INDEX

CS19541 - COMPUTER NETWORKS

NAME: Caroline Suja. J.S STD: ..... SEC.: ..... ROLL NO.: 220710 48

S.No.	Date	Title	Page No.	Teacher's Sign/ Remarks
1.	13/7/24	Study of Various Network commands	9	
2.	27/7/24	Study of Network cables	10	
3.	30/7/24	Experiments on CISCO Tracer packet.	15	
4.	17/8/24	Setup and configure LAN using a switch and ethernet cable		
5.	17/8/24	Experiment on packet capture tool wireshark		
6.	21/8/24	Hamming code - Error detection & correction		
7.	3/9/24	Sliding window Protocol		
8.	21/9/24	a) Simulate Virtual LAN configuration using CISCO packet tracer		
		b) Configuration of wireless LAN using CISCO packet tracer		
9.	24/9/24	Implementation of subnetting in CISCO packet Tracer Simulator		
10.	28/9/24	a) Inter networking with router's in CISCO Packet Tracer		
		b) Enter network using wireless router DHCP server cloud.		
11.	1/10/24	a) Simulate static routing configuration		
		b) Simulate RIP using CISCO packet Tracer.		
12.	5/10/24	a) Implement echo client server using TCP/UDP sockets.		
		b) Implement echo file chat client server using TCP/UDP sockets		
13.	8/10/24	Implementing ping Program		
14.	12/10/24	Implement packet sniffing using RAW sockets code		
15.	15/10/24	Using webalizer for web log analysis.		
		completed		
		2024		

Palaniappa

13/7/24 EXP: 1

AIM:
Study of various network commands used in Linux and Windows

BASIC NETWORKING COMMANDS:

1. arp -a: IP address of computer

Interface: 172.16.75.83 --- 0x13

Internet address	Physical address	Type
172.16.72.1	7c-5a-1c-cf-be-41	dynamic
172.16.72.133	4c-ac-a3-65-97-f3	dynamic
172.16.73.255	ff-ff-ff-ff-ff-ff	static

2. hostname : Name of your computer

BH007

3. ipconfig/all : detailed configuration information

Windows IP Configuration

Hostname . . . . . : BH007
Primary DNS Suffix . . . . . :
Node Type . . . . . : Hybrid
IP Routing Enabled . . . . . : No
WINS Proxy Enabled . . . . . : No

4. nbstat -a: helps solve problems with NetBios name resolution
Displays Protocol statistics and current TCP/IP connections using NBT (NetBios over TCP/IP)

NBTSTAT [[-a RemoteName][-A IP Address][-c][-n]
[-r][-R][-RR][-s][-S][Interval]]

-a (adapter status) Lists the remote machine's name table given its name
-A (Adapter status) Lists the remote machine's name table given its IP Address

5. netstat : variety of statistics about a computers active TCP/IP connections

Interface List

18...20 88 10 86 79 89 ..... Intel(R) Ethernet Connection (17) I219-LM
12...4e 82 a9 78 8c 65 ..... Microsoft Wifi Direct Virtual Adapter #5
10...42 82 a9 78 8c 65 ..... Microsoft Wifi Direct Virtual Adapter #6
1 ..... Software Loopback Interface 1

6. nslookup : Used to perform DNS lookups

```
Server:  Unknown
Address:  172.16.72.1

Non-authoritative answer:
Name:    www.google.com
Addresses:  2404:6800:4007:81e::2004
          142.250.183.228
```

7. pathping : combination of Ping and Tracert commands

Usage : pathping [-g host-list] [-h maximum_hops] [-i address] [-n]
[-p period] [-q num_queries] [-w timeout] [-4] [-6] target-name

Options:

-g host-list Loose source route along host-list
-h maximum_hops Maximum number of hops to search for target
-n Donot resolve addresses to host names

8. ping : test connectivity between two nodes

Usage: ping [-t] [-a] [-n count] [-l size] [-f] [-i TTL] [-v TOS]
[-r count] [-s count] [[-j host-list] | [-k host-list]]
[-w timeout] [-R] [-S srcaddr] [-c compartment] [-p]
[-4] [-6] targetname

Options:

-t Ping the specified host until stopped
-a Resolve adresses to hostname
-n count Number of echo requests to send

9. Route : show the IP routing table
Manipulates network routing tables

ROUTE [-f] [-p] [-4|-6] command [destination]
[MASK Netmask] [gateway] [METRIC metric] [IF interface]

-f clears the routing tables of all gateway entries
-p When used with ADD command makes a route persistent across
boots of the system
-4 Force using IPV4
-6 Force using IPV6

IMPORTANT LINUX NETWORKING COMMANDS

1 ip : show address information, manipulate routing, plus display network various devices, interfaces and tunnels

ip <OPTIONS> <OBJECT> <COMMAND>

a) To show IP addresses assigned

#ip address show

1: lo: <LOOPBACK, UP, LOWER-UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 172.0.0.1/8 scope host lo
valid-lift forever preferred-lft forever
inet ::1/128 scope host
valid-lft forever preferred-lft forever

b) To assign an IP to an interface

#ip address add 192.168.1.254 /24 dev lo

c) To delete an IP on the interface

#ip address del 192.168.1.254/24 dev lo

d) Alter the status of the interface by bringing it online

#ip link set lo up

e) Alter the status of the interface by bringing it offline

#ip link set lo down

f) Alter the status of the interface by enabling promiscous mode

#ip link set lo promisc on

g) Add a default route

#ip route add default via 192.168.1.254 dev lo

h) Add a route via gateway:

#ip route add 192.168.10/24 via 192.168.1.254

i) Add a route to 192.168.1.0/24 that can be reached on device lo

#ip route add 192.168.1.0/24 dev lo

j) Delete the route for 192.168.1.0/24 via gateway

#ip route delete 192.168.1.0/24 via 192.168.1.254

K) Display the route taken for IP 10.10.1.4

#ip route get 10.10.14

10.10.14 via 172.16.8.1 dev ens20 src 172.16.8.105 uid 0 cache

2. ifconfig

for configuring and troubleshooting networks

lo: flags = 8.29 <UP, LOOPBACK, RUNNING, PROMISC> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10 <host>
loop txqueuelen 1000 (Local Loopback)
RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

3. mtr

mtr is a program with a commandline interface that serves as a network diagnostic tool

mtr <options> hostname / IP

a) Basic mtr command that shows you the statistics

#mtr google.com

localhost.localdomain (0.0.0.0)

Host	Packets			Pings			
	Loss%	Snt	Last	Avg	Best	Wrst	StDev
1. 172.16.8.1	0.0%	42	0.4	0.4	0.1	0.1	0.0
2. static-41.229.249.49	0.0%	42	0.2	0.2	0.2	0.3	0.0
3 142.250.171.162	0.0%	42	6.0	5.0	17.0	7.7	1.1

b) Show numeric ip addresses

#mtr -g google.com

No GTK support sorry

c) Show numeric ip address and hostnames

#mtr -b google.com

localhost.localdomain (0.0.0.0)

Host	Packets			Pings			
	Loss%	Snt	Last	Avg	Best	Wrst	StDev
1. 172.16.8.1	0.0%	150	0.2	1.1	0.1	11.1	2.4
2. 142.250.171.162	0.0%	150	7.3	1.5	6.3	80.5	16.4
3. 142.251.227.217	1.4%	150	8.0	6.2	4.7	100.1	16.4

d) Set the number of pings

#mtr -c 10 google.com

localhost.localdomain (0.0.0.0)

Host	Packets			Pings			
	Loss%	Snt	Last	Avg	Best	Wrst	StDev
1. 172.16.8.1	0.0%	1	0.2	1.1	0.1	11.6	2.1
2. 142.250.171.162	0.0%	1	1.2	1.5	6.3	40.1	2.4
3. 142.251.227.217	1.4%	1	2.4	6.2	4.7	50.2	6.4

4 tcpdump

It is designed for capturing and displaying packets

a. # dnf install -y tcpdump

Last metadata expiration check: 0:00:14 ago on Sat 27 July 2024 12:07:01 PM IST

Package tcpdump-14:4.9.0-2.f126.i686 is already installed skipping

Dependencies resolved

Nothing to do Complete

b. # tcpdump -D

1. enp2s0 [up, running]
2. any (pseudo-device that captures on all interfaces) [up, running]
3. lo [up, running, loopback]
4. wlp3s0 [up]
5. Bluetooth0 (Bluetooth, adapter number 0)

c. # tcpdump -i lo

tcpdump: verbose output suppressed, use -v or -vv for full protocol

d. # tcpdump -i lo -o w

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

e. # tcpdump -i lo -c 10 host 8.8.8.8

tcpdump: verbose output suppressed use -v or -vv for full protocol decode

f. # tcpdump -i lo src host 8.8.8.8

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode, listening on lo linktype EN10MB (Ethernet), capture size 262144 bytes

g. # tcpdump -i lo dst host 8.8.8.8

for outbound traffic going to 8.8.8.8

h. # tcpdump -i lo net 10.1.0.0 mask 255.255.255.0

Capture traffic to and from a network

i. # tcpdump -i lo net 10.1.0.0 124

Capture traffic to from port numbers

j. # tcpdump -i lo port 53

Capture only DNS port 53 traffic

k. # tcpdump -i lo -c 10 host www.google.com and port 443

To capture only HTTPS traffic

l. # tcpdump i lo port not 53 and not 25

To capture all port except port 80 and 25

5 Ping

Verifies IP level connectivity to another TCP/IP computer by sending Internet Control Message Protocol (ICMP) request messages.

a. #ping google.com

PING google.com (142.250.182.14) 56(84) bytes of data
64 bytes from maa05s18-in-f14.1e100.net

b #ping -c 4 google.com

64 bytes from maa05s18-in-f14.1e100.net
(142.250.182.14): icmp-seq=1 ttl=120 time=3.16 ms

Configuring an Ethernet Connection by using nmcli

If you connect a host to the network over ethernet you can manage the connection's settings on the command line by using the nmcli utility

Procedure

1. #nmcli connection show

NAME	UUID	TYPE	DEVICE
wired connection1	a5eb6490-cc20-3668-81f8-031	ethernet	enp1s0

2. #nmcli connection modify "wired connection1"
Rename the connection profile

3 #nmcli connection show
Display current settings

connection.interface-sthame	:enp1s0
connection.autoconnect	:yes
ipv4.method	:auto
ipv6 method	:auto

4. To configure IPV4 settings
#nmcli connection modify "wired connection1" ipv4.method auto

5. #nmcli connection modify "wired connection1"
ipv4.method manual ipv4.addresses 192.0.21/24
ipv4.dns 192.0.2.200 ipv4.dns-search example.com

6. #nmcli connection up Internal-LAN
Activate the profile

RESULT:

Thus the network commands used in linux and windows are studied.

Student observation

1. Which command is used to find the reachability of a host machine from your device?

The ping command in networking is used to test the reachability of a host on an Internet Protocol IP network

2. Which command will be give the details of hops taken by a packet to reach its destination

Trace route command

3. Which command displays the ip configuration of your machine?

ipconfig command

4. Which command displays the TCP port status in your machine?

netstat command

5. Write the modify the ip configuration in a Linux machine?

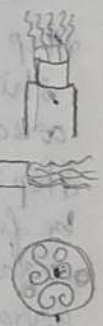
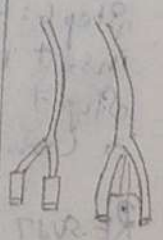
1. Use the 'ipconfig' command followed by the name of your network interface and the new IP Address to be changed on your computer
2. Use the 'ip addr add' command followed by the new ip address and subnet mask
3. Change the ip address in the file '/etc/systemfile/network scripts/ifcfg-Etho' and restart the system.

27/7/24 Exp: 2

AIM:

Study of different types of Network cables

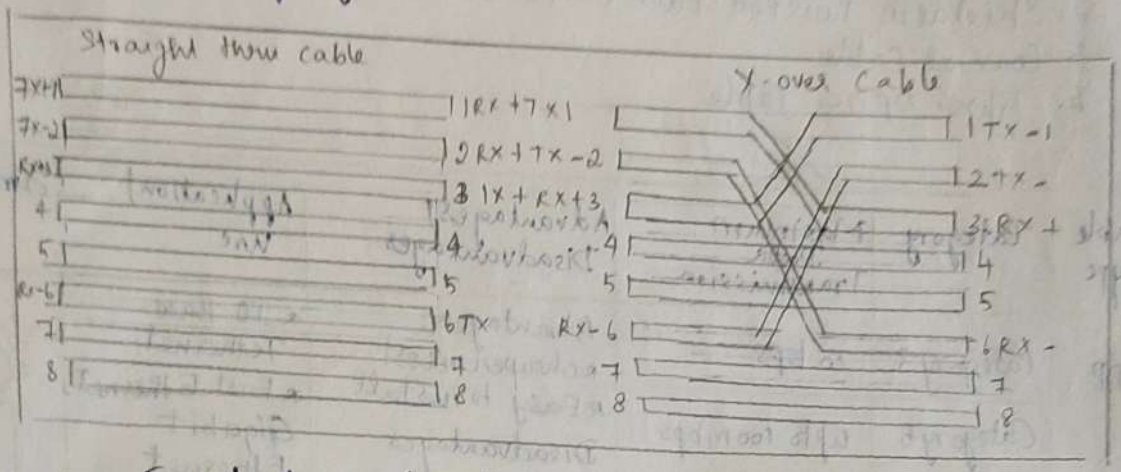
a) Understand different types of network cables
1. Unshielded Twisted Pair (UTP) Cable
2. Shielded Twisted Pair (STP) Cable
3. Coaxial Cable
4. Fibre Optic Cable

Cable type	Category	Maximum Data Transmission	Advantages/ Disadvantages	Application/ Use	Image
UTP	Category 3	10 bps	Advantages * cheaper in cost * Easy to install Disadvantages * More prone to EMI	* 10 Base-Tethernet * Fast Ethernet Gigabit Ethernet	
	Category 5	upto 100 mbps			
	Category 5e	1 Gbps			
STP	Category 6A	10 Gbps	Advantages * shielded * faster than UTP * less prone to noise Disadvantages * Expensive * Greater installation Effort	Gigabit Ethernet 10G Ethernet (<55m) Gigabit Ethernet 10G Ethernet	
ssTP	Category 7				
Coaxial Cable	RG-6 RG-59 RG-11	10-100 Mbps	Advantages * High bandwidth * Immune to interface * Low loss bandwidth Disadvantages * Limited distance * Cost * Size is bulky	Speed of signal is 200m Television network High speed internet connected	
Fibre optic Cable	Single Mode Multi Mode	100 Gbps	Advantages * High speed * High bandwidth * High security * Long distance Disadvantage * Expensive	Maximum distance of fibre optic cable around 100 metres	

b) Make your own Ethernet cross over cable / straight cable

Tools and parts needed:

- Ethernet cabling: CAT5e is certified for gigabit support but CAT5 cabling works as well, just over shorter distances
- A crimpling tool. This is an all in one network tool shaped to push down the pins in the plug and strip and cut the shielding of the cables
- Two RJ 45 plugs
- Optional two plug shields



Step 1: To start construction of the device, begin by threading shields on the cable

Step 2: Next, strip approximately 5 cm of cable shielding from both sides ends. The crimping tool has a round area to complete this task.

Step 3: After you will need to untangle the wires there should be four "twisted pairs" referencing back to the sheet arrange them from top to bottom. One end should be in arrangement A and the other in B

Step 4: Once the order is correct, bunch them together in a line and if there are any that stick out farther than others, clip them back to create an even level. The difficult aspect is placing these into the RJ45 plug without messing up the other

Step 5: Next push the cable right in. The notch at the end of the plug needs to be just over the cable shielding and if it exists that means that you stripped off too much shielding

Step 6: After the wires are securely sitting inside the plug insert it into crimping tool and push down

Step 7: Lastly repeat for the other end using diagram B (to make a crossover cables) / using diagram A (to make straight through cable)

RESULT:

Thus different types of network cables are understood and studied

Student Observation

1. What is the difference between cross cable and straight cable

Straight cable	Cross cable
* The wiring of both ends of the cable is identical	* The transmit and receive wires are crossed on one end of the cable
* It is used for connecting different types of devices Eg: PC to switch/router	* It is used for connecting similar devices Eg: PC to PC

2. Which type of cable is used to connect two PC.? (straight/cross cable)
A cross cable is used to connect two PCs directly

3. Which type of cable is used to connect a router/switch to your PC ? (straight / cross cable)
A straight cable is used to connect a router or switch to a PC

4. Find out the category of twisted pair cable used in your LAN to connect the PC to the network socket ?
You need to physically inspect the ethernet cable connected to your PC. The cable typically has its category printed along the length of the cable sheath
Common categories
cat 5 : Supports up to 100 Mbps
Cat 5e : supports up to 1 Gbps
Cat 6 : supports upto 10 Gbps for shorter distances
Cate 5e or cat6 used mostly.

5. Write your understanding, challenges faced and output received while making a twisted pair straight cable
Understanding : making a twisted pair cable involves arranging wires in a specific order and crimping connectors
challenges faced : Ensuring the wires are in the correct order neatly aligned and fully inserted before crimping can be difficult
Output received : Successfully made cables will allow proper network communication indicated by a functioning network connection between devices.

30/7/24 Exp: 3

AIM:

To study the packet tracer tool installation and user interface overview

c) To understand environment of Cisco packet tracer to design simple network

Introduction:

A simulator, as the name suggests, simulates network devices and its environment. Packet Trace is an exciting network design, simulation and modelling tool.

1. It allows you to model complex systems without the need for dedicated equipment.
2. It helps you to practice your network configuration and trouble shooting skills via computer or an Android or ios based mobile device
3. It is available for both the Linux and windows desktop environments
4. Protocols in packet tracer are coded to work and behave in the same way as they would on real hardware

Installing Packet Tracer:

To download packet Tracer, go to https://www.netcad.com and login with your Cisco Networking Academy credentials Then click on the packet tracer graph and download the package appropriate for your operating system

Windows

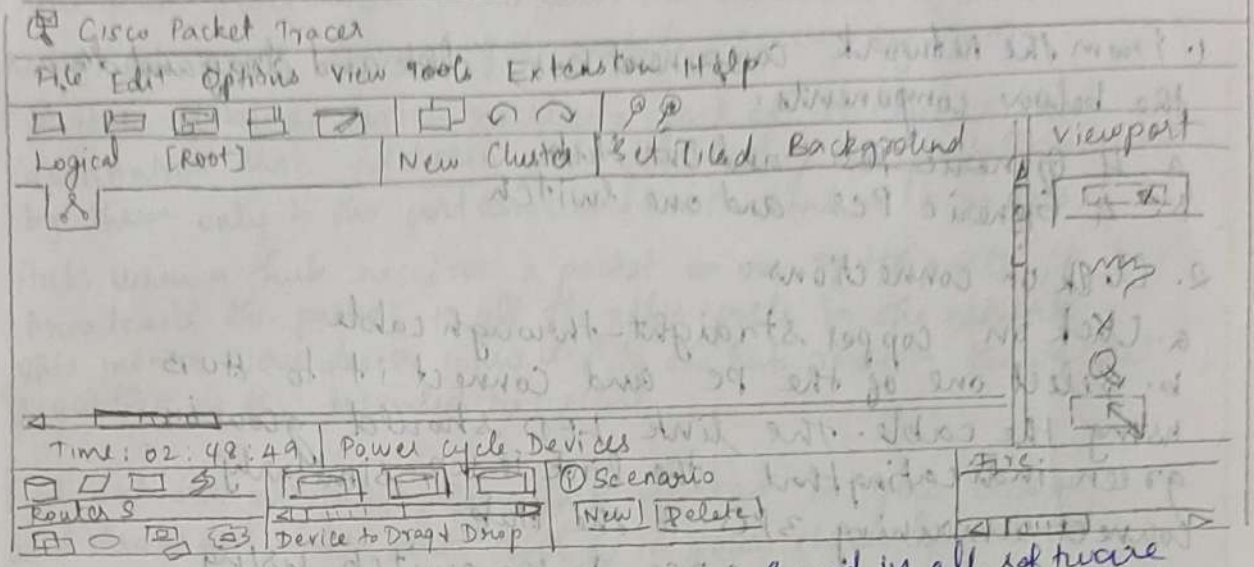
To download Packet Tracer, go to https://www.netacad.com and login with your Cisco Networking Academy Credentials. Then click on the packet tracer graph and download the package appropriate for your operating system.

Windows

Installation in windows is pretty simple and straight forward. The setup comes in a single file named packet tracer - setup 8.0.1 exe

Open this file to begin the setup wizard accept the license agreement choose a location and start the installation

USER INTERFACE OVERVIEW



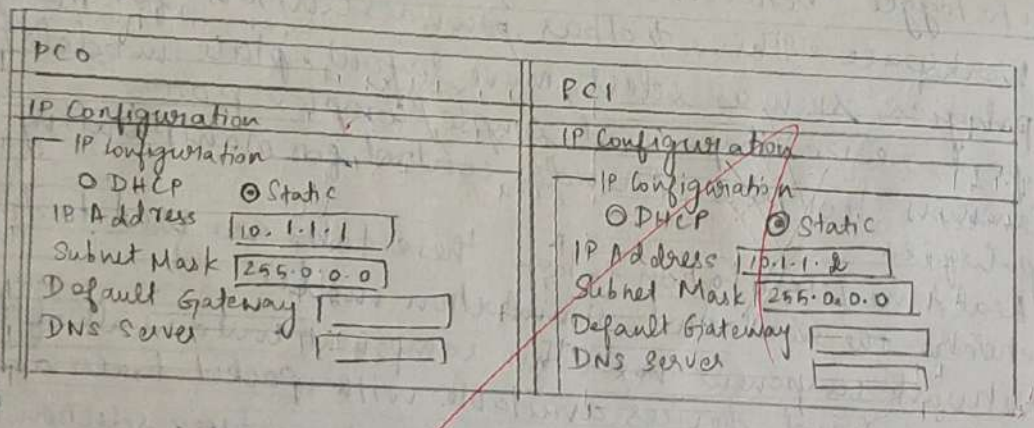
1. Menubar - This is a common menu found in all software applications; it is used to open, saves, print, change and so on
2. Main toolbar - This bar provides shortcut icons to menu options that are commonly accessed, such as open, save, zoom, undo, redo on the righthand side is an icon for entering Network information for the current network
3. Logical / Physical workspace tabs - These tasks allows you to toggle between the Logical and physical work areas
4. Workspace - This toolbar provides controls for manipulating topologies, such as select, move layout, place, note, delete, inspect, resize shape and simple / complex PDU
5. Common tool bars - Provides control for manipulating topologies
6. Real time / simulation tabs - these tabs are used to toggle between the real and simulation modes
7. Network component box - This component contains all of the network and end devices available with packet tracer and is further divided into two areas: Area 7a : Device type - selection box This area contains device categories Area 7b: Device specific selection box when a device category is selected this selection box displays the different device models within that category
8. User created packet box: Users can create highly customized packets to test their topology from this area and the results are displayed as a list

d) Analyse the behaviour of network devices using Cisco packet Tracer simulator

1. From the network component box, click and drag and drop the below components:
   a. 4 Generic PCs and one Hub
   b. 4 Generic PCs and one switch

2. Click on connections:
   a. Click on copper straight-through cable
   b. Select one of the PC and connect it to Hub using the cable. The link LED should grow in green indicating that the link is up, similarly connect remaining 3 PCs to the Hub
   c. Similarly connect 4 PCs to the switch using copper straight through cable

3. Click on PCs connected to hub, go to desktop tab, click on IP configuration and enter an IP address and subnet mask. Here the default gateway and DNS server information is not needed as there are only two end devices in the network



Click on the PDU (message icon) from the common tool bar
a. Drag and drop it on one of the PC and then drop it on another PC connected to the Hub

4. Observe the flow of PDU from source PC to destination PC by selecting the Realtime mode of simulation
5. Repeat steps 3 to steps 5 for the PCs connected to the switch
6. Observe how Hub and switch are forwarding the PDU and write your observation and conclusion about the behaviour of switch and Hub

RESULT:
Thus exp on Cisco packets are understood & executed.

Student observation

a) From your observation write down the behaviour of switch and hubs interms of forwarding the packets received by them.

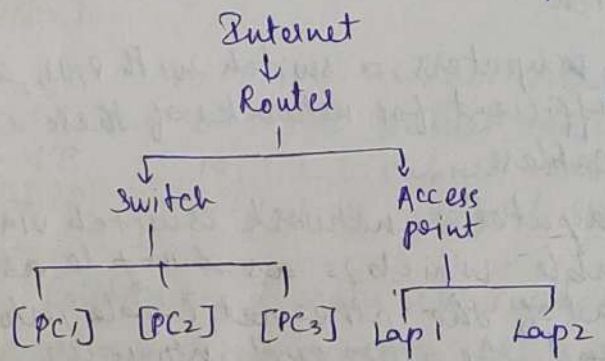
Switch: When a switch receives a packet it checks the destination MAC address and forwarding the packets received by them only to the port associated with that MAC address

Hub: When a hub receives a packet on one of its ports, it broadcasts the packet to all the other ports in the network. This means every device connected to the hub receives the packet regardless of the intended recipient

b) Find out the network topology implemented in your college and draw and label that topology in your observation note.

Star Topology

* All devices are connected to a central switch or hub
* This is one of the most common and widely used topologies in modern networks due to its simplicity and efficiency



17/8/24 EXP: 4

AIM:

Setup and Configure a LAN using a switch and ethernet cables in your lab

What is a LAN?

A LAN connects devices within a limited area like an office or school, allowing users to share resources such as data, printers and internet access. A LAN switch acts as a central device, managing and directing communication between connected devices for fast and secure data transfer

How to setup LAN?

Step 1: Plan and design an appropriate network topology taking into account network requirements + equipment location

Step 2: You can take 4 computers, a switch with 8, 16, 24 ports which is sufficient for networks of these sizes and 4 ethernet cables

Step 3:- Connect your computer to network switch via an ethernet cable which is as simple as plugging one end of the ethernet cable into your computer and the other end into your network switch

Step 4: Assign IP address to your PC's

Log on to the client computer as Administrator or as owner

click Network & Internet connections

Right click Local Area connection / ethernet →

Go to properties → Select Internet protocol (TCP/IPV4)

→ click on Properties → select use the following ip and assign IP address

Internet Protocol version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

- O Obtain IP address automatically
- O Use the following IP address:
  - IP address: 10.1.1.1
  - Subnet mask: 255.0.0.0
  - Default gateway: . . .
- O Obtain DNS server address automatically
- O Use the following DNS server addresses:
  - Preferred DNS server: . . .
  - Alternate DNS server: . . .

☐ Validate settings upon exit

Advanced...

Similarly assign IP address to all the PCs connected to switch

PC1 - IP : 10.1.1.1 subnet mask 255.0.0.0
PC2 - IP : 10.1.1.2 subnet mask 255.0.0.0
PC3 - IP : 10.1.1.3 subnet mask 255.0.0.0
PC4 - IP : 10.1.1.4 subnet mask 255.0.0.0

Step 5 Configure a network switch.

Connect your computer to the switch to access the switch's web interface. You will need to connect your computer to the switch using an ethernet cable

Login to the web interface open a browser & enter the IP address of the switch in the address bar. This should bring up the login page for the switch's web interface. Enter the username & password to login

Configure basic settings: Once you're logged in, you will be able to configure basic settings for the switch

Assign IP address as 10.1.1.15 subnet mask 255.0.0.0

Step 6: Check the connectivity between switch & other machine using ping

Step 7: Select a folder → go to properties → sharing tab → share it with everyone on the same LAN

Step 8: Try to access the shared folder from other computers of the network

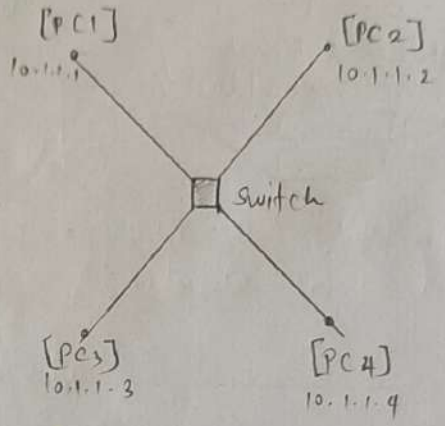
Done

RESULT:

Thus the experiment of setup and configure LAN using ethernet is executed and verified successfully

Student observation

Draw a neat diagram of the LAN in the configuration observation book that you have implemented in your lab
Write the ip configuration of each and every device
Write the outcome and challenges faced while configuring the LAN



LAN was successfully setup and all devices could communicate with eachother using their assigned ip address. Shared resources like folders were accessible from all connected PCs

Challenges Faced

ensuring each PC has a unique IP address to avoid conflict
Initially difficulty accessing the switches web interface due to incorrect IP address entry or logon credentials
Ensuring proper cable connections to avoid loose connections that could lead to network issues
Properly configuring folder sharing permissions to ensure all devices could access shared resources

17/8/24 EXP : 5

AIM:

Experiments on packet capture tool wireshark

Packet sniffer:

Monitors network traffic sent to and from your computers

Captures and display the details of various protocol fields within the data packets

Operates in passivemode

Never transmit packet itself

Doesnot receive packets directly addressed to it.

Obtains copies of all packets

Diagnostic tools:

Tcp dump:

Ex: Tcpdump -enx host 10.129.41.2-10 exe 3-out

wireshark

Ex: wireshark -r exe-3.out

Description:

Wireshark:

It is a network analysis tool that captures & displays network packets in real time. It provides features such as filters and colour coding to help you analyse network traffic and troubleshoot issues effectively.

What can we do with wireshark

Capture network traffic

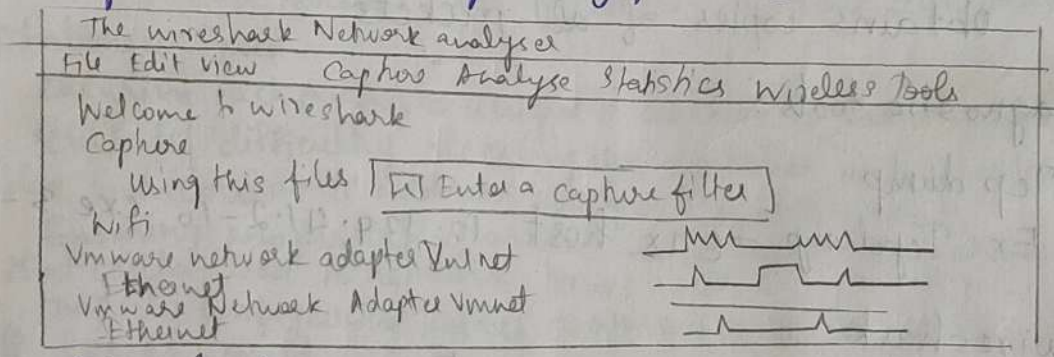
Decode various packet protocols

Apply filters to capture & display specific data
Monitor statistics & analyse problems
Interactivity explore network traffic

Uses:
Network administrators
Security engineers
Developers
Learners

Getting wireshark:
For windows - Download from official website
For linux - Available in package repository

Capturing packets
Launch wireshark
Double click the network interface under capture to start capturing packets



Wire shark Interface Overview

1. Stop capturing traffic
Click red 'stop button' near left corner

2. 'packet list' pane:
Displays all packets in a current capture file
Each file corresponds to one packets
Selecting a line should show more details in packet details and packet bytes panes

3. 'packet details' pane
Shows detailed information of selected packet. Display protocol & its field in a format called tree

4. 'packet bytes' pane:

Shows the selected packets data in hexdump style shows the data of the current packet

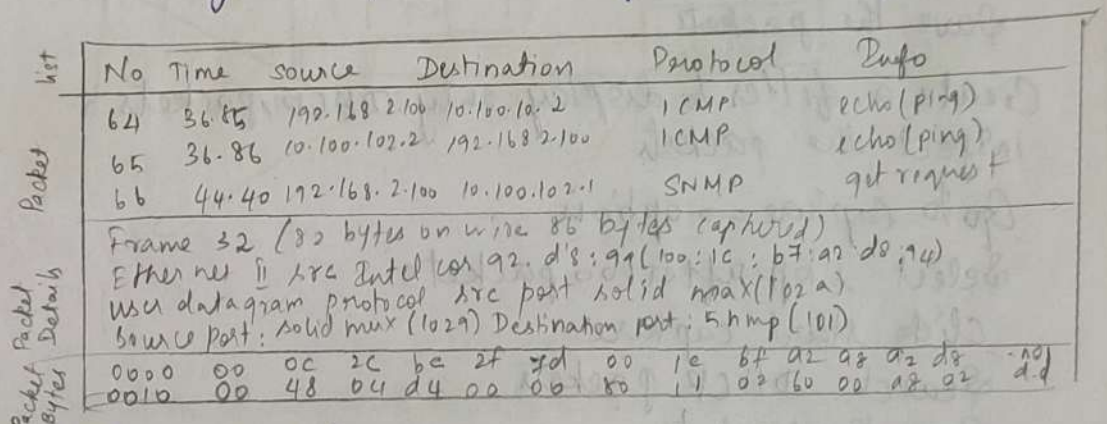
Color coding:

light purple - TCP Traffic
light blue - TCP Traffic
Black - packet with errors

To customize color - Go to view > coloring rules

Sample:

Use sample files to practice in wireshark
open file via → open
Save your capture with file > save, for later review



Filtering Packets:

Apply filters to focus on specific network traffic
Use other apps to isolate traffic for analysis
Type a filter, enter eg: dns
use analyse > display filters to pick or save filters, see the docs for more

Create a filter to display only DNS packets + provide flow graph

Go to capture → option
Select stop capture after 100 packets
Click start capture
Search DNS packets
See flow graph by statistics → flow graph
Save the packets

Create a filter to display only http packets

Go to capture → option
Select stop after 100 packets
Click start capture
Search http packets
Save the packets

Create a filter to display only IP/TCP packets and inspect the packet

Select LAN, go to capture → option
Select stop capture after 100 packets
Click start capture
Search ICMP/IP packets in search bar
Save the packets

Create a filter to display only DHCP, packets & inspect the packets

Go to capture → option
Select stop after 100 packets
Click start capture
Search DHCP packets
Save the packets
Info
Click a packet, choose follow STCP stream to see the full conversation use follow for other protocols

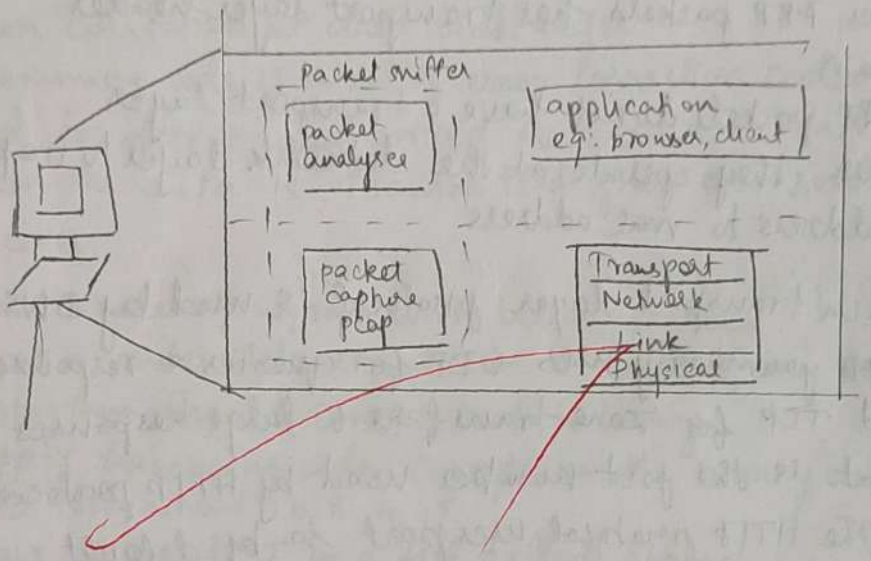
Capturing & Analyzing packets using wireshark tool

To filter, view capture packets, capture 100 packets from the ethernet

Procedure

Select LAN, go to capture → option
select stop captures automatically after 100 packets
then check start capture
Save the packets

Create a filter to display only TCP/UDP packets
Inspects the packets & provide flow graph
Select LAN, go to capture → option
select stop capture after 100 packets
Check start capture
Save the packets



4/2018

RESULT:

Thus the experiments on packet capture tool wireshark is executed succesfully

Student Observation

1. What is promiscous mode?

A network interface in promiscous mode captures all traffic on the network segment regardless of destination address, allowing for comprehensive monitoring or sniffing

2. Does ARP packets has transport layer header explain

ARP packets do not have a transport layer header, they operate at the datalink layer to map IP address to mac address

3. Which transport layer protocol is used by DNS

DNS primarily uses UDP for queries & responses and TCP for zone transfers & large responses

4. What is the port number used by HTTP protocol?

The HTTP protocol uses port 80 by default for communication

5. What is a broadcast IP addresses

It is used to send data packets to all devices on a LAN segment for IPV4 if typically ends in 255 for a given subnet ex: 192.168.1.255

27/8/24

EXP:6

AIM:

Write a program to implement error detection and correction using hamming code concept. Make a test run to input data stream and verify error correction feature

Error correction at data link layer

Hamming code is a set of error correction codes that can be used to detect and correct the errors that can occur when the data is transmitted from the sender to the receiver

Create sender program with below features

1. Input to sender file should be a text of any length
2. Program should convert text to binary
3. Apply hamming code concept on the binary data + add redundant bits to it
4. Save this output in a file called channel

Create a receiver program with below features

1. It should read the input from channel file
2. Apply hamming code on the binary data to check for errors
3. If there is an error, display the position of the error
4. Else remove the redundant bits + convert the binary data to ascii + display the output

Code:

```
import numpy as np

# fn to convert text to binary
def text_to_binary(text)
    return ''.join(format(ord(char), '08b') for char in text)

# fn to convert binary to text
def binary_to_text(binary)
    chars = [binary[i:i+8] for i in range(0, len(binary), 8)]
    return ''.join([chr(int(char,2)) for char in chars])
```

```
# fn to calculate redundant bits
def calc_red_bits(m)
    r=0
    while(2**r < m+r+1):
        r+=1
    return r
# fn to insert redendant bits into the data
def pos_red_bits(data,r):
    j=0
    k=0
    m=len(data)
    res=''
# Adding redundant bits at pos that are powers of 2
    for i in range(1, m+r+1):
        if i==2**j
            res=res+'0'
            j+=1
        else:
            res=res+data[k]
            k+=1
        return res
# fn to calculate parity bits
def cal_parity_bits(arr, r):
    n=len(arr)
    arr=list(arr)
    for i in range(r):
        parity=0
        position=2**i
    for j in range(1, n+1):
        if j & position:
            parity^=int(arr[j-1])
    arr[position-1]=str(parity)
    return ''.join(arr)
```

```
# fn to detect + correct errors
def detect_correct(data, r):
    n = len(data)
    res=0
# calculate parity bits
    for i in range(r):
        parity=0
        position = 2**i
        for j in range(1, n+1):
            if j & position:
                parity ^= int(data[j-1])
        if parity != 0:
            res += position
    if res != 0
    print("error at pos: (res)")
    data = list(data)
    # correct the error
        if res <= n
            data[res-1] = '0' if data[res-1] == '1' else '1'
            print(f" Error corrected at pos :(res)")
        else
            print("error position out of range. No correction")
        Corrected data = ''.join(data)
        return corrected data
        else
            print("No error detected")
            return data

# fn to remove redundant bits
def remove_rdbits(data, r)
    j=0
    original data = ''
    for i in range(1, len(data)+1):
        if i == 2**j:
            j+=1
        else
            original data += data[i-1]
    return original data
```

```
# fn to introduce an error
def introduce_error(data, position):
    if position < 1 or position > len(data):
        print("Error pos is out of range")
    return data
    data = list(data)
    # flip the bit at specified pos
    data[position - 1] = '0'
    if data[position - 1] == '1' else '1'
    print(f"Introduced error at pos : {position}")
    return ''.join(data)

# Sender program
def sender(text):
    binary_data = text_to_binary(text)
    m = len(binary_data)
    r = calc_red_bits(m)
    arr = pos_redundant_bits(binary data, r)
    arr = calc_parity_bits(arr, r)
    print(f'Sender output (binary with redundant bits):
        {arr}")
    return arr

# receiver program
    def reciver(data):
        r = calc_red_bits(len(data))
        corrected data = detect_correct(data, r)
    ascii output = binary to text(original data)
    print(f"decoded text: {asci op}")

# Main program
if name == "main"
input text = Input("Enter the text")

channel data = sender(input text)

Corrupted data = introduce_error(channel data, 2)
receiver(corrupted_data)
```

OUTPUT:

```
Enter the text to be encoded: Caroline
Position of redundant bits [1,2,4,8,16,32,64]
P bit in pos 1: 0        P bit in pos32: 0
P bit in pos 2: 0        P bit in pos64: 0
P bit in pos 4: 0
P bit in pos 8: 1   Binary: 0000100011000110011001011001011100011011
P bit in pos 16: 1
Sender output : 0000110100011011100001011100100101011110110100011010010
Enter the bit pos to introduce error: 5
Introduce error at pos: 5
Error detected at pos: 5
Error corrected at pos: 5 output with redundant bit:
Decoded text: Caroline   01100010110011101000100110011100011
                         011011101
```

RESULT:

Thus Hamming code is executed successfully

19/11.

3/7/24

Exp : 7

AIM:

Write a program to implement flow control at data link layer using sliding window protocol. Simulate the flow of frames from one node to another

Create a sender program with following features
1. Input window size from the user
2. Input a text message from the user
3. Consider 1 character per frame
4. Create a frame with following fields (frame no, data)
5. Send the frames (Print the output on screen)
6. Wait for the acknowledgement from the receiver
7. Read a file called receiver Buffer
8. Check ACK field for the acknowledgement no
9. If the no is as expected, send new set of frames accordingly. Else if NACK is received resend the frames accordingly

Create a receiver file with following features
1. Read a file called Sender_buffer
2. Check the frame no
3. If the frame no are as expected, write the appropriate ACK no in the Receiver_Buffer_file
Else write NACK no in the receiver Buffer file

Code:

```
import time
import random

class Frame:
    def __init__(self, frame_no, data):
        self.frame_no = frame_no
        self.data = data
        self.acknowledged = False
```

```
def send_frames(frames, window_size)
    print("Sending frames"):
    for i in range(window_size):
      if i < len(frames) and not frames[i].acknowledged.
    print(f"Sent Frame {frames[i].frame.no}:
                {frames[i].data}")
    print("Frames sent, waiting for acknowledgement")

Def receive_frames(frames, window_size):
  print("Receiving frames")
    for i in range(window_size):
      if i < len(frames) and not frames[i].acknowledged:
      if random.random()<0.2:
        print(f"received frame {frames[i].frame_no}:
                {frames[i].data} [ERROR]")
      frames[i].acknowledged = False
      else
        print(f"Received frame {frames[i].frame_no}:
                {frames[i].data} received")
        frames[i].acknowledged = True

def sliding window protocol():
    window_size = int(input("Enter window size:"))
    message = input("Enter a message to send:")
    frames = [Frame(i, message[i])
    for i in range(len(message))]

    base = 0
    while base < len(frames):
    send_frames(frames[base:], windowsize)
    time.sleep(2)
    receive_frames(frames[base:], windowsize)
    while base < len(frames) and frames[base].acknowledged
    base += 1
```

```
if base < len(frames):
    print("Resending unacknowledged frames")
    time.sleep(2)
    print("All frames sent & unacknowledged!!")

if __name__ == "__main__":
    sliding_window_protocol().
```

OUTPUT

```
Enter window size: 5
Enter a message to send:

Sending frames
Sent Frame 0: C    Sent frame 2: r    Sent frame 4: l
Sent Frame 1: a    Sent frame 3: o
Frames sent waiting for acknowledgement
Receiving Frames
Received frame 0: C (ok) received frame 2: r(ok) received frame 4: l(ok)
received frame 1: a(ok) received frame 3: o(ok)
Resending unacknowledged frames
Sending frames
Sent Frame 5: l    Sent Frame 7: e
Sent Frame 6: n
Frames sent waiting for acknowledgement
Receiving frames
Received frame 5: l (ok) Received frame 7: e (ok)
Received frame 6 n (Error)
Resending unacknowledged frames
Sending frames
Sent frame 6: n
Frame sent waiting for acknowledgements..
Receiving frames
Received frame 6: n (ok)
all frames sent & acknowledged
```

RESULT

Thus sliding window protocol is executed successfully

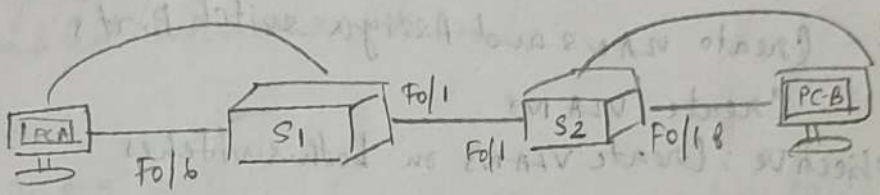
21/9/24

Ex. No: 8a

AIM

Simulate Virtual LAN configuration using Cisco packet tracer Simulation

Packed Tracer – Configure VLANs & TRUNKING



Addressing Table

Device	Interface	IP Address	Subnet Mask	Default Category
S1	VLAN1	192.168.1.11	255.2555.255.0	N/A
S2	VLAN1	192.168.1.12	255.255.255.0	N/A
PC-A	NIL	192.162.10.2	255.255.255.0	192.168.10.1
PC-B	NIL	192.168.10.4	255.255.255.0	192.168.10.1

Part 1

Build the network and configure basic device settings.

Step 1: Build the network

Objective: Connect the devices as shown in the topology

Steps:

Drag switches S1 & S2 to the rack

Drag PC-A and PC-B to the Table and power them on

Connect the devices with copper straight through cables

Connect PC-A to S1 PC-B to S2 using console cables

Step 2: Configure Basic switch settings

Objective: Configure both switches

Steps:

Use the terminal in each PC to console into the switch and enter privileged EXEC mode

Set the device name for each switch

Set the privileged exec password to class

Set the console password and enable login

Set the vty password and enable login

Encrypt plaintext passwords

Step 3: Configure PC hosts
Objective: Assign IP addresses to PC A and PC B from Addressing table
Steps:
In IP configuration input the IP address for PCs
Step 4: Test Connectivity
Objective: Test pings between devices

close configuration window

Part 2 - Create VLANs and Assign switch Ports
Step 1: Create VLANs
Objective: Create VLANs on both switches
Steps:
use the VLAN command on S1 and S2 to create VLANs operations, Parking lot, Management & Native
use the VLAN command on S1 and S2 to create VLANs Operations parking lot, Management & Native
Step 2: Assign VLANs to switch interfaces
Objective Assign port to VLANs
Step
Assign PC A to VLAN 10 (operations)
Remove the management IP addresses from VLAN 1 & configure it on VLAN 99 (management)
Verify with show VLAN brief and show ip interface brief

Part - 3: Maintain VLAN Port Assignments and the VLAN database
Step 1: Assign VLAN to multiple interfaces
Objective: Assign multiple interfaces to VLAN
Steps:
Assign VLAN 99 to interfaces F0/1-24 on S1
Step 2: Remove VLAN assignment from interfaces
Objective: Remove VLAN assignment
Steps
use the no switch port access VLAN command & remove VLAN assignment from F0/24

Part 4: Configure an 802.1Q Trunk between switches

Step 1: Use DTP to initiate trunking

Objective: Configure dynamic Trunking Protocol DTP on interface Fo/1 to negotiate a trunk between S1 and S2

Steps

Set the trunk mode using switch port mode dynamic desirable on S1

Verify using show interfaces trunk & ensure trunking is enabled between S1 & S2

Questions:

1. Can S1 ping S2?
   Yes, if trunking is successfully configured S1 can ping S2
2. Can PC-A ping PC-B?
   Yes, if VLANs are properly configured & trunking is enabled PC-A can ping PC-B

Step 2

Use the command switch port mode trunk to force trunking on Fo/1 on both switches

Final Testing

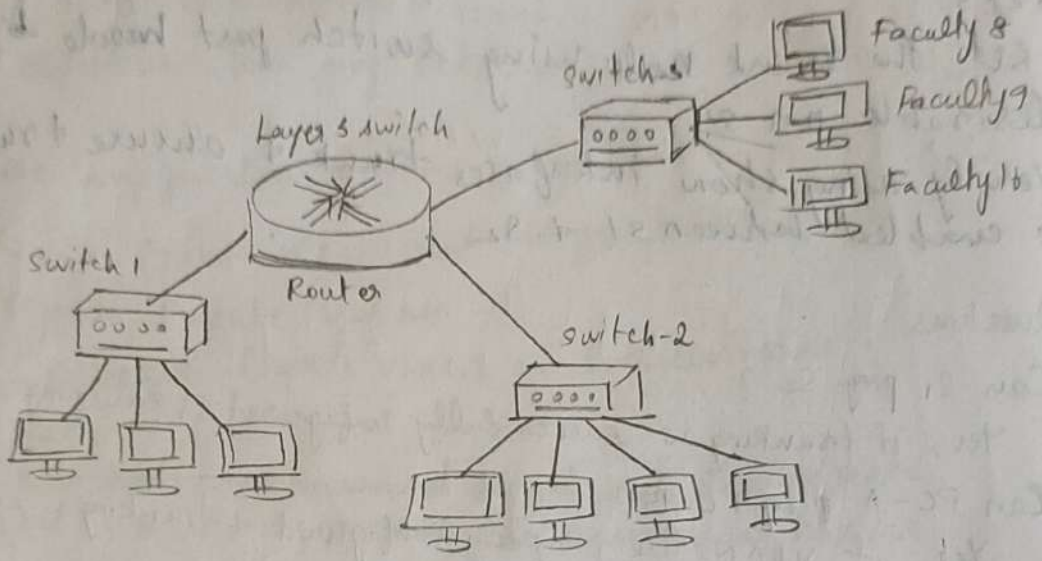
After completing all parts test the connectivity to ensure that the VLAN configuration & trunking are properly functioning.

Reflection Questions

1. What is needed to allow hosts on VLAN 10 to communicate to host on VLAN 99?
   We need a 3 layer device such as a router or a 3 layer switch the inter VLAN routing configured
2. What are the primary benefits that an organisation can receive through effective use of VLANS?
   * Improved network segmentation
   * Enhanced security by isolating traffic
   * Reduced broadcast traffic
   * Better network management & flexibility

Student Observation

a) Draw & Label the VLAN for 10 faculty in Robotics department, sitting in 3 different blocks



b) Show the ip configuration for each device

Faculty-1 : 192.168.10.1/24 Faculty - 2 : 192.168.10.2/24
Faculty-3 : 192.168.10.3/24 Faculty - 4 : 192.168.10.4/24
Faculty-5 : 192.168.10.5/24 Faculty-6 : 192.168.10.6/24
Faculty-7 : 192.168.10.7/24 Faculty-8 : 192.168.10.8/24
Faculty-9 : 192.168.10.9/24 Faculty-10 : 192.168.10.10/24

c) Write the commands for VLAN configuration in switch

```
Switch(config)# Vlan 10
Switch (config-vlan) #name Robotics-VLAN
Switch (config-vlan)# exit
Switch (config) # interface range fa0/1-10
switch(config-if-range) # switchport mode access
switch (config-if-range) # switchport access Vlan 10
switch(config-if-range) # exit
switch(config) # interface fa0/24
switch (config-if) # switchport mode trunk
switch(config-if) # exit
```

RESULT:

Virtual LAN Configuration using Cisco Packet is executed successfully

21/7/24

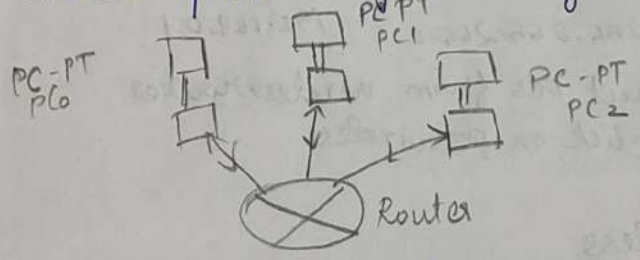
EX. No: 8 b

AIM:

To design a topology with three PCs connected from linksys wireless routers

Procedure:

Configure static IP on PC and wireless router
Set SSID to Mother Network
Set IP address of router to all PC's
Secure your network
Connect your PC by WAP Key



Step1 Click on wireless Router

Setup	wireless Security	Access restriction	Administration

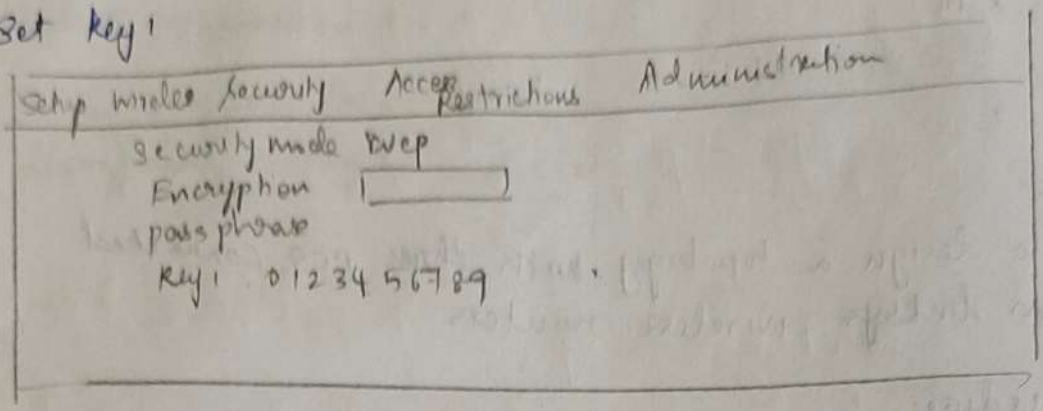
Management — Router Pass: admin
Reenter Pass: admin

Router Access

Setup	wireless	security	Access restriction	Administration

Security mode — Disabled / Disabled / Wep

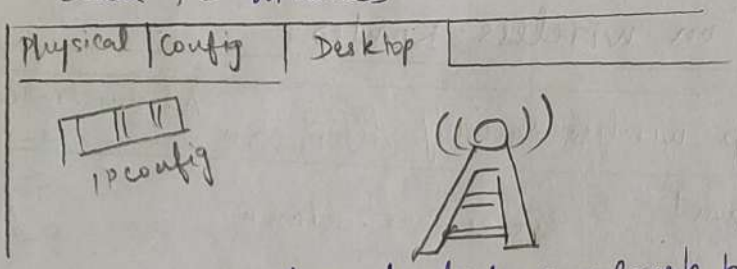
Set key1



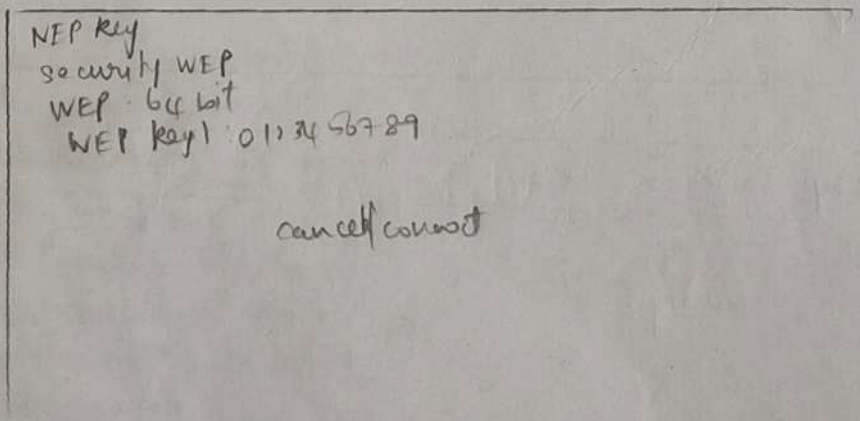
Now configure the static IP on all three PCs and set the subnet Mask

PC	IP	subnet Mask	Default gateway
PC0	192.168.0.2	255.255.255.0	192.168.0.1
PC1	192.168.0.3	255.255.255.0	192.168.0.1
PC2	192.168.0.4	255.255.255.0	192.168.0.1

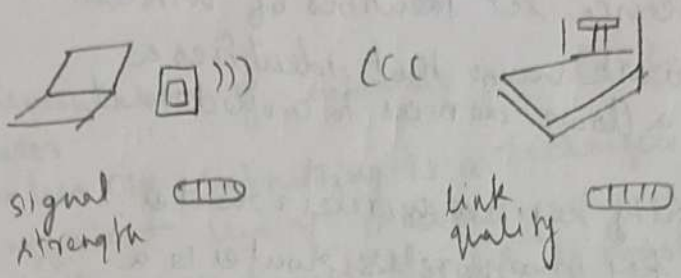
Now its time to connect PCs from wirelessrouter
So click PC select desktop click on PC wireless



Click on connect tab and click on refresh button



Click on connect button to connect Network
It will ask for WAP key



Repeat for all PC's

19/11.

RESULT:
Thus the above exercise is executed and verified successfully

Student observation

c) What is SSID of a wireless router

The SSID service set identifies of wireless router is the network name that identifies a wifi network. It allows devices to connect to it

d) What is a security key in wireless router?

A security key in a wireless router is a password used to protect a wifi network ensuring that only authorized users can connect common types include WEP, WPA, WPA2 keys

e) Configure a simple wireless LAN in your LAB using a real access point & write down the configurations in your network

Configuration:

Access point setup: Connect the access point to power and network then access

eg: 192.168.1.1

set SSID Name your network

Set Configuration:

security mode WPA2-PSK

password: Set a key

save settings: Apply the changes and save the access point

ex:

SSID: cab-wifi

Security key - cabsecure DB

Security mode: WPA2-PSK

IP range 192.168.1.100 to 192.168.1.150

Channel 6

Mode 802.11 b/g/n

Ex: 9
Date: 24/9/24

AIM:
Implementation of subnetting in CISCO Packet tracer Simulator

Classless IP subnetting is a technique that allows for more efficient use of IP addresses by allowing for subnet masks that are not just the default masks for each IP class. This means that we can divide our IP address space into smaller subnets which can be useful when we have limited no of IP addresses but need to create multiple network

Creating a network topology

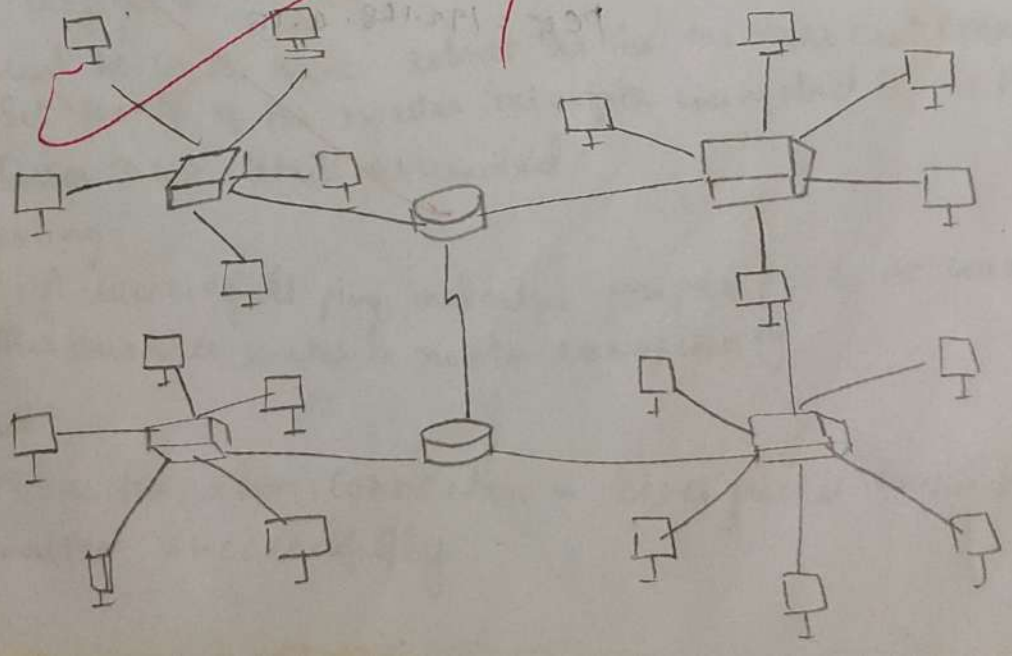
The first step in implementing classless IP subnetting is to create a network topology in packet tracer. To create a network topolog in packet tracer select the new button in the topleft corner the select 'network' & 'generic'

Adding the devices:

Once we have created our network topology we can add devices to it. Here we will be adding routers, switches, PCs To add a device select the device from the bottomleft corner and drag it onto the network topology.

Subnetting

To subnet the network address of 192.168.1.0/24 to provide enough space for at least 5 addresses for end devices, the switch & the router we can use a 27 subnet mask. This will give us a 8 subnet with 30 host addresses each



IP addressing for the network shown in the topology can be as follows

Router R1
Switch S1

Fast Ethernet 0/1
192.168.1.0/27

PC1 : 192.168.1.11
PC2 : 192.168.1.12
PC3 : 192.168.1.13
PC4 : 192.168.1.14
PC5 : 192.168.1.15

Fast Ethernet 0/2
192.168.2.0/27

PC1 : 192.168.2.11
PC2 : 192.168.2.12
PC3 : 192.168.2.13
PC4 : 192.168.2.14
PC5 : 192.168.2.15

Router R2
Switch S2

Fast ethernet 0/1
192.168.3.0/27

PC1 : 192.168.3.11
PC2 : 192.168.3.12
PC3 : 192.168.3.13
PC4 : 192.168.3.14
PC5 : 192.168.3.15

Fast ethernet 0/2
192.168.4/27

PC1 : 192.168.4.11
PC2 : 192.168.4.12
PC3 : 192.168.4.13
PC4 : 192.168.4.14
PC5 : 192.168.4.15

Configuring the devices

Router configuration

Access the CLI Right click on the router and select 'CLI' to open the command line interface

Configure interfaces

Enter enable & configure terminal to begin configuration mode

Fast ethernet 0/0:

Enter interface Ethernet 0/0

Set IP: 'IP Address {IP address} {subnet mask}

Active: no shutdown

Exit: Interface configuration: exit

Fast ethernet 0/1:

Repeat the above steps

Configure Gigabit Ethernet

Use interface gigabit ethernet 0/0 set IP and subnet activate with no shutdown and exit

Switch configuration

Enter enable and configure terminal

Enter interface Fast Ethernet 0/1 then switch port mode access

exit

Repeat for Fast ethernet 0/2 for connecting to the second pc

Pc Configuration

* Must be in the same subnet as the routers fast Ethernet 0/1
* Set the IP of the router interface connected to the PC
* Enter DNS details as needed

Testing

A successfull ping indicates proper pc to pc communication
This ensures router to router connectivity

Result:

Thus the above connection in cisco packet tracer was executed successfully

Student Observation

a Write down your understanding of subnetting

Subnetting is the process of dividing a larger network into smaller manageable subnetworks each subnet operates with its own IP address range helping to organize & manage network traffic

b What is the advantage of implementing subnetting within a network

Efficient IP Management
Improved network performance
Enhanced security

c Find out whether subnetting is implemented in your college

College subnetting

subnet 1 (Admin)	192.168.1.0/24
subnet 2 (library)	192.168.2.0/24
subnet 3 (labs)	192.168.3.0/24

RESULT:

Thus subnetting is executed successfully using CISCO packet

Ex. No: 10 a)
Date: 08/9/24

Inter networking with routers in Cisco Packet Tracer

Aim

To create a simple network with a router connecting two PCs using a copper straight through cable then testing connectivity by sending a PDU from PC0 to PC1

Steps:

Router configuration
Open CH enter privileged mode
Enter global configuration
Configure fast ethernet 0/0
set IP : 192.168.10.1 255.255.255.0
Bring interface up (no shutdown)
Configure fast ethernet 0/1
set IP : 192.168.20.1 255.255.255.0
Bring interface up (no shutdown)

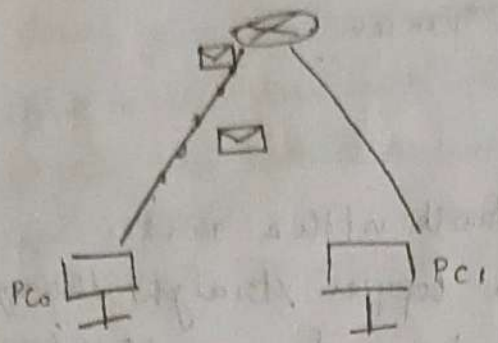
PC configuration
PC0 : IP address : 192.158.10.2
Subnet mask : 255.255.255.0
Default gateway : 192.168.10.1

PC1 : IP address : 192.168.20.2
Subnet mask : 255.255.255.0
Default gateway : 192.168.20.1

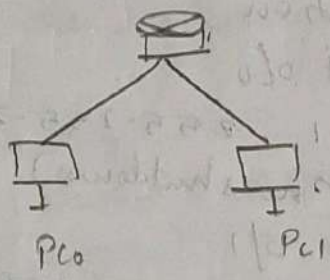
Cable connections
Connect PC0 (fast ethernet0) to router1 (Fast ethernet 0/0)
Connect PC1 (Fast ethernet0) to router1 (Fastethernet 0/1)

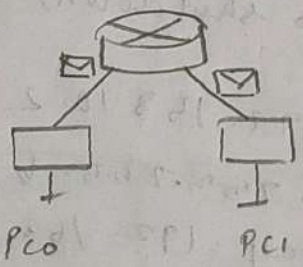
Testing connectivity
Send a PDU from PC0 to PC1 do view Verify functionality

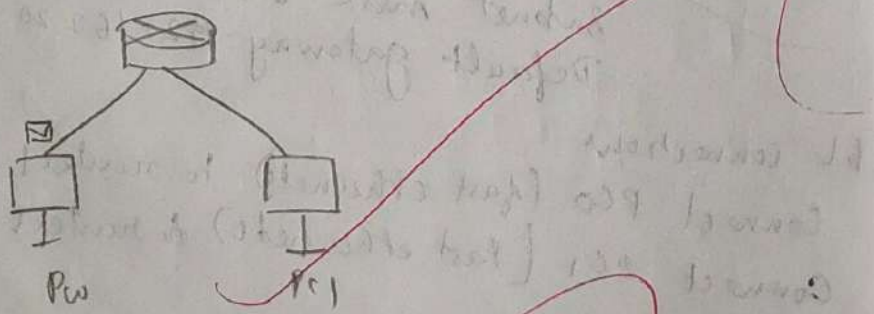
Sending a PDU from PC0 to PC1



Acknowledgement from PC1 to PC0







RESULT:

Thus subneting is executed successfully using cisco packet

Ex. No : 10 b)
Date : 29/9/24
Internetwork using wireless router
DHCP server cloud

Aim:
To design and configure an Internetwork using wireless router DHCP server & Internet cloud

Steps
Launch packet tracer
Build the topology
* Add Devices : In packet tracer, select & place each network device in the workspace according to the topology diagram
* Rename device
* Connect devices with cable
PC to wireless router
Wireless router to cable modem
Cable modem to internet cloud
Cloud to Cisco.com server

Configure the network devices
Step 1: Configure the wireless router
Setup wireless network:
Open the wireless router configuration
In the GUI tab go to the wireless setting
Change the network name to 'Home Network'
Setup internet connection:
In the internet connection its IP service is enabled
Set the DNS server IP Address to 208.67.220.220
Clicks save settings

Steps: Configure the Laptop

Install wireless module

Insert the wireless WPC 300 module & turn the laptop back on

Connect to wireless network

In the desktop tab under PC wireless settings under connect select 'Home Network' from the list of networks and connect

Step: 3

Configure the PC

Enable DHCP

Verify IP address

open command prompt on the PC

Run IP config Call to confirm the PC received an IP address in the 192.168.0.x range

Physical	Config	Desktop	Programming	Attributes
IP Config				
Interface	FastEthernet0			
DHCP	DOstatic		DHCP request	
Subnet mask	255.255.255.0			
Default gateway	192.168.0.1			
DNS server	209.67.220.220			

Step 4: Configure the internet cloud

1. Install network modules (if needed)
- Click the internet cloud icon go to the physical tab
- If missing power off the device and install PT cloud NM 1CX and PT-Cloud-NM-1CFE modules
- Power the device back on

Set from and to ports

- On the Config tab, select cable under connections
- Set from port as coaxial and to port as Ethernet

then click add to establish the connection

Steps: configure the Cisco.com server

Select DHCP from the services in the left panel

Click on the turn the DHCP service on

Pool name: DHCP pool
Default gateway: 208.67.220.220
DNS server: 208.67.220.220
Subnet mask: 255.255.255.255
Max no of users: 50

a) Click add to the pool

b) Configure the cisco.com server as a DNS server to provide, domain name to IPV4 address

While still in the service tab select DNS from the services listed in the left pane

Click on to turn the DNS service on

Name: cisco.com
Type: A record
Address: 208.67.220.220

Click Add to add the DNS service setting

Configure as follows:

Select: static
Gateway: 208.67.220.1
DNS server: 208.67.220.220

Fast Ethernet 0 Interface settings

Select: static
IP Address: 208.67.220.220
Subnet mask: 255.255.255.0

Verify connectivity

Refresh IPV4 settings on PC.

open Desktop > command prompt

Run ipconfig / release and ipconfig/renew to confirm the IP is in the 192.168.0.X range

From the PC's configure prompt issue ping cisco.com to verify connection

19/11

Result:

Thus all the connections are given and executed successfully

Student Observation

1. Write down the key features of configuring wireless router and DHCP server

Configuring wireless router:

SSID configuration: set the SSID for wireless network

Security settings: Configure wireless security protocols (WPA2-WPA3)

DHCP server configuration:-

Ensure DHCP is enabled to assign IP address

Internet connection setup:

Configure WAN settings for Internet connectivity

Configuring DHCP server

IP Address Range: Define the range of IP address

Lease Time: set the duration for which an IP address is valid before renewal

Static IP Assignments: optionally configure static IP address for specific devices

DNS configuration: specify DNS server addresses for devices on the network

What is the significance of DHCP server in internetworking

Significance

Dynamic IP Assignment

Efficient IP Management

Centralised configuration

Network Scalability

Design & configure & internetwork in your lab using switch router & ethernet cables

Internetwork design

Switch connects multiple devices within the local network

Router connects the local network to the Internet / other networks

Eg configuration

Switch IP Address : 192.168.1.1

Router WAN IP Address : obtain from ISP

LAN IP Address : 192.168.1.254

PC1 : IP address : 192.168.1.10

Subnet mask : 255.2555.2555.0

Default Gateway : 192.168.1.254

PC2 : IP Address : 192.168.1.11

Subnet Mask : 255.255.255.0

Default Gateway : 192.168.1.254

Wireless Device (laptop)

Dynamic IP from DHCP : 192.168.1.X

Subnet mask : 255.255.255.0

Default Gateway : 192.168.1.254

Ex. No: 11

Date: 11/10/24

Aim:

a) Simulate static routing configuration using Cisco Packet Tracer

In this static routing configuration using Cisco Packet Tracer, static routes are manually added to a router's routing table for networks not directly connected

Steps

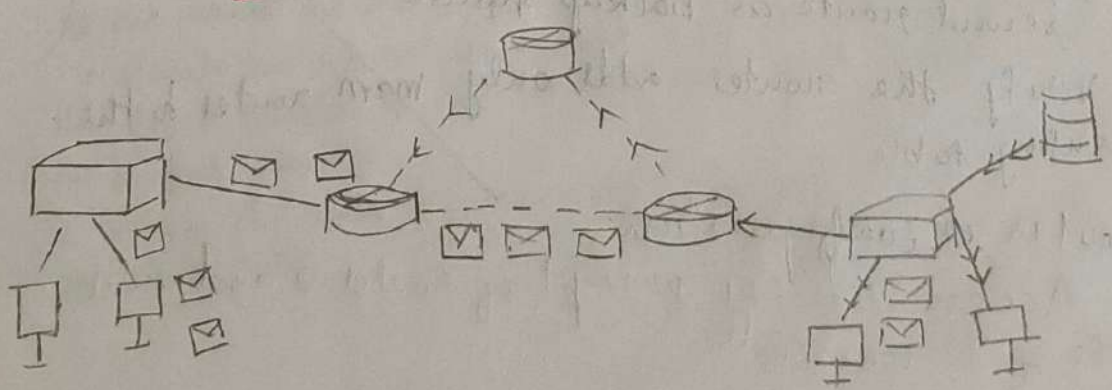
1. Lab setup: create a network with two routers for each network (main + backup)
2. Main and Backup routes: set the route with fewer routers or lower administrative Distance as the main route

Example commands

- ip route 30.0.0.0 255.0.0.0 20.0.0.2.10
- ip route 30.0.0.0 255.0.0.0 40.0.0.2.20

Routing table: use show ip route to verify that the main route is added, and the backup route activates if the main one fails

Static routing is used for networks not directly connected to the routers



The following table lists the connected networks of each router

Router	Available networks on local interfaces	Networks available on other router's interfaces
Router()	10.0.0.018 20.0.0.018 40.0.0.018	30.0.0.018 50.0.0.018
Router 1	20.0.0.018 30.0.0.018 50.0.0.018	10.0.0.018 40.0.0.018
Router 2	40.0.0.018 50.0.0.018	10.0.0.018 20.0.0.018 30.0.0.018

Lets create static router on each router for networks that are not available on the router

Router 0 requirements

- Create two routes for network 30.0.0.018 and configure the first route (via router 1) as the main route and the second route as a backup route
- Create two router for the host 30.0.0.100/8 and configure the first route as the main route and the second route as backup route
- Verify the router adds only main routes to the routing table

Router 0 Configuration

Access the CLI prompt of Router 0 and run the foll commands

Router enable

Router # configure terminal

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

Router (config)# ip router 30.0.0.0-255.0-0.0-20.0.0.210

Router (config)# ip route 30.0.0.0-255.0.0.0 40.0.0-250

Router(config)# ip route 30.0.0.100.255.255.255.255.255.
40.0.0.2.10

Router (config)# ip route 30.0.0.100.255.255.255.255-255
20.0.0.2.20

Router (config)# ip route 50.0-0.0 255.0.0.0 40.0.0.2.10

Router (config)# ip route 50.0.0.0 255.0.0.0 30.0.0.2.20

Router (config)# exit

Router # show ip route static

30.0.0.0/8 is variably subnetted, 2 subnets, 2 masks

S 30.0.0.0/8 [10/0] via 20.0.0.2

S 30.0.0.0/32 [10/0] via 40.0.0.2

S 50.0-0.0/8 [10/0] via 40.0.0.2

Router #

Router 1 requirements

- create two routes for network 10.0.0.0/8 and configure the first route (via route 0) as the main route and the second route as backup
- Create two routes for network 40.0.0.0/8 and configure the first router as the main route and the second route as backup route
- Verify the router words only main route to the routing table

Router 1 configuration

Router > enable
Router # Configure terminal
Enter Configuration commands one per line end with CNTL/Z

Router (config) # ip route 10.0.0.0 255.0.0.0 20.0.0.1
Router (config) # ip route 10.0.0.0 255.0.0.0 50.0.0.2
Router (config) # ip router 40.0.0.0 255.0.0.0 20.0.0.1 10
Router (config) # ip router 40.0.0.0.255.0.0.0 50.0.0 120
Router (config) # exit
Router # show ip route static
S 10.0.0.0/8 [10/0] via 20.0.0.1
S 40.0.0.0/8 [10/0] via 20.0.0.1

Router #

Router 2 requirements

Create static router for network 10.0.0.0/8 and network 30.0.0.0/8 and Verify the router adds both routes to the routing table

Router 2 configuration
Router > enable

Router # configure terminal
Enter configuration commands one per uni end with CNTL/Z
Router (config) # ip route 10.0.0.0 255.0.0.0 40.0.0.1
Router (config) # ip route 30.0.0.255.0.0.0.0 50.0.0.2

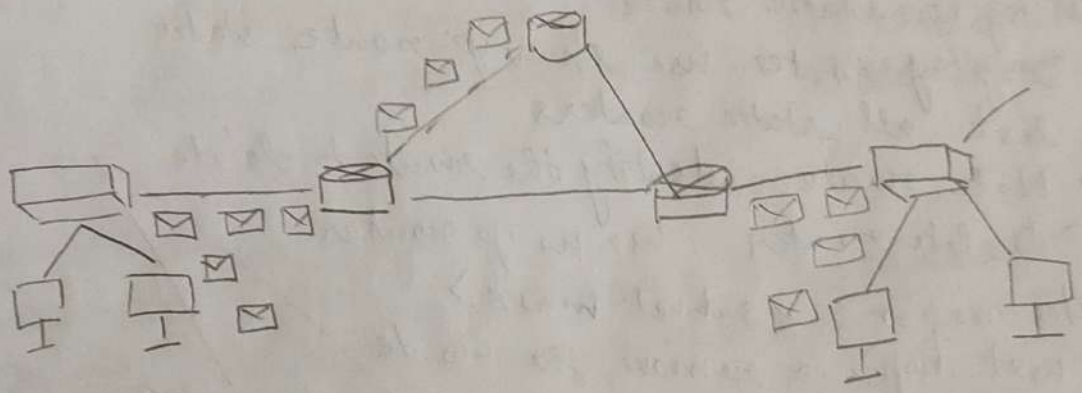
Router (config) # exit
Router # show ip route static
S 10.0.0.0.0/8 [1/0] via 40.0.0.1
S 30.0.0.0/8 [1/0] via 50.0.0.2

Router #

Verifying static routing on router 0

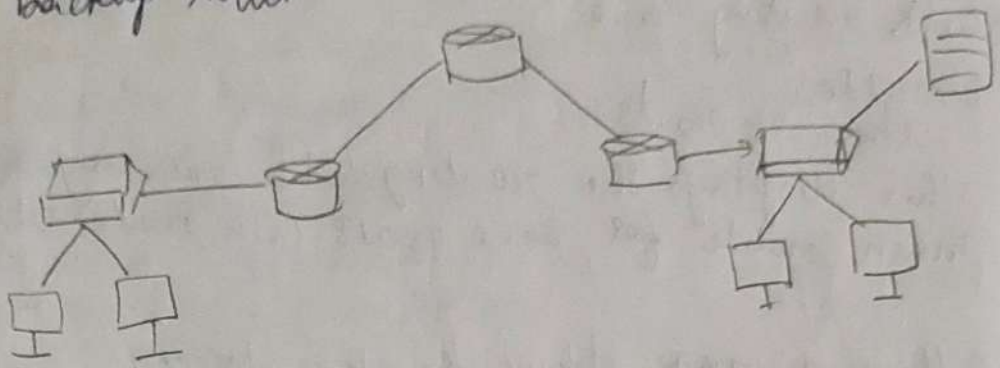
1. Ping and Trace route:
   → use the command:
   tracert 30.0.0.100
   → this path checks to 30.0.0.018 and confirms if the main route via Router 1 is used
2. Check routing table
   1. Execute
   show ip route
   This displays the routing table ensuring only the main route for 30.0.0.018 (via router 1) is listed

The following image shows the above testing



Verifying host route 30.0.0.100/8
→ Ping test. use tracext 80.0.0.100 to check route usage
→ Routing Table: Run show ip route to confirm the static route is present
The following image shows this testing

Verifying Backup route for 30.0.0.0/8
→ Simulate Failure: Disconnect the link between Router 0 and Router 1
→ Ping Test: Use ping 30.0.0.0 to verify the backup route is used
→ Routing Table: Run show ip route to confirm the backup route is active



Deleting a static route:
→ Display Routes: use show ip route static to list all static routers
→ Note route: Identify the route to delete
→ Delete route: use no ip router <destination> <subnet mask> <next hop> to remove the route
After deletion, the backup route becomes the main route automatically

Result:
Successfully simulated static routing in Cisco packet tracer, configured main and backup routes and verified routing table entries

Ex : 11b
Date : 1/10/24

b) AIM:
Simulate RIP using CISCO packet tracer

Initial IP configuration

Device	Interface	IP Configuration	Connected with
PC0	Fast Ethernet	10.0.0.2/8	Router 0's Fa 0/1
Router0	Fa 0/1	10.0.0.1/8	PC0's Fast Ethernet
Router0	S0/1/1	192.168.1.254/30	Router 2's S0/1/1
Router0	S0/1/0	192.168.1.249/30	Router 1's S0/1/0
Router1	S0/1/0	192.168.1.250/30	Router 0's S0/1/0
Router1	S0/1/1	192.168.1.246/30	Router 2's S0/1/0
Router2	S0/1/0	192.168.1.245/30	Router 1's S0/1/1
Router2	S0/1/1	192.168.1.253/30	Router 0's S0/1/1
Router 2	Fa 0/1	20.0.0.1/30	PC1's Fast ethernet
PC1	Fast Ethernet	20.0.0.3/30	Router 2's Fa 0/1

1. Assign IP Addresses to PCs:
Double-click the PC, go to desktop > IP configuration and assign the IP addresses according to the table

2. Assign IP addresses to Router interfaces
- Access router CLI by double clicking the router
- Enter global configuration mode:
Router > enable
Router # configure terminal
- Assign IP Addresses to Fast ethernet and serial interfaces
Router (config) # interface fast Ethernet 0/0
Router (config-if) # ip address [IP] [subnet mask]
Router (config-if) # no shutdown

Command Prompt

Packet Tracer PC command line 1.0

PC ipconfig

Fast ethernet 0 connection . (default port ( )

link - local IPV6 Address	FE80::260:70PB
link - local	20.0.02
Subnet mask	255.0.0.0
Default gateway	20.0.0.1

PC a ping 10.0.02

Pinging 10.0.02 with 32 bytes of data

Request timed out

Reply from 10.0.0.2 : bytes = 32 time = 3ms TTL = 126

Reply from 100.0.2 : bytes = 32 time = 3ms TTL = 126

Reply from 10.0.02 : bytes : 32 time = 3ms TTL = 126

Ping statistics for 10.0.02

Packets sent = 4 received 3 Lost = 1

Approximate round trip times in

milli seconds

Minimum = 3 ms Maximum = 3 Average = 3ms

RIP (Routing Information Protocol) automatically manages routes and switches to an alternative route if one goes down in the current setup

There we two routes b/w PC0 and PC1

Route 1 : Fewer loops, choosen by RIP by default

Route 2 : More loops, used if Route 1 fails

If route 1 goes down (eg by disconnecting) the cable b/w router 0 & router 2) RIP will automatically switch Router 2

You can verify this with the tracert command before and after the disconnection to see how RIP dynamically reroutes traffic

RESULT:

Thus, when Route 1 fails RIP automatically reroutes traffic through Route 2 ensuring continuous connectivity between PC0 & PC1

Ex: 12 a

Date: 5/10/24

AIM:

a) Implement echo client saver using TCP/UDP sockets

TCP echo client-server algorithm

TCP server algorithm

1. Create a TCP socket
2. Bind the socket to a local address & port
3. Listen for incoming client connections
4. Accept a client connect
5. Loop
   Receive data from the client
   If data is received send it back to client
   else break the loop
6. Close the connection

TCP client Algorithm

1. Create a TCP socket
2. Connect to the server using specified address & port
3. Send a message to server
4. Receive the echoed message from the server
5. Display the received message
6. Close the socket

tcp-server.py

```
import socket
def tcp_server()
    server_socket = socket.socket(socket.AF_INET, Socket.
                                  Sock_STREAM)
    server_socket.bind(("localhost", 12345))
    server_socket.listen())
    Print("TCP server is waiting for connection")
    connection, client_address = server_socket.accept()
     Print(f"connected to {client_address}")
```

```
try:
    while True
        data = connection.recv(1024)
        if data:
            print(f"received {data.decode()}")
        else
            break
finally:
    connection.close()
    if __name__ == "__main__"
        tcpserver()
```

tcp_client.py

```
import socket
def tcp_client():
    client_socket = socket.socket(socket.AF_INET, socket.
                                  SOCK_STREAM)
    client_socket.connect(("localhost", 12345))
try:
    message = input("Enter a message to send")
    client_socket.sendall(message.encode())
    data = client_socket.recv(1024)
    print(f"Received from server {data.decode()}")
finally:
    client_socket.close()

if __name__ == "__main__":
    tcp_client()
```

output

> python tcp-client.py

Enter a msg to send : Hi, this is Caroline
Received from server : Hi, this is Caroline

> python tcp_server.py

TCP server is waiting for a connection
connected to ('127.0.0.1', 56893)
Received Hi this is Caroline

Result :

The program for using echo client server using TCP / UDP socket has been executed successfully

20/1

Ex. 12 b
Date : 5/10/24

AIM

To implement the chat client server using TCP/UDP sockets

Algorithm

Chat server

1. Start the server by creating a socket bind to a specific address & port, listener incoming connections
2. When a new client connects add direct a list of a connected client start a new process to talk to the clients
3. For each connected client keep checking for new messages
4. If a client disconnects remove that client from the list & stop talking to that client
5. Keep running the process till the server stops

Chat client

1. connect to the server by creating a socket and connect it to server address & port
2. start a process to listen to messages from the server
3. Keep asking for the new message
4. Keep running till the user decides quit

Chat_client.py

```
import secret
import threading
def receive_message (client_socket):
while True:
    try:
        message = client_socket.recv(1024).decode
                                        ('UTF-8')
if message:
    print(f"Server: {message}"):
exception as e
    print( f"an error occured = {e}")
break
```

```
def start_client()
    client_socket = socket.socket(socket.AF_INET,
                                  socket.SOCK_STREAM)

    host = '127.0.0.1'
    port = 12345
    client_socket.connect((host, port))
    print("Connected to chat server")
    threading.Thread(target=receive_message, args=(client_sock),
        daemon=True).start()

    while True:
        message = input("You: ")
        client_socket.send(message.encode('utf-8'))

if __name__ == "__main__":
    start_client()
```

```
# chat_server.py
import socket
import threading
def handle_client(client_socket):
    while True:
        try:
            message = client_socket.recv(1024).decode
                                        ('utf-8')

            if not message:
                break
            print(f"Received message from client
                                    {message}")
        client_socket.send(response.encode('utf-8'))
    except exception as e:
        print(f"An error has occured")
        break
    client_socket.close()
def start_server():
    server = socket.socket(socket.AF_INET,
                           socket.SOCK_STREAM)
```

```
server.bind(('127.0.0.1',12345))
server.listen(5)
print("chat server has started on 127.0.0.1,12345")
while True:
    client_socket, addr = server.accept()
    print(f"new connection from {addr}")
    client_handler = threading.Thread(target=handle_client,
    client_handler.start() args=(client_socket,))

if __name__ == "__main__":
    start_server()
```

Output

```
>python chat_server.py
chat server started on 127.0.0.1:12345
new connection from (127.0.0.1, 57226)
received from client : Caroline
Type from message to client : Hello
>python chat_client.py
connected to chat server
You: Caroline
You. server : hello
```

RESULT:

Thus the implementation of chat client server using TCP/UDP socket has been successfully executed & verified

Ex. No: 13

Date : 1/10/24

Aim:

Implement your own ping program

Algorithm

ping client.py

* Socket creation
* Then set a timeout of 2 second to ensure that if no response is received it will stop waiting & print 'request time out'
* Send a ping msg to specified host & port
* It listens for a response & calculates the time difference b/w sending & receiving the packet

ping server.py

1. Initialize UDP socket
2. Bind to IP Address & port
3. Listen for incoming messages
4. receive data
5. Send response

Program

ping_server.py

```
import socket
    def start_service(host='127.0.0.1', port=12345)
    with socket.socket(AF_INET, socket.SOCK_DGRAM)
    s.bind((host, port))
    print("UDP server is running on {host}:{port}")

while true
    data, addr = receive(4)
    print(received msg from {addr} {addr}: {decoded})
    s.send to (b, ping add)

if addr_name == "main":
    start_server()
```

Ping_client.py

```
import socket
import time
def ping_server(host='127.0.0.1', port=12345):
    with socket(socket.AF_INET, socket.
                SOCK_DGRAM) as s:

    try:
        s.settimeout(1)
        start = time.time()
        s.sendto(b'Ping', (host, port))
        data, addr = s.recvfrom(1024)
        end = time.time()
        print(f"Received {data.decode()}
        from {addr} in {end-start:.3f}")
    except socket.timeout
        print("Request timed out")
if __name__ == "__main__":
        ping_server()
```

Output:

```
>python ping_server.py
UDP server running on 127.0.0.1:12345
received message from('127.0.0.1', 1573) ping
>python ping_client.py
received ping from ('127.0.0.1', 12345) in 0.03 sec
```

4/2011

Result:

The program to create and implement ping ping message has been successfully executed.

Ex No: 14
Date: 12/10/24

AIM:

Write a code using RAW sockets to implement sniffing

Algorithm

1. Install python and scapy
2. Create a program open text editor and create a file in notepad called packet_sniffing.py and code to capture & analyse the network packets
3. Set up packet tracer by check if the packet has IP layer, identify the packet's protocol such as TCP, UDP, ICMP
4. Capture network packets
5. Run the packet sniffer by using command
6. Generate network traffic by running the program

Program

```
from scapy.all import IP, TCP, UDP, ICMP
def packet_callback(packet):
    if IP in packet:
        ip_layer = packet[IP]
        protocol = ip_layer.proto
        src_ip = ip_layer.src
        dst_ip = ip_layer.dst
        protocol_name = ""
    if protocol == 1:
        protocol_name = "ICMP"
    elif protocol == 6:
        protocol_name = "TCP"
    elif protocol == 17:
        protocol_name = "UDP"
    else
        protocol_name = "unknown protocol"
```

```
print("Protocol: {protocol_name}")
print("Source IP: {src_ip}")
print("Destination IP: {dst_ip}")
print("-", 80)

def main():
    sniff(prn=packet_callback, filter="ip", store=0)
if __name__ == "__main__":
    main()
```

Output

```
> python packet_sniffer.py
Protocol: TCP
Source IP: 192.168.25.227
Destination IP: 142.250.67.78
```

Result:

The program to implement packet sniffing is done successfully

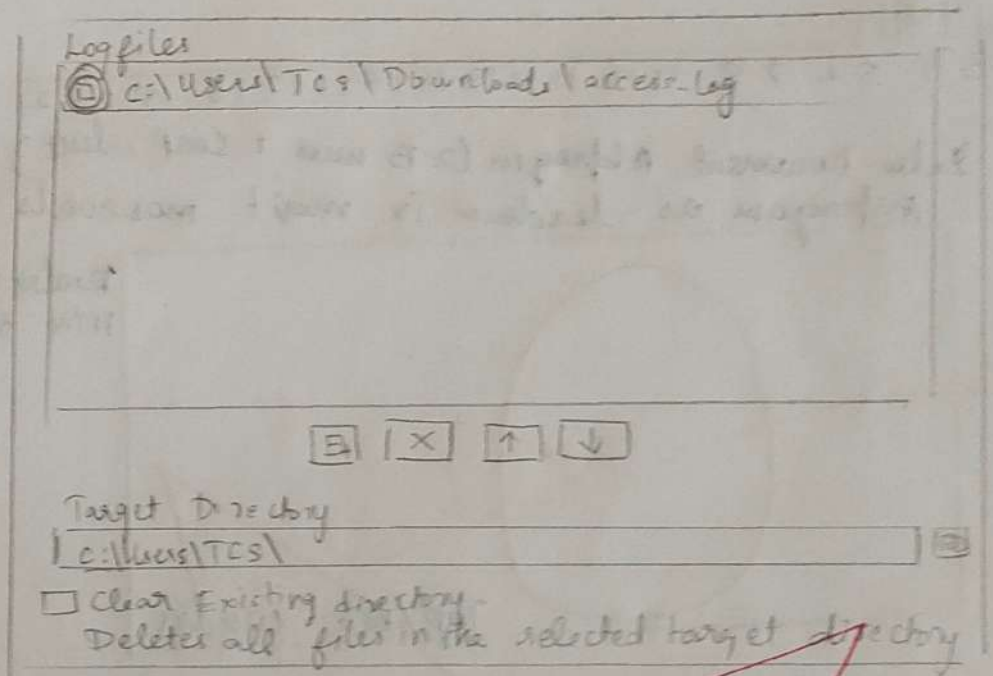
Ex No: 15
Date: 15/10/24.

AIM:

To analyse the different types of weblogs using wilize tool

Algorithm

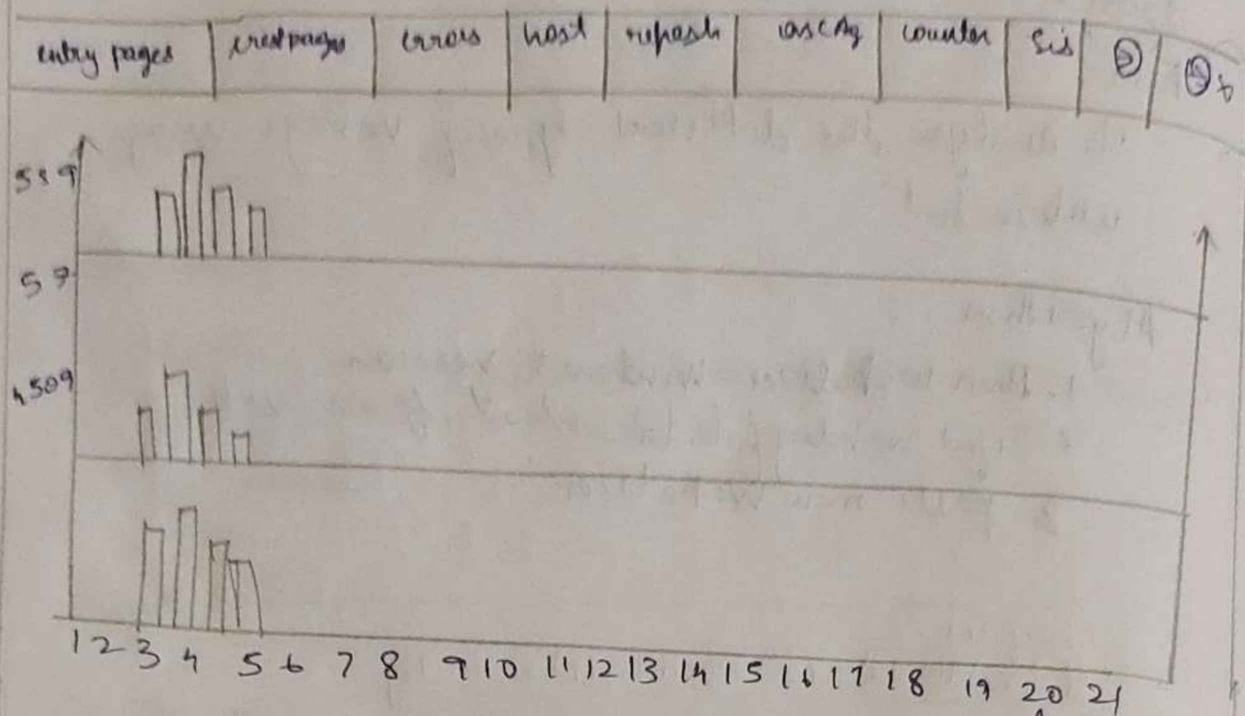
1. Run Webalizer Window's version
2. Input weblog file (download from web)
3. press run webalizer



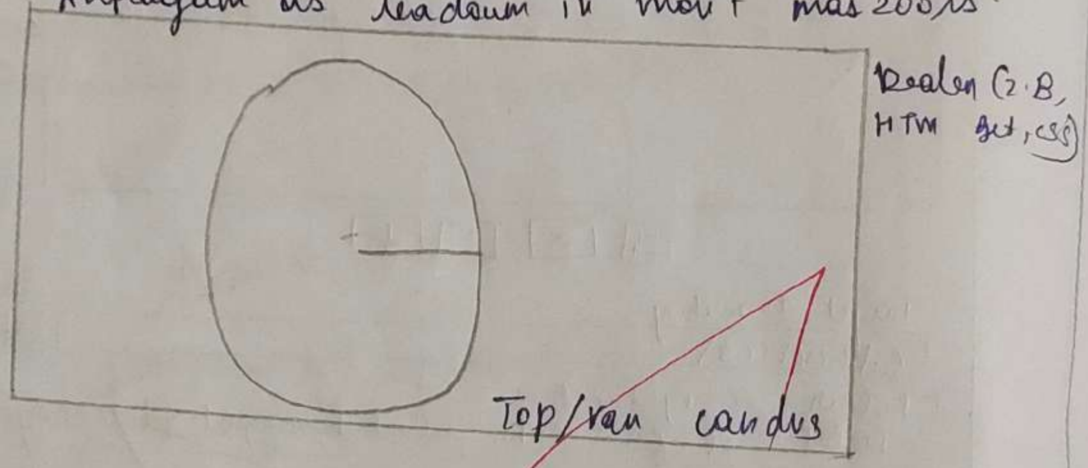
RESULT:

Thus the experiment for using webalizer for weblog analysis executed Verified

output:-



Pulen-Document Abfragen (z-B mum 1 sarp, lent)
Abfragen as leadeum in moit mas 200ls.



Some step webalize