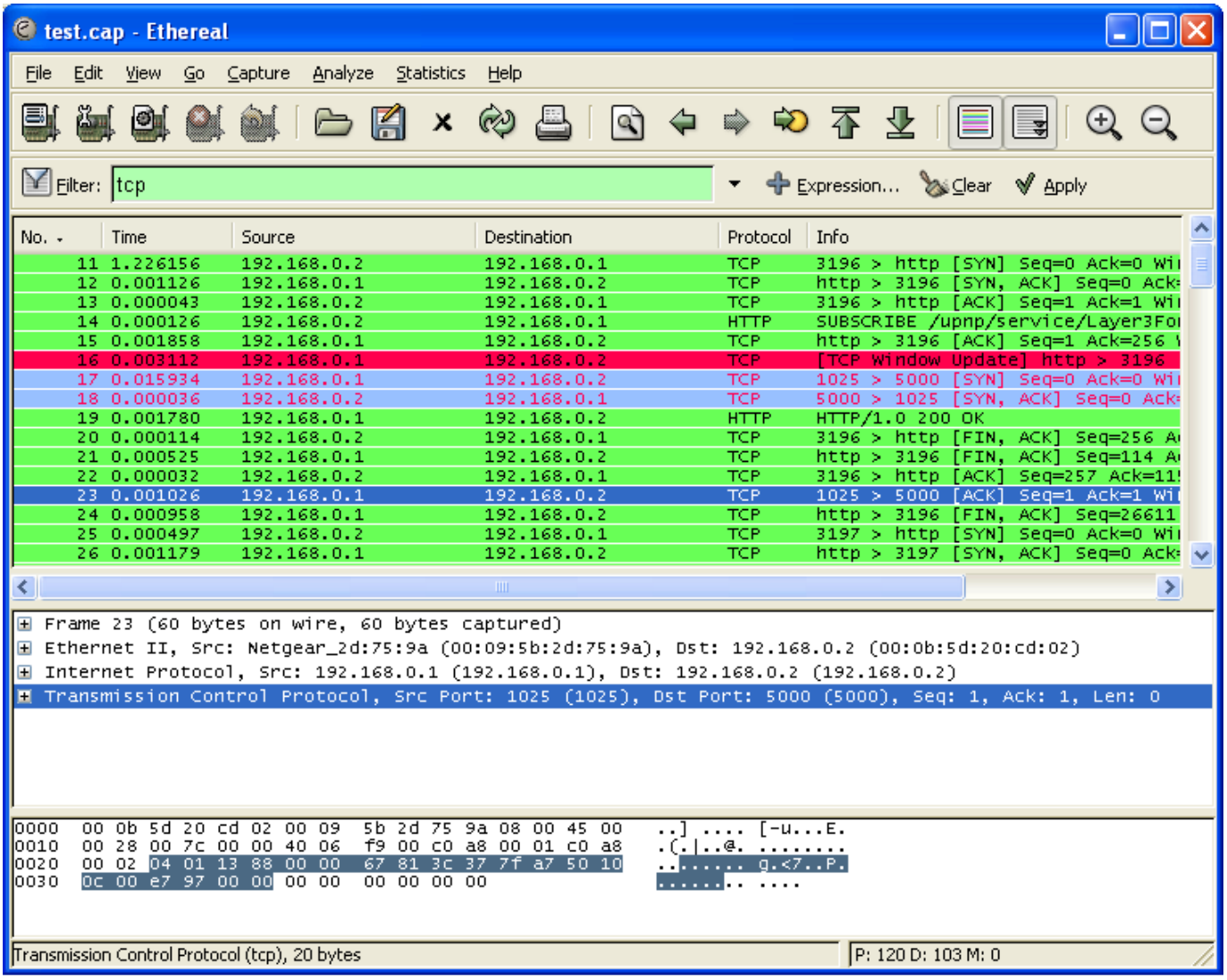
Wireshark tool

Introduction:

Wireshark, formerly known as Ethereal, is one of the most powerful tools in a network security analyst's toolkit. As a network packet analyzer, Wireshark can peer inside the network and examine the details of traffic at a variety of levels, ranging from connection-level information to the bits comprising a single packet. This flexibility and depth of inspection allows the valuable tool to analyze security events and troubleshoot network security device issues.

Installation:

You can find the source code at www.wireshark.org, download and install on your computer



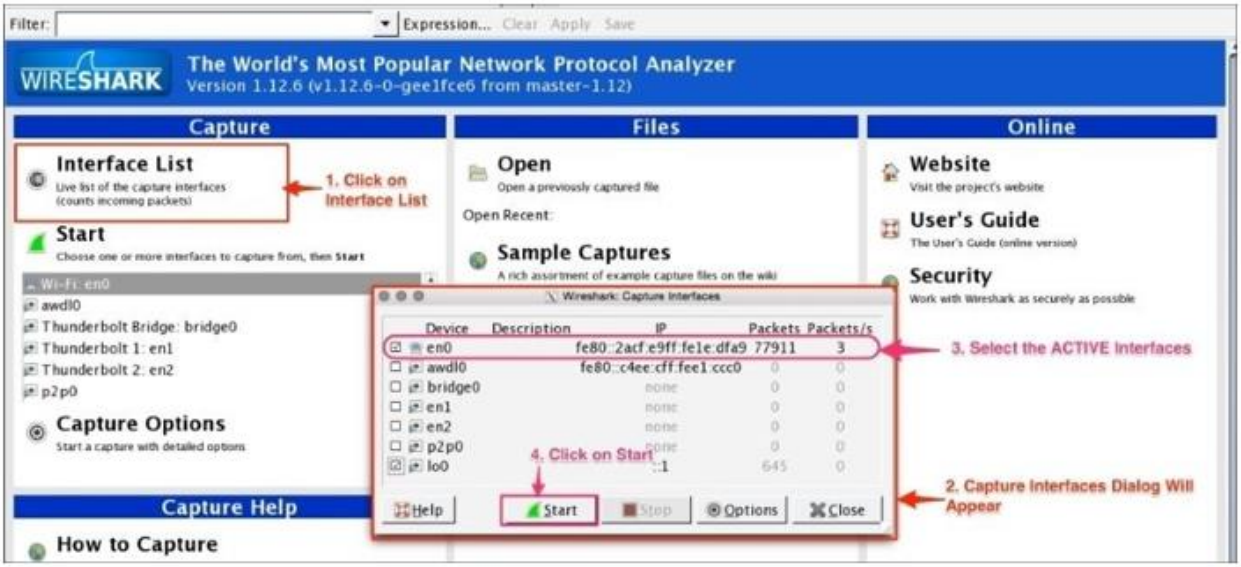
Filter: field in the filter toolbar of the Wireshark window and press enter to initiate the filter.

Wireshark has the following cool built-in features, few of them are listed as follows:

- Available in both UNIX and Windows
- Ability to capture live packets from various types of interface
- Filters packets with many criteria
- Ability to decode larger sets of protocols
- Can save and merge captured packets
- Can create various statistics

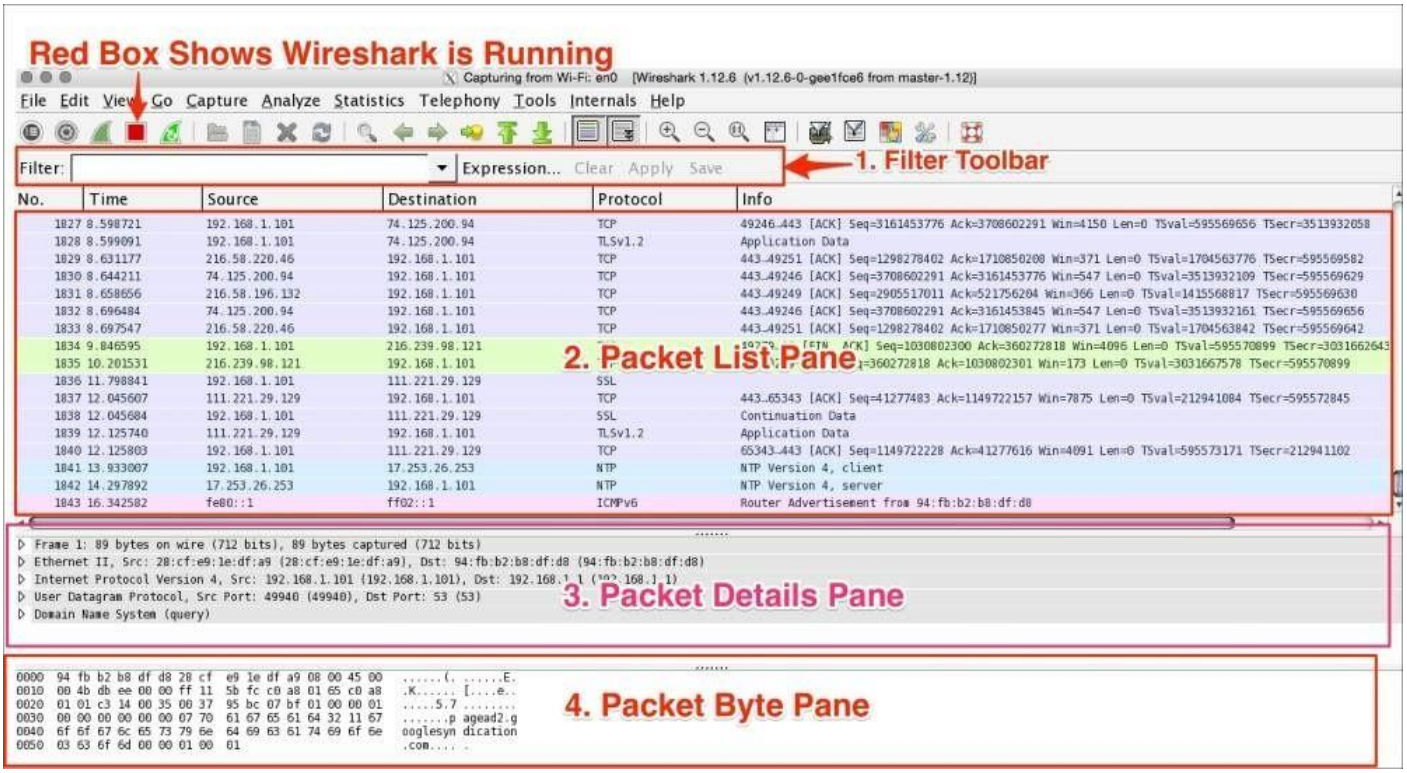
Capturing packets with Interface Lists

Click on **Interface List**; Wireshark will show a list of available network interfaces in the system and which one is active, by showing packets going in and out of the Interface, as shown in the following screenshot:



Wireshark user interface

The Wireshark main window appears when Wireshark starts capturing a packet, or when a .pcapfile is open for offline viewing. It looks similar to the following screenshot:



The Wireshark UI interface consists of different panes and provides various options to the user for customizing it. In this chapter, we will cover these panes indetail:

Item	What is it?
The red box	This shows that Wireshark is running and capturing a packet
1	This is the **Filter** toolbar, used for filtering packets based on the applied filter
2	This is the Packet List pane, which displays all captured packets
3	This is the Packet Details pane, which shows the selected packet in a verbose form
4	This is the Packet Byte pane, which shows the selected packet in a hex dump format

First, just observe pane **2** in the screen; the displayed packets appear with different colors. This is one of Wireshark's best features; it colors packets according to the set filter and helps you visualize the packet you are looking for.

To manage (view, edit, or create) a coloring rule, go to **View** | **Coloring Rules**. Wireshark will display the **Coloring Rules** dialog box, as shown in the screenshot:

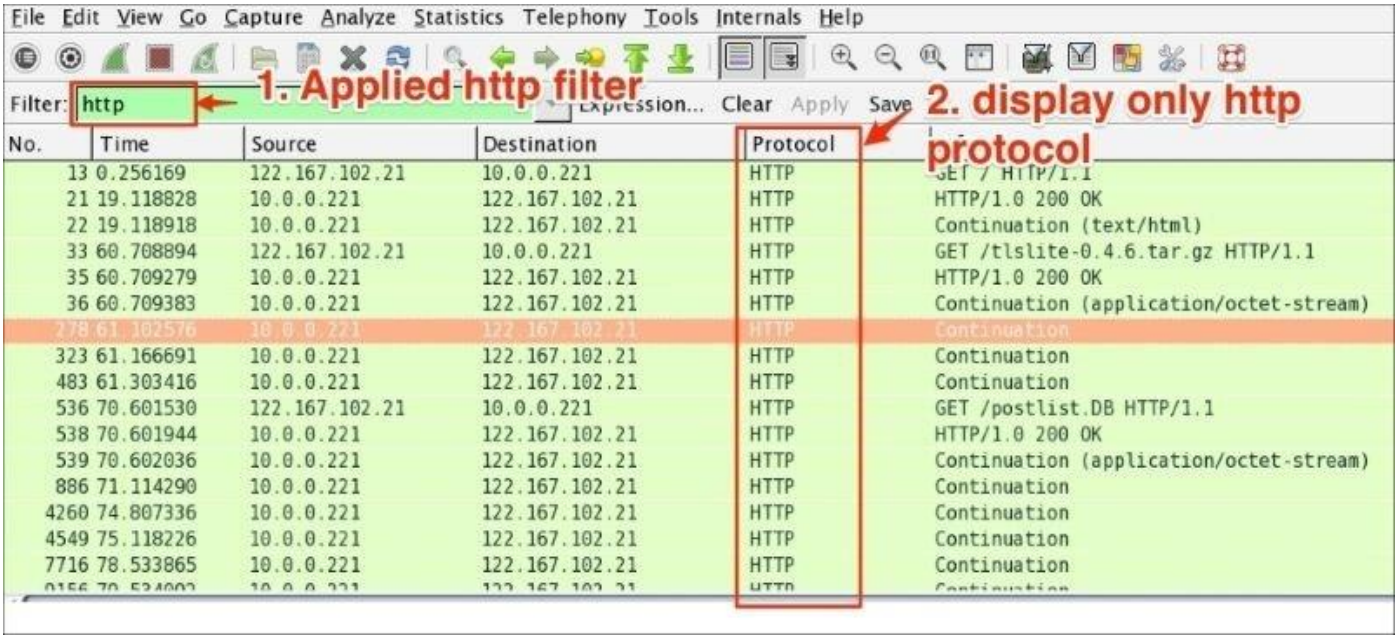
The Filter toolbar

The Wireshark display filter displays packets with its available coloring options. Wireshark display filters are used to change the view of a capture file by providing the full dissection of all packets, which helps analyzing a network tracefile efficiently. For example, if a user is interested in only HTTP packets, the user can set the display filter to http, as shown in the next screenshot.

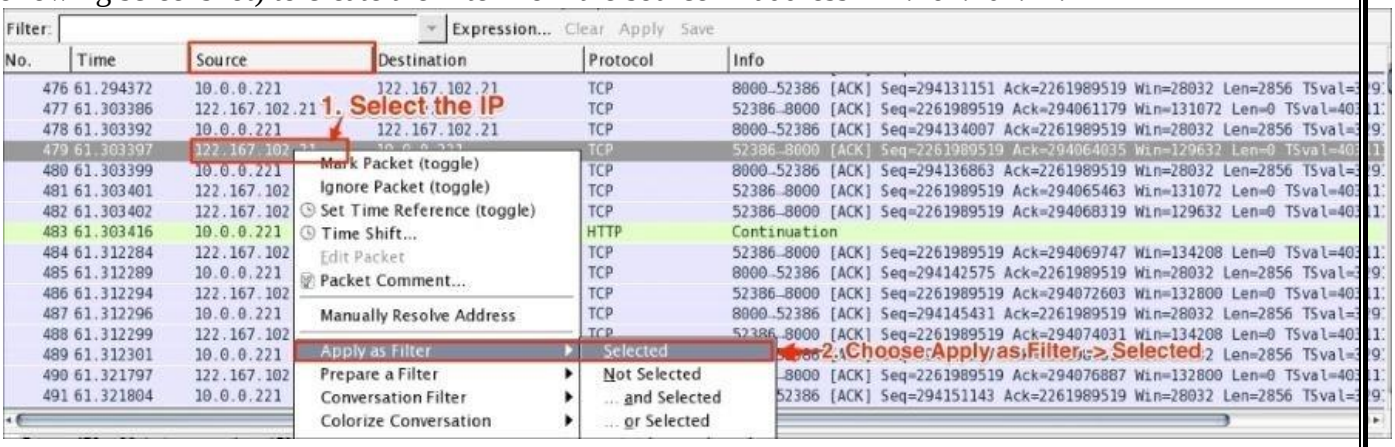
The steps to apply display filters are as follows:

3. Open the http_01.pcapfile.
4. Type the httpprotocol in the filter area and click on **Apply**.

Once the filter is applied, the Packet List pane will display only HTTP protocol-related packets:



Wireshark display filter can be applied or prepared from the column displayed in the Packet List pane by selecting the column, then right-clicking and going to **Apply as Filter** | **Selected** (as shown in the following screenshot) to create the filter from the source IP address 122.167.102.21:



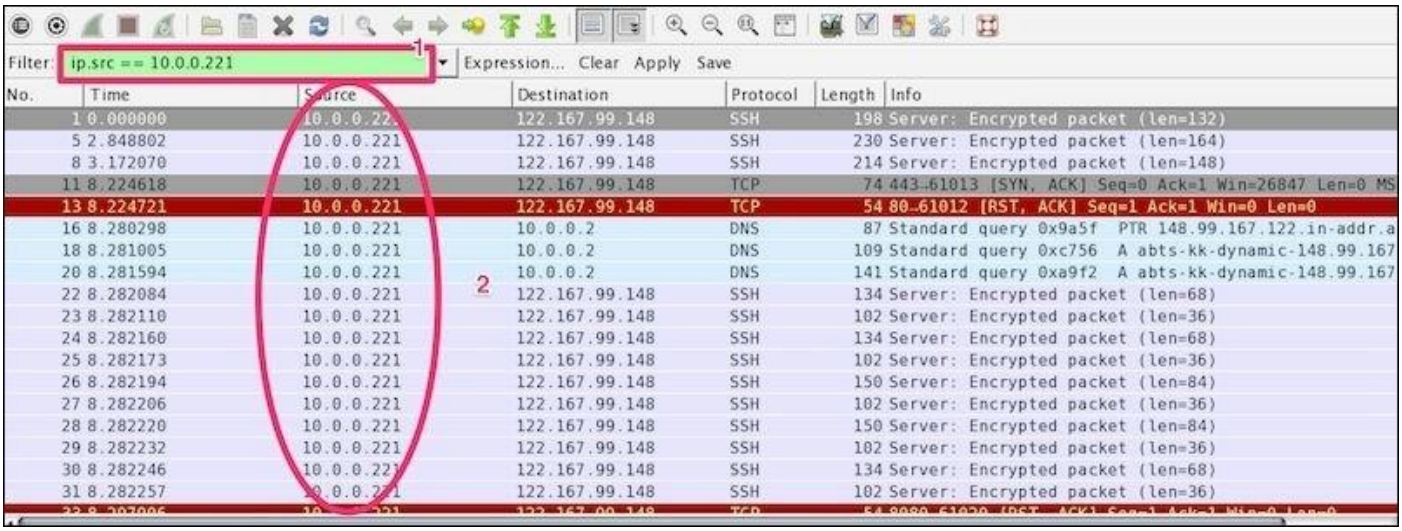
Wireshark provides the flexibility to apply filters from the Details pane; the steps remain the same.

Wireshark also provides the option to clear the filter. To do this click on **Clear** (available in the **Filter** toolbar) to display the entire captured packet.

Filtering techniques

Capturing and displaying packets properly will help you with packet captures. For example, to track a packet exchanged between two hosts: HOSTA(10.0.0.221) and HOSTB (122.167.99.148), open the SampleCapture01.pcapfile and apply the filter ip.src ==

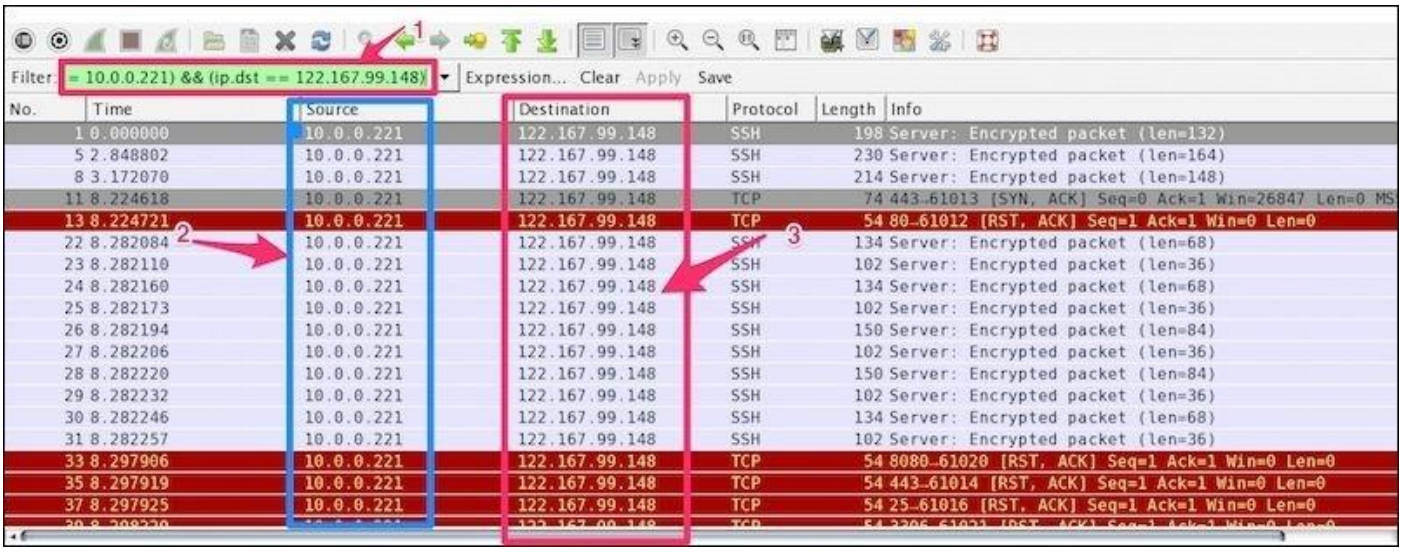
10.0.0.221as shown:



Let's see what the highlighted sections depict:

Item	Description
1	Apply filter ip.src == 10.0.0.221.
2	The Packet List pane displays the traffic from source to destination. The source shows the constant IP address 10.0.0.221. There is no evidence as to which packet is sent from host 122.167.99.148to host 10.0.0.221.

Now modify the filter (ip.src == 10.0.0.221) && (ip.dst == 122.167.99.148)to (ip.src == 10.0.0.221) or (ip.dst ==122.167.99.148). This will give the result shown in the following screenshot:

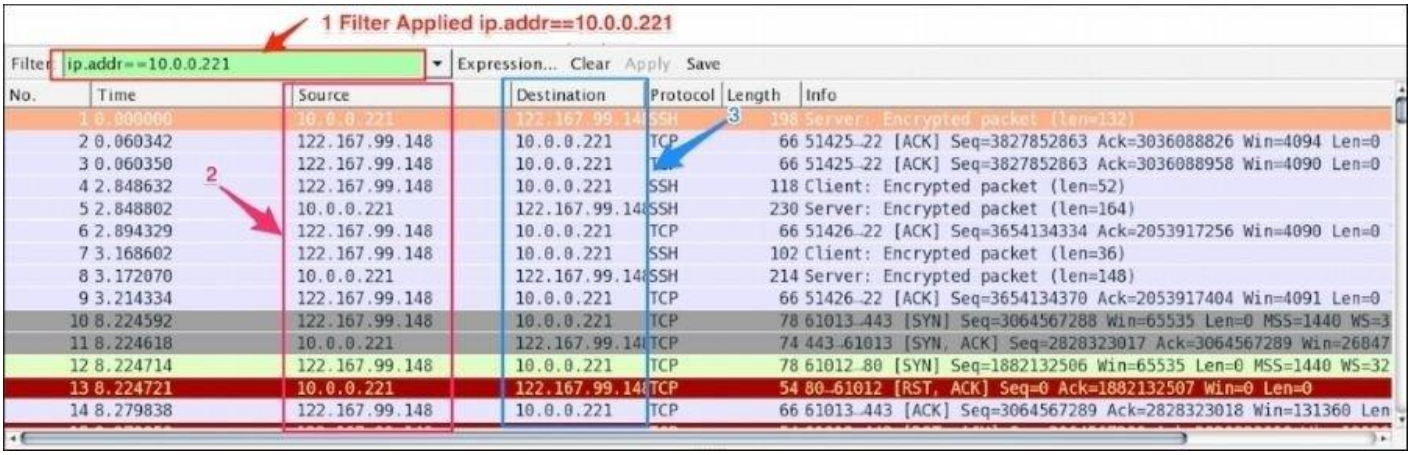


The highlighted sections in the preceding screenshot are explained as follows:

Item	Description
1	Applied filter (ip.src == 10.0.0.221) && (ip.dst == 122.167.99.148)
2	The source IP address (10.0.0.221) is not changed
3	The destination IP address (122.167.99.148) is not changed

Again the Packet List pane is not displaying the conversation between the two hosts.

Now modify the filter ip.addr == 122.167.99.148. The ip.addrfield will match the IP header for both the source and destination address and display the conversation between the hosts. Remember to choose the destination IP address as shown:



Let’s see what the highlighted sections depict:

Item	Description
1	Applied filter ip.addr == 122.167.99.148
2	The source IP is not constant; it shows the conversation between the two hosts
3	The destination IP is not constant; it shows the conversation between the two hosts

The same conversation is captured by choosing the destination MAC address using the display filter eth.addr == 06:73:7a:4c:2f:85.

Filter examples

Some common filter examples are as follows:

Filter/capture name	Filter value

Packet on a given port	tcp.port == 443
Packet on the source port	tcp.srcport=2222
SYN packet on port 443	(tcp.port == 443) && (tcp.flags == 0x0010)
The HTTP protocol	http
Based on the HTTP getmethod	http.request.method == "GET"
Using &&, tcp, and http	tcp && http
Checking the tcpwindow size	tcp.window_size <2000
No Arpused for normal traffic	!arp
The MAC address filter	eth.dst == 06:43:7b:4c:4f:85
Filter out TCP ACK	tcp.flags.ack==0
Check only RST and ACK packets	(tcp.flags.ack == 1) && (tcp.flags.reset == 1)
Filter all SNMP	Snmp
HTTP or DNS or SSL	http || dns | ssl

There is no need to memorize the filter; there is an easy way to apply it. The display filter Autocomplete feature lists all dissectors after the first period "." that have been added to the display filter, as shown in the following screenshot:

Display Filter comparison operators

eq	==	**Equal** ip.addr==10.0.0.5
ne	!=	**Not equal** ip.addr!=10.0.0.5
gt	>	**Greater than** frame.pkt_len > 10
lt	<	**Less than** frame.pkt_len < 128
ge	>=	**Greater than or equal to** frame.pkt_len ge 0x100
le	<=	**Less than or equal to** frame.pkt_len <= 0x20

Display Filter Logical Operations

English	C-like	Description and example
and	&&	**Logical AND** ip.addr==10.0.0.5 and tcp.flags.fin
or	||	**Logical OR** ip.addr==10.0.0.5 or ip.addr==192.1.1.1
xor	^^	**Logical XOR** tr.dst[0:3] == 0.6.29 xor tr.src[0:3] == 0.6.29
not	!	**Logical NOT** not llc

C-like syntax	Shortcut	Description	Example
==	eq	Equal	ip.addr == 192.168.1.1 or ip.addr eq 192.168.1.1
!=	ne	Not equal	!ip.addr==192.168.1.1 or ip.addr != 192.168.1.1 or ip.addr ne 192.168.1.1
>	gt	Greater than	frame.len > 64
<	lt	Less than	frame.len < 1500
>=	ge	Greater than or equal to	frame.len >= 64
<=	le	Less than or equal to	frame.len <= 1500
	Is present	A parameter is present	http.response
	contains	Contains a string	http.host contains cisco
	matches	A string matches the condition	http.host matches www.cisco.com

C-like syntax	Shortcut	Description	Example
`&&`	`and`	Logical AND	`ip.src==10.0.0.1 and tcp.flags.syn==1` all SYN flags sent from IP address `10.0.0.1` practically—all connections opened (or tried to be opened) from `10.0.0.1`
`||`	`or`	Logical OR	`ip.addr==10.0.0.1 or ip.addr==10.0.02` all packets going in or out the two IP addresses
`!`	`not`	Logical NOT	`not arp and not icmp` all packets that are neither ARP nor ICMP packets

Address format	Syntax	Example
Ethernet (MAC) address	`eth.addr == xx:xx:xx:xx:xx:xx` where `x` is `0` to `f`	`eth.addr == 00:50:7f:cd:d5:38`
	`eth.addr == xx-xx-xx-xx-xx-xx` where `x` is `0` to `f`	`eth.addr == 00-50-7f-cd-d5-38`
	`eth.addr == xxxx.xxxx.xxxx` where `x` is `0` to `f`	`eth.addr == 0050.7fcd.d538`
Broadcast MAC address	`Eth.addr == ffff.ffff.ffff`	
IPv4 host address	`ip.addr == x.x.x.x` where `x` is `0` to `255`	`Ip.addr == 192.168.1.1`
IPv4 network address	`ip.addr == x.x.x.x/y` where `x` is `0` to `255`, `y` is `0` to `32`	`ip.addr == 192.168.200.0/24` (all addresses in the network `192.168.200.0` mask `255.255.255.0`)
IPv6 host address	`ipv6.addr == x:x:x:x:x:x:x:x` `ipv6.addr == x::x:x:x:x` where in the format of `nnnn`, `n` is `0` to `f` (hex)	`ipv6.addr == fe80::85ab:dc2e:ab12:e6c7`

ARP filters:

- Display only ARP requests:
  - `arp.opcode == 1`
- Display only ARP responses:
  - `arp.opcode == 2`

IP and ICMP filters:

- Display only packets from specific IP addresses:
  - `ip.src == 10.1.1.254`
- Display only packets that are not from a specific address:
  - `!ip.src == 64.23.1.1`
- Display only packets between two hosts:
  - `ip.addr == 192.168.1.1 and ip.addr == 200.1.1.1`
- Display only packets that are sent to multicast IP addresses:
  - `ip.dst == 224.0.0.0/4`
- Display only packets coming from network `192.168.1.0/24` (mask `255.255.255.0`):
  - `ip.src==192.168.1.0/24`
- Display only IPv6 packets to/from specific addresses:
  - `ipv6.addr == ::1`
  - `ipv6.addr == 2008:0:130F:0:0:09d0:666A:13ab`

Complex filters:

- Packets from network `10.0.0.0/24` to a website that contains the word `packt`:
  - `ip.src == 10.0.0.0/24 and http.host contains "packt"`
- Packets from networks `10.0.0.0/24` to websites that end with `.com`:
  - `ip.addr == 10.0.0.0/24 and http.host matches ".com$"`
- All broadcasts from source IP address `10.0.0.3`:
  - `ip.src == 10.0.0.0/24 and eth.dst == ffff.ffff.ffff`
- All broadcasts that are not ARP requests:
  - `not arp and eth.dst == ffff.ffff.ffff`
- All packets that are not ICMP and not ARP:
  - `!arp || !icmp or not arp&&not icmp`

TCP and UDP port number display filters

For TCP or UDP port numbers, use the following display filters:

- `tcp.port == <value>` or `udp.port == <value>` for specific TCP or UDP ports (source or destination)
- `tcp.dstport == <value>` or `udp.dstport == <value>` for specific TCP or UDP destination ports
- `tcp.srcport == <value>` **or** `udp.srcport == <value>` for specific TCP or UDP destination ports

- `tcp.flags`: These filters are used for finding out if flags are set or not:
  - `tcp.flags.syn == 1` to check if the SYN flag is set
  - `tcp.flags.reset == 1` to check if the RST flag is set
  - `tcp.flags.fin == 1` to check if the FIN flag is set
  - `tcp.window_size_value < <value>` to look for small TCP window sizes that are, in some cases, an indication of slow devices

---

Some examples of filters in TCP/UDP filters are as follows:

- All packets to the HTTP server:
  - `tcp.dstport == 80`
- All packets from network `10.0.0.0/24` to HTTP server:
  - `ip.src==10.0.0.0/24 and tcp.dstport == 80`

- All packets from `10.0.0.5` to the DNS server:
  - `ip.src == 10.0.0.5 && udp.port == 53`
- All packets in TCP or protocols in TCP (for example HTTP) that contain the string `cacti` (case-sensitive):
  - `tcp contains "cacti"`
- All packets from `10.0.0.3` that are retransmitted:
  - `ip.src == 10.0.0.3 and tcp.analysis.retransmission`
- All packets to any HTTP server:
  - `tcp.dstport == 80`

Some display Filter Comparison Operators

ip.dst == computer IP

ip.src == computer IP

tcp.port == 80

type http.request in the filter box you will get HTTP GET

type http.response in the filter box you will get HTTP OK

By default the value of the time column in the packet listing window is the amount of time in seconds

To display the time field in Time-of-day format

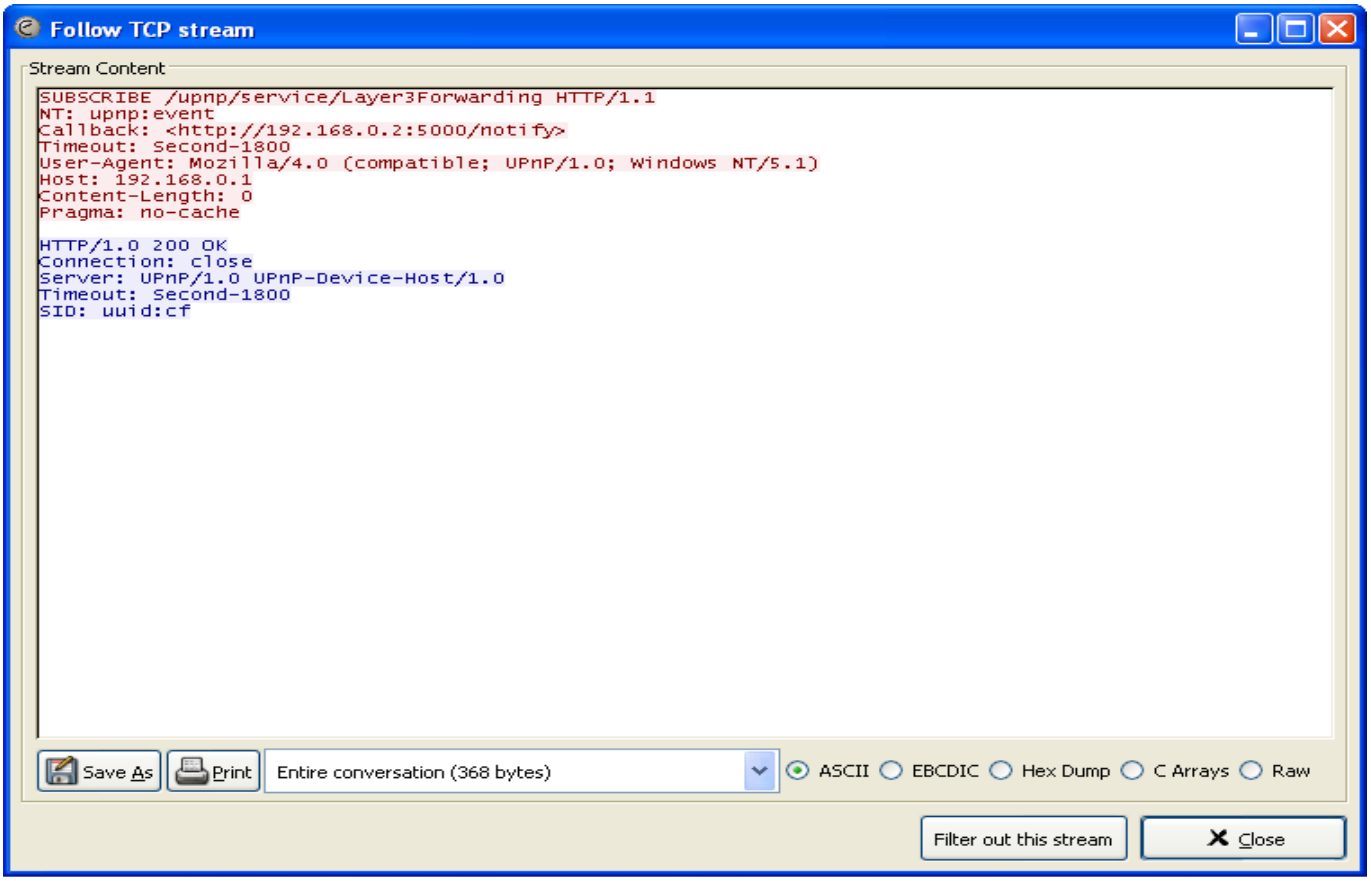
Select View ---→ Time display format ---→ Time –of-day

Refer to the following table for sample capture filters:

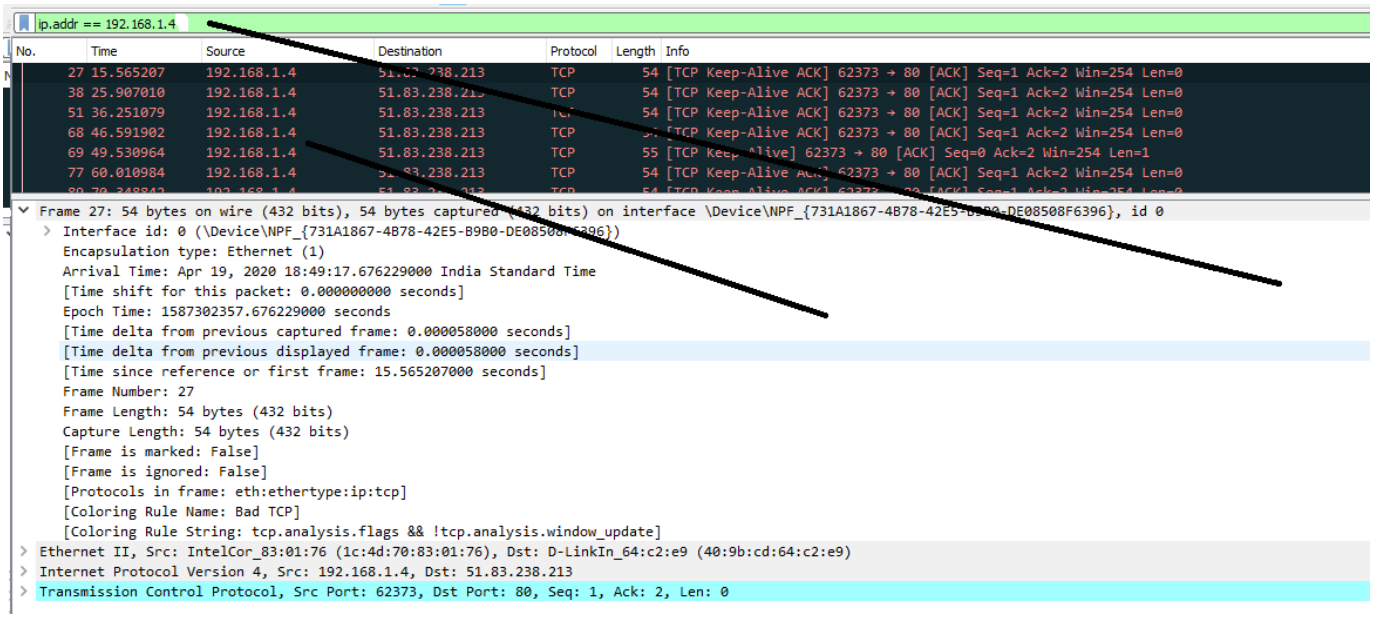
Filters	Description
host192.168.1.1	All traffic associated with host 192.168.1.1
port 8080	All traffic associated with port 8080
src host192.168.1.1	All traffic originating from host 192.168.1.1
dst host 192.168.1.1	All traffic destined to host 192.168.1.1
src port 53	All traffic originating from port 53
dst port 21	All traffic destined to port 21
src192.168.1.1and tcp port 21	All traffic originating from 192.168.1.1and associated with port 21
dst192.168.1.1or dst 192.168.1.2	All traffic destined to 192.168.1.1or destined to host 192.168.1.2
not port 80	All traffic not associated with port 80
not src host 192.168.1.1	All traffic not originating from host 192.168.1.1
not port 21 and not port 22	All traffic not associated with port 21 or port 22
tcp	All tcp traffic
Ipv6	All ipv6 traffic
tcp or udp	All TCP or UDP traffic
host www.google.com	All traffic to and from Google's IP address
ether host 07:34:AA:B6:78:89	All traffic associated with the specified MAC address

Following TCP streams

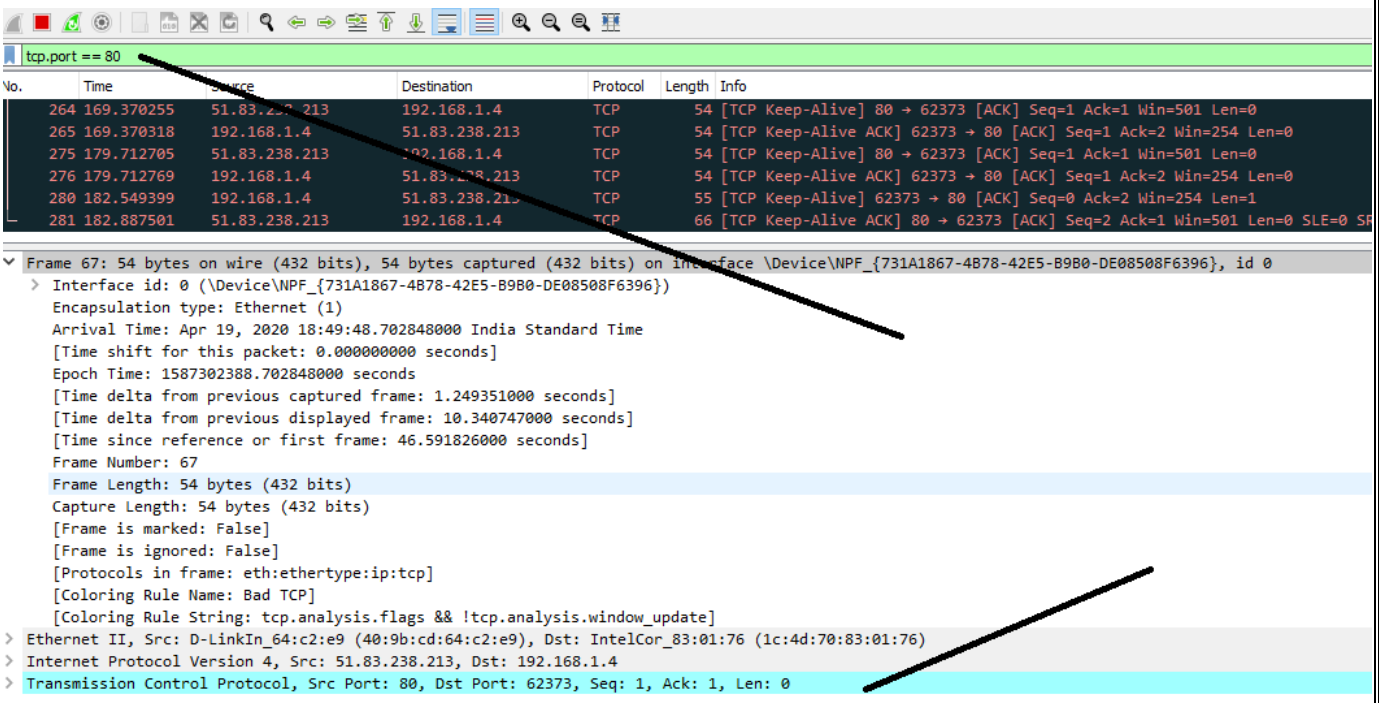
If you are working with TCP based protocols it can be very helpful to see the data from a TCP stream in the way that the application layer sees it. Perhaps you are looking for passwords in a Telnet stream, or you are trying to make sense of a data stream. Maybe you just need a display filter to show only the packets of that TCP stream. If so, Wireshark's ability to follow a TCP stream will be useful to you. It is worthwhile noting that Follow TCP Stream installs a display filter to select all the packets in the TCP stream you have selected. **The "Follow TCP Stream" dialog box**



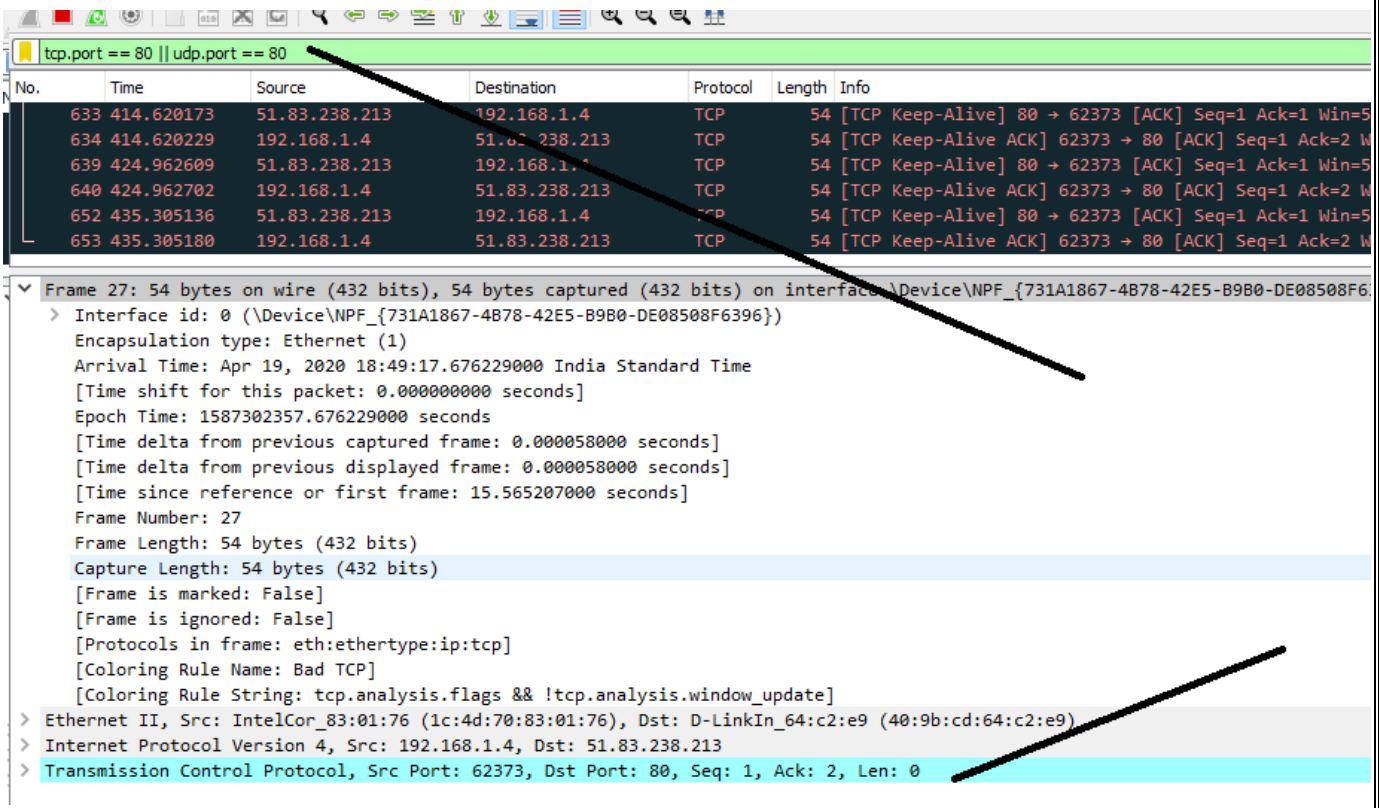
Filter the particular IP address (192.168.1.4)



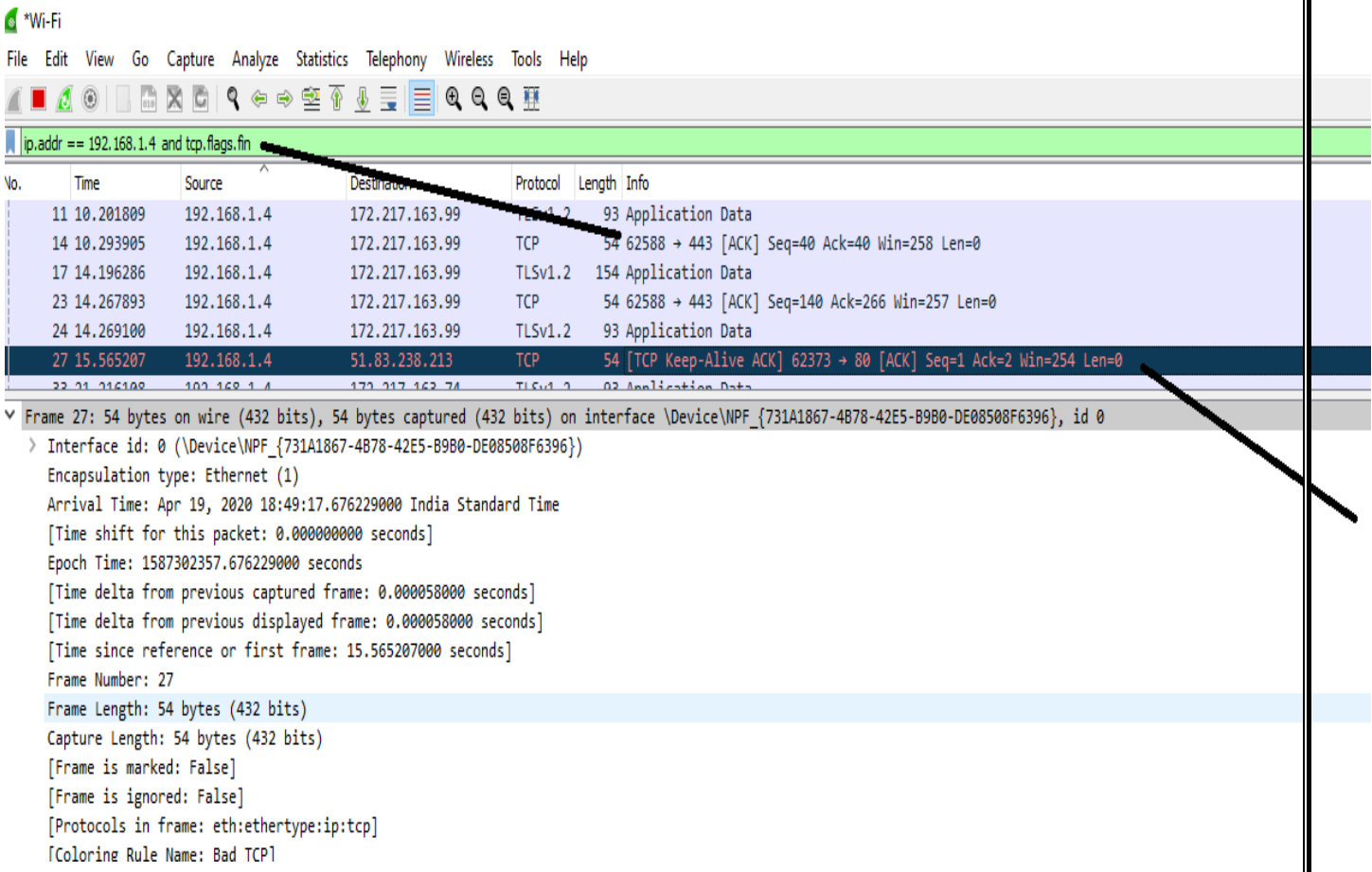
Filter and Display tcp port= 80



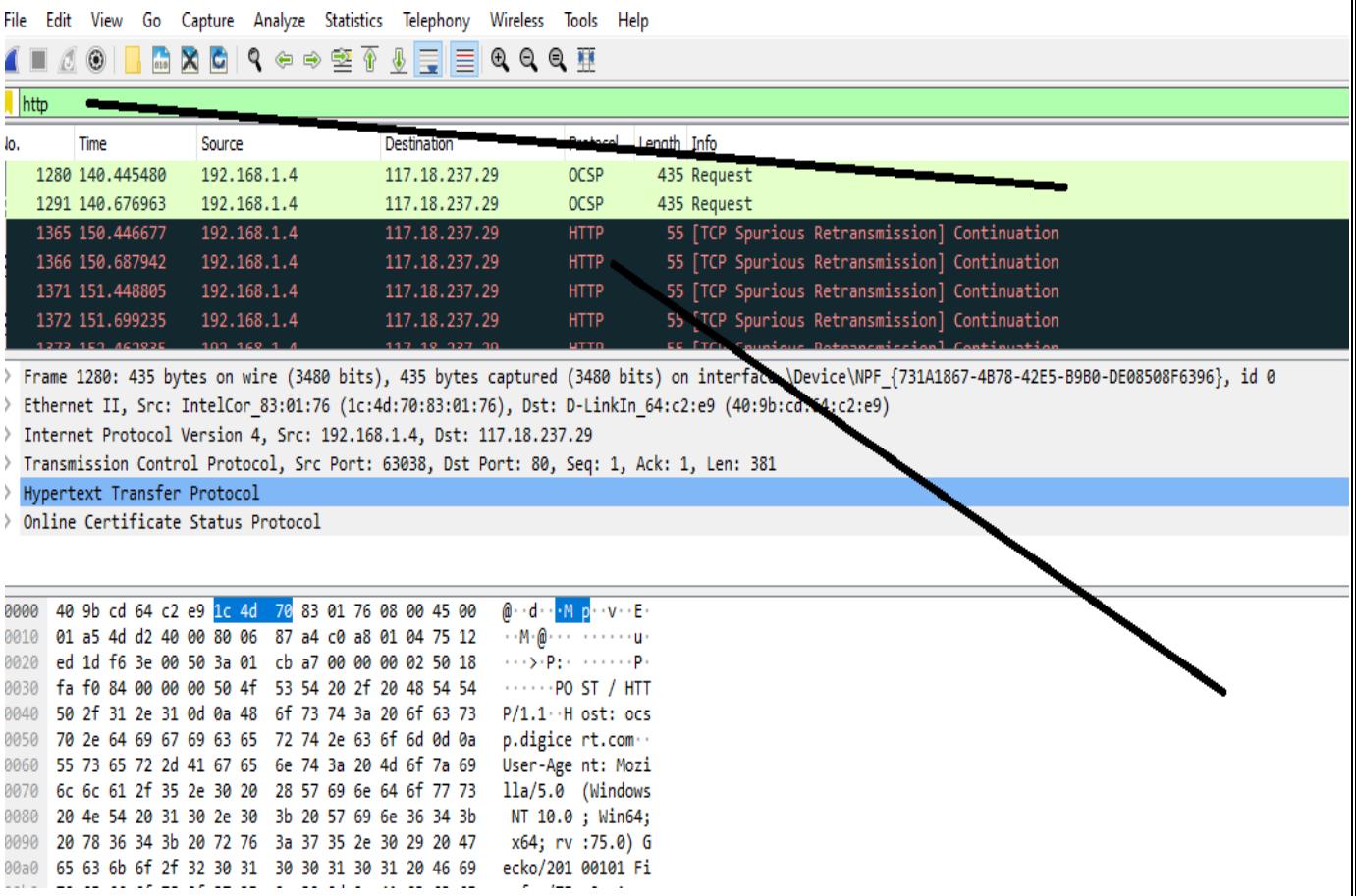
Display tcp port =80 or UDP port =80



Filter the Packet of IP address 192.168.1.4 and tcp.flags.fin



Filter the Http protocol


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
lo. Time Source Destination Protocol Length Info
1280 140.445480 192.168.1.4 117.18.237.29 OCSP 435 Request
1291 140.676963 192.168.1.4 117.18.237.29 OCSP 435 Request
1365 150.446677 192.168.1.4 117.18.237.29 HTTP 55 [TCP Spurious Retransmission] Continuation
1366 150.687942 192.168.1.4 117.18.237.29 HTTP 55 [TCP Spurious Retransmission] Continuation
1371 151.448805 192.168.1.4 117.18.237.29 HTTP 55 [TCP Spurious Retransmission] Continuation
1372 151.699235 192.168.1.4 117.18.237.29 HTTP 55 [TCP Spurious Retransmission] Continuation
Frame 1280: 435 bytes on wire (3480 bits), 435 bytes captured (3480 bits) on interface \Device\NPF_{731A1867-4B78-42E5-B9B0-DE08508F6396}, id 0
Ethernet II, Src: IntelCor_83:01:76 (1c:4d:70:83:01:76), Dst: D-LinkIn_64:c2:e9 (40:9b:cd:64:c2:e9)
Internet Protocol Version 4, Src: 192.168.1.4, Dst: 117.18.237.29
Transmission Control Protocol, Src Port: 63038, Dst Port: 80, Seq: 1, Ack: 1, Len: 381
Hypertext Transfer Protocol
Online Certificate Status Protocol
0000 40 9b cd 64 c2 e9 1c 4d 70 83 01 76 08 00 45 00 @··d···M p··v··E·
0010 01 a5 4d d2 40 00 80 06 87 a4 c0 a8 01 04 75 12 ··M·@··· ·····u·
0020 ed 1d f6 3e 00 50 3a 01 cb a7 00 00 00 02 50 18 ···>·P:· ·····P·
0030 fa f0 84 00 00 00 50 4f 53 54 20 2f 20 48 54 54 ·····PO ST / HTT
0040 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 6f 63 73 P/1.1··H ost: ocs
0050 70 2e 64 69 67 69 63 65 72 74 2e 63 6f 6d 0d 0a p.digice rt.com··
0060 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 User-Age nt: Mozi
0070 6c 6c 61 2f 35 2e 30 20 28 57 69 6e 64 6f 77 73 lla/5.0 (Windows
0080 20 4e 54 20 31 30 2e 30 3b 20 57 69 6e 36 34 3b NT 10.0 ; Win64;
0090 20 78 36 34 3b 20 72 76 3a 37 35 2e 30 29 20 47 x64; rv :75.0) G
00a0 65 63 6b 6f 2f 32 30 31 30 30 31 30 31 20 46 69 ecko/201 00101 Fi