INDEX

Name: R. Deeksha Age:

Class: CSE-III Sec: 'A' Subject:

Address:

Emergency Contact :

Sl No		Particulars	Page No
1.	13.7.24	Study of various Network commands used in Linux and windows	1 9
2.	27.7.24	Study of Network cables	8 10
3.	30.7.24	Experiments on CISCO PACKET TRACER (simulation Tool):	11 10
4	10/8/24	Setup & configure a LAN using switches Ethernet	13
5	13/8/24	Experiment on packet capture tool: Wireshark	16
6	28/8/24	Hamming code	24
7	31/8/24	sliding window protocol simulation	36
8.	3/9/24	Virtual LAN Configuration	48
9.	7/9/24	Implementation of subnetting in CISCO Packet tracer simulator	56
10.	10/9/24	Interneworking with routers in CISCO Packet tracer simulator	64
11.	14/9/24	a) Simulate static routing configuration using Cisco Packet tracer	66
	17/9/24	b) Simulate RIP using cisco Packet tracer	71
12.	21/9/24	a) Echo client server using TCP/UDP	75
	24/10/24	b) chat client server using TCP/UDP	79
13.	28/10/24	Ping program	82
14.	9/11/24	Raw sockets to implement packet sniffing	85
15.	9/11/24	Web logs using Webalizer tool.	89.

Date: 13.7.24 EXERCISE - 01

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

BASIC NETWORKING COMMANDS:

arp -a: Interface : 172.16.75.53 --- 0x19

Internet Address	Physical Address	Type
172.16.72.1	7c-5a-1c-cf-bc-41	dynamic
172.16.73.97	2a-3c-13-0a-cc-17	dynamic

HOSTNAME:

DESKTOP-C01BHTD

ipconfig/all: Windows IP configuration

Hostname . . . . . : DESKTOP-C01BHTD
Primary Dns Suffix . . . . . :
Node Type . . . . . : Hybrid

nbtstat -a: [[-a Remote Name] [-A Ip address] [-c] [-n] [-r] [-R] [-RR] [-S] [-s] [interval]]

netstat: Active Connections

Proto	Local Address	Foreign Address	State
TCP	127.0.0.1:46978	DESKTOP-C01BHTD	ESTABLISHED

nslookup:

Default Server: Unknown
Address: 172.16.72.1

Pathping: Usage: pathping [-g host-list] [-h maximum-hops] [-i address] [-n] [-p period] [-q num-queries] [-w timeout] [-4] [-6] target-name

Route : ROUTE [-f][-p] [-4|-6] command [destination] [MASK netmask] [gateway] [metric metric] [IF interface]

SOME IMPORTANT LINUX NETWORKING COMMANDS:

1) ip: Syntax: ip <options> <object> <command>

a. [root@server ~]# ip address show
1: lo: <LOOPBACK, UP, LOWER-UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 100 link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

b) [root@server ~]# ip address add 192.168.1.254/24 dev enp303
An IP to an interface is assigned

c) [root@server ~]# ip address del 192.168.1.254/24 dev enp303
An IP to an interface is deleted

d) [root@server ~]# ip link set eth0 up
The status of an interface is altered by bringing the interface online

e) [root@server ~]# ip link set eth0 down
The status of an interface is altered by bringing the interface offline.

f) [root@server ~]# ip link set eth0 promisc on
The status of an interface is altered by enabling promiscuous mode.

g) [root@server ~]# ip route add default via 192.168.1.254 dev eth0
A default route is added (for all addresses) via the local gateway 192.168.1.254 that can be reached on the device used.

h. [root@server ~]
192.168.1.254
A route is add
gateway at 192

i) [root@server ~]
A route is ad
can be reach

j) [root@server ~

The route is
via the gat

k) [root@server ~]
10.10.1.4 d
cache

2) if config:
enp30: flags = 4
inet 172.16.8.
172.16.11.255
prefix lin -64
d8:7e txqu
bytes 10836 33

3) mtr: syntax:
a) [root@server
Host
172.16.8.1
static.41.229.249

h. [root@server ~]# ip route add 192.16.1.0/24 via 192.168.1.254

A route is added to 192.168.1.0/24 via the gateway at 192.168.1.254

i) [root@server ~]# ip route add 192.168.1.0/24 dev eth0

A route is added to 192.168.1.0/24 that can be reached on the device used.

j) [root@server ~]# ip route delete 192.168.1.0/24 via 192.168.1.254

The route is deleted for the route 192.168.1.0/24 via the gateway at 192.168.1.254.

k) [root@server ~]# ip route get 10.10.1.4

10.10.1.4 dev enp2s0 src 192.168.1.254 uid 0

cache

2) if config:

enp2s0: flags=4163<UP, BROADCAST, RUNNING, MULTICAST> mtu 1500 inet 172.16.8.100 netmask 255.255.252.0 broadcast 172.16.11.255 inet6 fe80::7661:bfa8:24c2:c76c prefixlen 64 scopeid 0x20<link> ether 50:9a:4c:34 d8:7e txqueuelen 1000 (Ethernet) RX packets 120153 bytes 108363389 (103.3 MiB) RX errors 0 dropped 24

3.) mtr: syntax: mtr < options> hostname / IP

a) [root@server ~]# mtr google.com

Host	Packets Loss%	Snt	Last	Avg	Pings Best	Wrst	StDev
172.16.8.1	0.0%	78	0.3	0.3	0.2	0.5	.03
static.41.229.249.49-tataidc.co.in	0.0%	78	30	3.9	2.3	21.7	3.3

-9 google.com

c) [root@servers ~]# mtr -b google.com

Host	Packets Loss%	Snt	Last	Pings Avg	Best	Wrst
172.16.8.1	0.0%	64	0.7	0.4	0.1	13.6

static-41.229.249.49-tataidc.co.in (49.249 20.4 229.41)

d) [root@servers ~]# mtr -c 10 google.com

Host	Packets Loss%	Snt	Last	Pings Avg	Best	Wrst
172.16.8.1	0.0%	10	0.4	0.4	0.3	0.4
static-41.229.249.49	0.0%	10	29.6	27.4	3.5	326.3
	0.0%	10	5.3	3.2	2.9	3.5

3

A) tcpdump:

a) [root@servers ~]# dnf install -y tcpdump

package tcpdump-14:4.9.0-2.fc26.i686 is already installed, skipping. Dependencies resolved

Nothing to do. complete!

b) [root@servers ~]# tcpdump -D

1. enp50 [UP, Running]
2. any [Pseudo-device that captures on all interfaces] [UP, Running]
3. lo [UP, Running, Loopback]

c. [root@servers ~]# tcpdump -i eth0

09:13:07:34 28 92 ARP Request who-has 142.250.19646 tell localhost.localdomain, length 46

09:13:07.393514 IP localhost.localdomain.mdns > 224.0.0.251.mdns: 0 [1n] ANY (QM)? 254.1.168.192.in-addr.arpa (73)

d) . [root@servers ~J# tcpdump -i etho -c10
89:13:40.541331 IP 172.16.8.168.55264 >339.255.25
250.ssdp: UDP, length 176
09:13:40.548780 ARP, Request who has 172.16.8.1
tell 172.16.9.144, length 46.

A. tcpdump
i) [root@servers ~J# dnf install -y tcpdump
Last metadata expiration check: 0:09:21 ago on Sat
27 Jul 2024 12:06:57 PM IST.
Package tcpdump-14.4.9.0-2.fc26.i686 is
already installed, skipping.
Dependencies resolved.
Nothing to do.
Complete!

ii) [root@servers ~J# tcp dump -D
1. enp2s0 [up, running]
2. any (Pseudo-device that captures on all interfaces)
[up, running]
3. lo [up, running, Loopback]

iii) [root@servers ~J# tcpdump -i lo
tcpdump: verbose output suppressed, use -v or -vv for
full protocol decode listening on lo, link-type
EN10MB (Ethernet), capture size 262144 bytes
^C

iv) [root@servers ~J# tcpdump -i lo -c 10
tcpdump: verbose output suppressed, use -v or -vv for
full protocol decode listening on lo, link-type
EN10MB (Ethernet), capture size 262144 bytes
^C

v) [root@servers ~J# tcpdump -i lo -c 10 host 8.8.8.8
tcpdump: verbose output suppressed, use -v or -vv for
full protocol decode listening on lo, link-type
EN10MB (Ethernet), capture size 262144 bytes
^C

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

[root@servers ~]# tcpdump -i lo dst host 8.8.8.8
tcpdump: verbose output suppressed, use -v or -vv
for full protocol decode listening on lo,
link-type EN10MB (Ethernet), capture size
262144 bytes
^C

[root@servers ~]# tcp dump -i lo net 10.1.0.0 mask
255.255.255.0
tcpdump: verbose output suppressed, use -v or -vv
for full protocol decode listening on lo, link-type
EN10MB (Ethernet), capture size 262144 bytes
^C

[root@servers ~]# tcpdump -i lo port 53
tcpdump: verbose output suppressed, use -v or -vv
for full protocol decode listening on lo, link-type
EN10MB (Ethernet), capture size 262144 bytes
^C

[root@servers ~]# tcpdum -i lo host 8.8.8.8 and post 53
tcpdump: verbose output suppressed, use -v or -vv
for full protocol decode listening on lo, link-type
EN10MB (Ethernet), capture size 262144 bytes
^C

[root@servers ~]# tcpdump -i lo -c 10 host www.google.com
tcpdump: verbose output suppressed, use -v or -vv and post 443
for full protocol decode listening on lo, link-type
EN10MB (Ethernet), capture size 262144 bytes
^C

[root@server ~]# tcpdump -i lo port not 53 and
tcpdump: verbose output suppressed, use -v or -vv
for full protocol decode listening on lo, link-type
EN10MB(Ethernet), capture size 262144 bytes

^C

5. Ping
[root@server ~]# ping google.com
PING google.com (142.250.182.14) 56(84) bytes of data
64 bytes from maa05s18-in-f14.1e100.net (142.250.182
.14): icmp-seq=1 ttl=120 time=3.63 ms

^C

6. [root@server ~]# ping -c 10 google.com
PING google.com (142.250.182.14) 56(84) bytes of data.
64 bytes from maa05s18-in-f14.1e100.net (142.250
.182.14): icmp-seq=1 ttl=120 time=3.15ms

^C
configuring an Ethernet connection by using nmcli
1. # nmcli connection show

NAME	UUID	TYPE	DEVICE
New 802-3-ethernet connection	7fefdf34-0c83-457c-94e4	802-3-ethernet	enp2s0
802-ethernet	93d082fe-260-4144-a7e7-76254606	802-3-ethernet	—

2. [student@localhost ~]$ nmcli connection add con-name deek ifname enp2s
type 802-3-ethernet
connection 'deek' (6194515e-3990-4486-9a5a-eff41ba474eb) successfully
added.

3) [student@localhost ~]$ nmcli connection show

NAME	UUID	TYPE	DEVICE
New 802-3-ethernet connection	7fefdf34-0c83-457c-94e4	802-3-ethernet	enp2s0
802-ethernet	93d082fe-260-4144-a7e7-76254606	802-3-ethernet	—
deek	6194515e-3990-4486-9a5a-eff41	802-3-ethernet	—

5/8

Result:
Thus the study of various network commands used in Linux and windows.

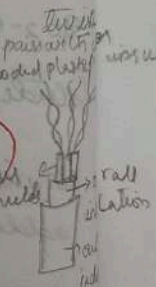
PRACTICAL-2

AIM: Study of different types of Network cables

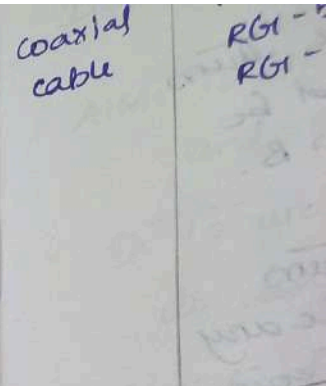
a) Understand different types of network cable

Different type of cables used in networking are

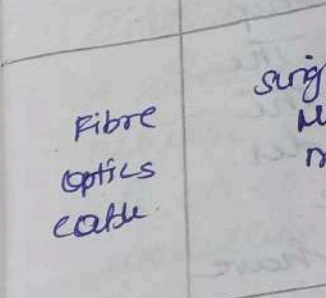
1. Unshielded Twisted Pair (UTP) Cable
2. Shielded Twisted Pair (STP) Cable
3. Coaxial Cable
4. Fibre Optic Cable

Cable Type	Category	Maximum Data Transmission	Advantages/ Disadvantages	Application/ Use	Image
UTP	Category 3	10 bps	Advantages • Cheapest in cost • Easy to install as they have a smaller overall diameter Disadvantages • More prone to (EMI) Electromagnetic interference and noise	10Base-T Ethernet	
	Category 5	Up to 100 Mbps		Fast Ethernet, Gigabit Ethernet	
	Category 5e	1 Gbps		Fast Ethernet, Gigabit Ethernet	
STP	Category 6, 6a	10 Gbps	Advantages • Shielded. • Faster than UTP. • Less susceptible to noise and interference Disadvantages	Gigabit Ethernet, 10G Ethernet (SSTP) Widely used in data centers	
SSTP	Category 7	10 Gbps	• Expensive • Greater installation effort	Gigabit Ethernet, 10 G Ethernet (100m)	

Coaxial cable



Fibre optics cable



b) Make yo

Tools

• Ethernet

but CAT

• A crimp

shaped

strip

• Two R

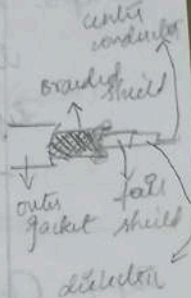
• Option

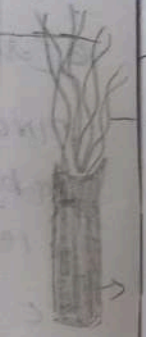
Take a

hardy

Step 1: -

Step 2: N

Coaxial cable	RG-6 RG-59 RG-11	10-100 Mbps	• High bandwidth • Immune to interference • Low loss bandwidth • versatile • Disadvantages • Limited distance • cost • Size is bulky	speed of signal is 500m Television network high speed industrial connections	
Fibre optics cable	single mode Multi mode	100 Gbps	Advantages • High speed • High bandwidth • High security • Long distance Disadvantages • Expensive • Requires skilled installers	• Maximum distance of fibre optics cable is around 100 meters	



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 crimping tool. This is an all in-one networking tool shaped to push down the pins in the plug and strip and cut the shielding off the cables.
- Two RJ45 plugs.
- Optional two plug shields.

Take a print out the diagram below or have it handy as a reference.

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

Step 2: Next, strip approximately 1.5cm of cable shielding from both ends. The crimping tool has a round area to complete this task.

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, snip them back to create an even end. The difficult aspect is placing these into the RJ45 plug without messing up the order. To do so, hold the plug with the clip side facing away from you and have the gold pins facing towards you, as shown.

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 isn't, that means that you stripped off too much shielding. Simply strip the cables back a little more.

Step 6: After the wires are securely sitting inside the plug, insert it into the 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 a straight through cable)

Result:
Thus the study of Network cables is studied

6/8

Date: 30.7.24 Practical- 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 Tracer 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 troubleshooting 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.netacad.com and log in with your Cisco Networking Academy credentials; then click on the Packet Tracer graphic and download the package appropriate for your operating system. (can be used to download on your laptop).

Windows

Installation in Windows is pretty simple and straightforward; the setup comes in a single file named PacketTracer_Setup_6.0.1.exe. Open this file to begin the setup wizard.

Linux users with an Ubuntu / Debian distribution should download the file for Ubuntu and those using Fedora/Redhat/CentOs must download the file for Fedora. Grant executable permission to this file by using chmod, and execute it to begin the installation

chmod +x PacketTracer601_i386-linstaller-rpm.bin

./PacketTracer601_i386-linstaller-rpm.bin

USER INTERFACE OVERVIEW:

The layout of Packet Tracer is divided into several components. The components of the Packet Tracer interface are as follows: match the numbering with explanations.

1. Menubar - This is a common menu found in all software applications. It is used to open, save, print and so on.

2. Main toolbar - This bar provides shortcut icons to menu options that are commonly accessed such as open, save, zoom, undo and redo and on the right-hand side is an icon for entering network information for the current network.

3 - Logical/Physical workspace tabs - These tabs allow you to toggle between the logical and physical work areas.

4. Workspace - This is the area where topologies are created and simulations are displayed

5. Common tools bar - This toolbar provides controls for manipulating topologies, such as select, move layout, place note, delete, inspect, resize shape and add simple/complex PDU.

6. Real-time/Simulation tabs - These tabs are used to toggle between the real and simulation modes.

4 Network compon
all of the net
Packet Tracer
two areas:
This area
device specif
category is
displays

5 User-created
customised pa
from this
displayed

d) Analyse th
CISCO PAC

1. From the
drag and d
a) 4 G
b) 4

2) Click on co
a) Click on
b) Select
Using th
in gr

c) Simila
copp

3) Click on
Desktop ta
an IP ad
default
is not
devices

c) 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

f) User-created packet box - Users can create highly customised 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 glow in green, indicating that the link is up.
   c) Similarly connect 4 PCs to the switch using copper straight-through cable

3) Click on the PCs connected to hub, go to the 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.

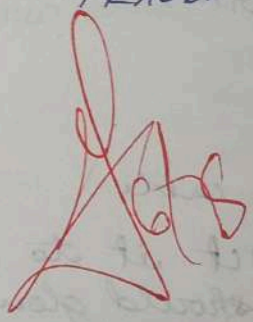
a) Drag and drop it on one of PC and then drop it on another PC (destination machine) connect to the HUB.

4. Observe the flow of PDU from source PC to destination PC by selecting the Realtime mode of simulation.

5. Repeat step #3 to step #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 behaviours of switch and Hub.

Result:
Thus the expesiments on CISCO PACKET TRACER are executed.



PRACTICAL
Student
1. Which
reachab
device
comm
a hos

2. which
hops it
destin
comm
by a

3. Which
of yo

4. Whic
s

5.

PRACTICAL-1

Student observation:

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

command to find the reachability of a host machine from your device:

- ping <hostname or IP address>

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

Command to get the details of hops taken by a paket to reach its destination:

- traceroute <hostname or IP address> (Linux/Unix)

3. Which commands display the ip configuration of your machine?

- ifconfig (Linux/Unix, older versions)
- ip addr (Linux/Unix, newer versions)
- ipconfig (Windows)

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

- netstat -tuln
- ss -tuln

5. Write the modify the ip configuration in a linux machine.

Temposary changes:

Using config (old method)
Using ip (Modern method)

Sudo ip addr add 192.168.1.10/24 dev eth0
Sudo ip route add default via 192.168.1.1

PRACTICAL 2

Student observation:

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

Straight cable:
- All the wires are in the same order on both ends of the cable.
- Used to connect different types of devices (eg: PC to switch, router to switch)

Cross cable:
- The transmit and receive wires are swapped on each end
- Used to connect similar device directly (eg: PC to PC, switch to switch)

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

3. Which type cable is used to connect two PC? (straight / cross cable)
- cross cable (crossover cable)
- used for direct PC to PC connections without a hub, switch or router.

4. Find out cable your network card cat 6 (ca for short

5 Write dow fault and o a twisted
- straight
  → used
devices.
→ Both en
- cross cabl
  → used
devices di
→ ~~Used~~
and th
standar

Challenges
- crimpin
- wire
- Testin

Output R
- Succes

4. Find out the category of twisted pair cable your lab to connect the PC to the network socket

cat 6 (category 6): supports up to 10Gbps for short distances.

5 Write down your understanding, challenges faced and output received while making a twisted pair cross/straight cable.

- straight cable:
  → used to connect different types of network devices.
  → Both ends have the same wiring orders
- cross cable
  → used to connect similar types of network devices directly.
  → ~~Crossed cable~~ One end follows FIA/EIA-56P-B and the other follows TIA/EIA-56P-A standard.

Challenges faced
- Crimping Issues
- Wire orders
- Testing

Output Received:
- Successfully Made cable

Student Observation:

a) From your observation write down the behaviours of switch and HUB in terms of forwarding the packets received by them.

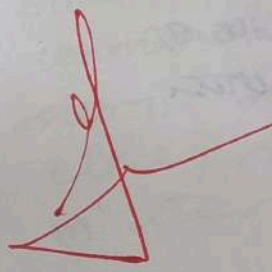
HUB

- Broadcasting:
  → A hub is a basic networking device that operates at the physical layer (Layer 1) of the OSI Model
  → When a hub receives a packet on one of its ports it broadcasts the packet to all other ports, regardless of the destination address.
  → This behaviour results in all devices connected to the HUB receiving the packets even if they are not the intended recipient.
- Collision Domain:
  → All devices connected to HUB are in same collision domains.

SWITCH:

- Intelligent Forwarding:
  → A switch operates at the data link layer (Layer 2) of the OSI model.
  → When a switch receives a packet it examines the MAC address of the destination device.
  → It forwards the packet only to the port associated with the destination MAC address, thus reducing unnecessary traffic and improving network efficiency
- Collision Domain:
  → Each port on a switch represents a seperate collision domain.

- This is
  of collisio
  are only
b) Find ou
your colle
topology
- Star
  → All
  switc
  → This
  widely
  due t



[PC1]

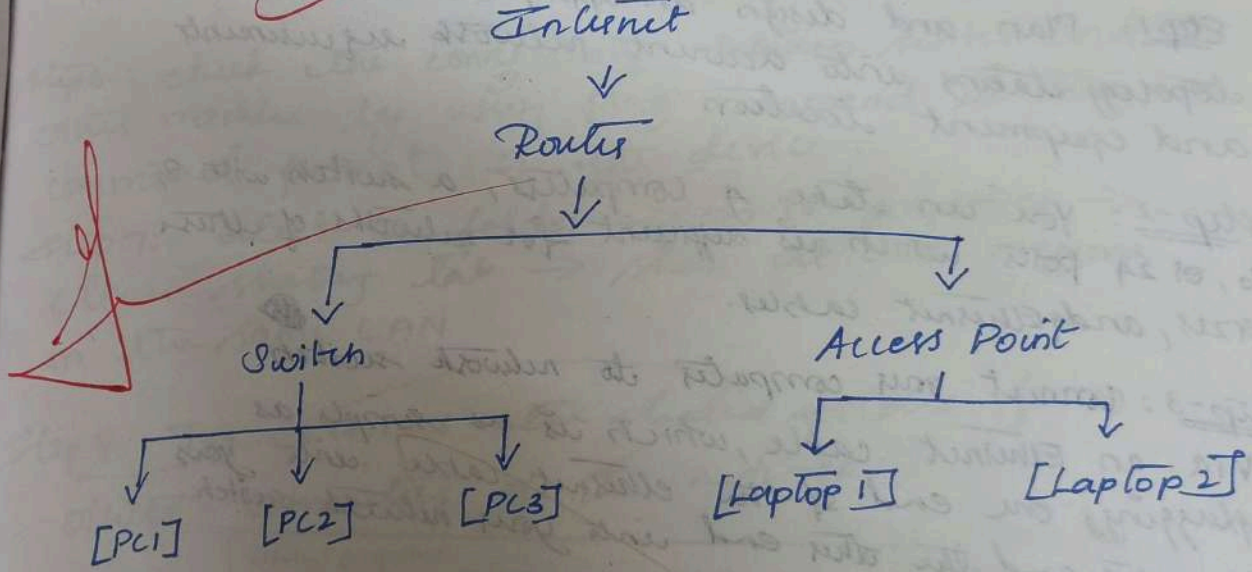
• This isolation significantly reduces the likelihood of collision compared to a hub, as packets are only forwarded to the intended recipients

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

• 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.



Practical-4

AIM:
Setup and configure a LAN (Local Area Network) using a switch and Ethernet cables in your lab

What is LAN?
A local Area Network (LAN) refers to a network that connects devices within a limited area, such as an office building, school or home. It enables users to share resources, including data, printers, and internet access. A LAN switch serves as the primary connecting devices, managing and directing communication within the local network. Each connected device on a LAN switch can communicate directly with each other, allowing for fast and secure data transfer.

How to setup a LAN:
Step 1: Plan and design an appropriate network topology taking into account network requirements and equipment location

Step-2: You can take 4 computers, a switch with 8, 16, or 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 PCs
1. Log on to the client computer as administrator or as owner.
2. Click Network and Internet connections
3. Right click Local Area connection / Ethernet → Go to properties → select Internet protocol (TCP/IP V4) → click on properties → select use the following IP address option and assign ip address

Similarly assign IP address to all the PCs connected to switch

Step-5: C
1. Connect the switch your comp cable.
2. Log in browser in the login pag the user
3. Configur you wi for eth
4. Assign mask

Step 6: che other ma comman

Step 7: click on the

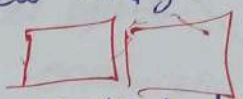
Step 8: T other

Step-5: configure a network switch:

1. 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.

2. Log in to the web interface: Open a web browser and 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 and password to login

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

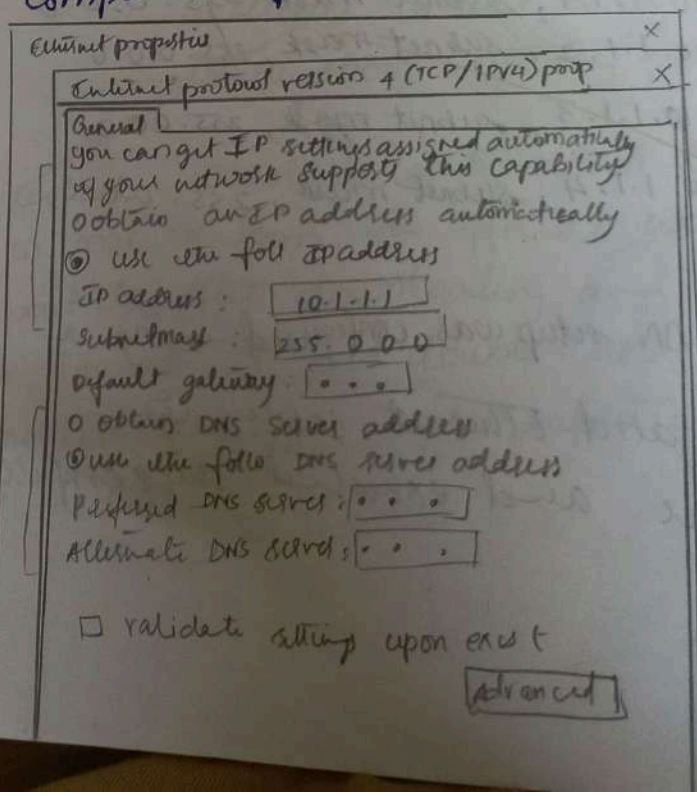


4. Assign IP address as: 10.1.1.5, subnet mask 255.0.0.0

Step6: check the connectivity between switch and other machine by using ping command in the command prompt of the device.

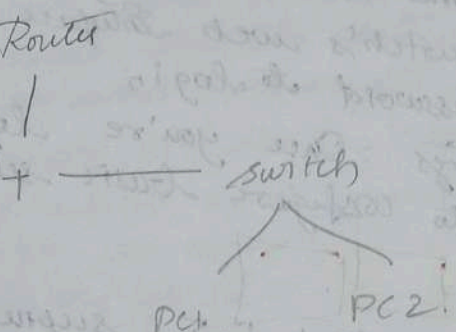
Step7: Select a folder → go to properties → click sharing tab → share it with everyone on the same LAN

Step8: Try to access the shared folder from other computers of the network.



Student observation:

Draw a neat diagram of the LAN in the 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.



IP configuration:

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

Result: The LAN setup was configured using a switch and Ethernet cable is done and executed successfully.

Exercise 5

Date:

AIM:

Experiments

Packet sni

sppets.

-> stores an

fields

-> Never s

- No packe

- Receives

Packet sniff

↳ Tcpdu

↳ Wire sh

Wireshark:

wiresha

as ethernet

display eth

-> what w

-> Used for

Getting wi

windows

Capturing

wireshark

of the

capturing

Exercise 5

Date:

-AIM:

Experiments on packet Capture tool: Wireshark

Packet sniffers: Captures messages sent/received from your sniffer.

- Stores and display the contents of the various protocol fields in message.
- Never sends packets itself.
- No packets addressed to it
- Receives a copy of all packets.

Packet sniffer structure Diagnostic tool

↳ Tcpdump (eg. tcpdump - ehost - 10.129.41.2-W

↳ Wireshark (wireshark - r exeout)

Wireshark:

Wireshark, a network analysis tool formerly known as ethereal, captures packets in real time and display them in human-readable format.

→ What we can do - Capture network traffic

- Decode packets protocols using dissectories

- Analyse problems.

→ Used for people - learn network protocol internals inspect network administrators troubleshoot network problems.

Getting wireshark: Wireshark can be down load for windows or mac os from the official website.

Capturing packets: After downloading and installing wireshark launch it and double check the name of the network interface under capture to start capturing packets.

The "packet list" pane : display all packets in current capture.

The "packet details" pane : Shows the current packet in a more detailed form.

The "packet bytes" pane : shows data of current packet.

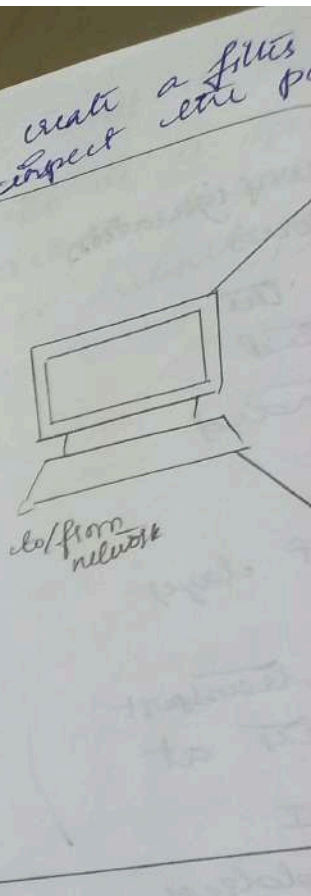
colouring, sample captures, filtering packet, Inspecting packets, flow graph: gives a better understanding of what we see.

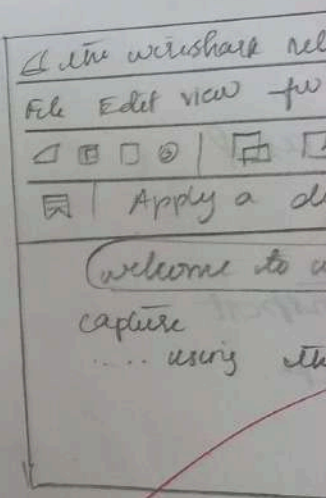
capturing and analysing packets using wireshark tool.

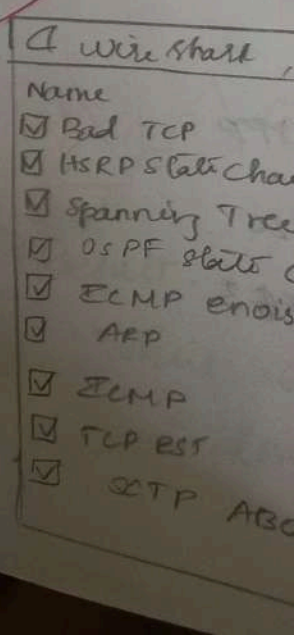
Procedure:

select local area connection in wireshark
go to capture → options
select stop capture automatically after 100 packets
then click start capture
save the packets.

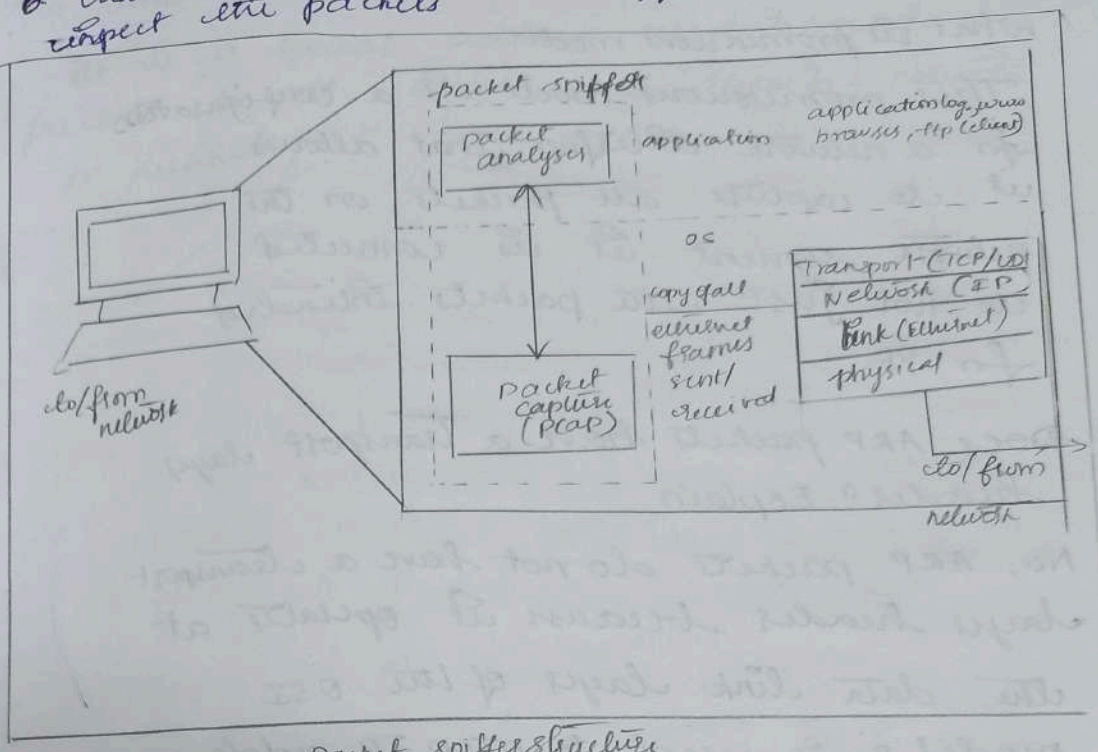
1. create a filter to display only TCP/UDP packets, inspect the packets and provide the flow graph
2. create a filter to display only ARP packets and inspect the packets.
3. create a filter to display only DNS packets and provide the flow graph.
4. create a filter to display only HTTP packets and inspect the packets
5. create a filter to display only IP/ICMP packets and inspect the packets.



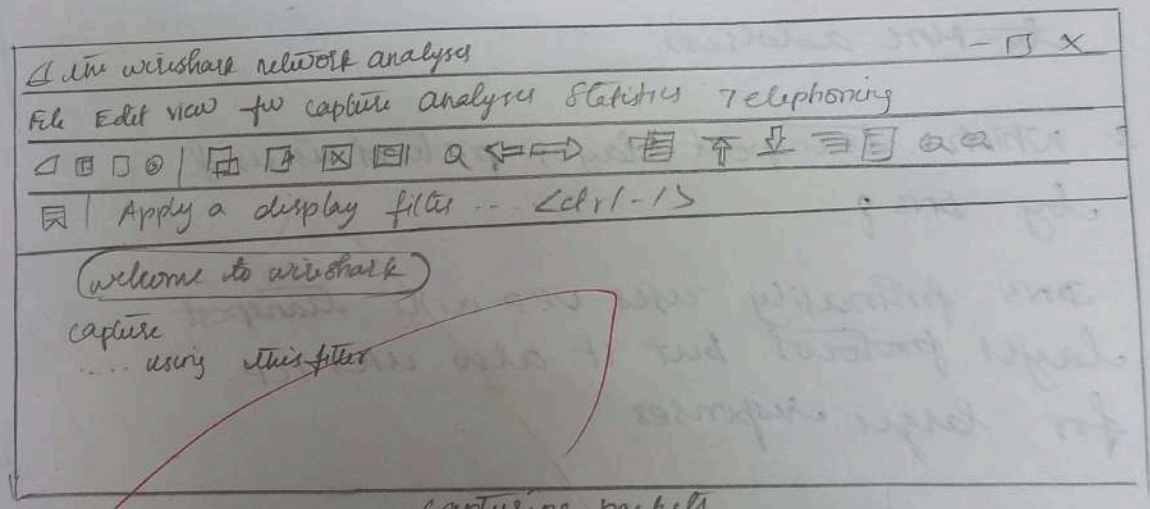




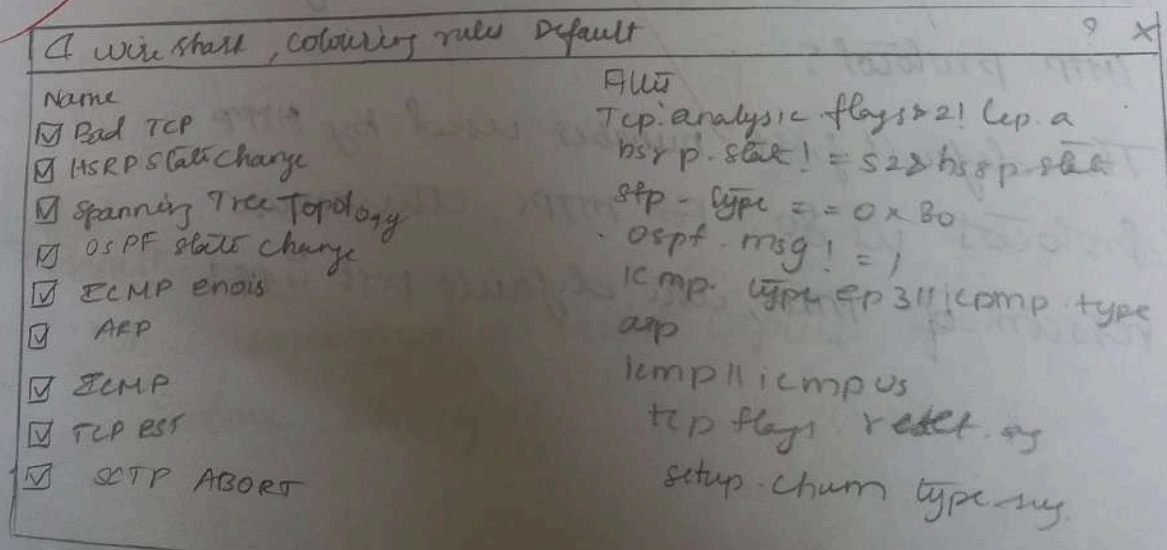
6 create a filter display only DHCP and inspect the packets



Packet sniffer structure



Capturing packets



Student Observation

1. What is promiscuous mode?

This promiscuous mode is a configuration for a network interface that allows it to capture all packets on the network segment it is connected to not just the packets intended for it.

2. Does ARP packets have a transport layer headers? Explain

No, ARP packets do not have a transport layer header because it operates at the data link layer of the OSI model & is used to map IP address to MAC address.

3. Which transport layer protocol is used by DNS?

DNS primarily uses UDP as its transport layer protocol but it also uses TCP for large responses

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

The default port numbers used by HTTP protocol is 80. For HTTPS, the secure version of HTTP, the default port is 443

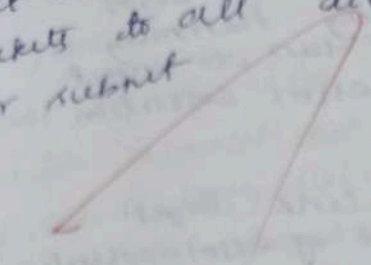
5. What is
It is a
packets to
or subnet

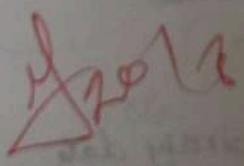
Result:

Thus,
is do

5. What is a Broadcast IP address?

It is a special address used to send packets to all devices on a specific network or subnet





Result:

Thus, experiment on Packet Capture Tool is done successfully.

Practical-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. It is a technique developed by R.W. Hamming

Create a sender program with below features:

1. Input to sender file should be a text of any length. Program should convert the text to binary.

2. Apply hamming code concept on the binary data and add redundant bits to it.

3. Save this file called channel.

Create a receiver program with below features:

1. Receiver program 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.

convert the binary data to ascii and display the output.

Program:

```
import numpy as np
# function to convert text to binary
def text_to_binary(text):
    return ''.join(format(ord(char), '08b') for char in text)
# function 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])
# function to calculate redundant bits needed for error
correction
    def calc_redundant_bits(m):
        r=0
        while(2**r<m+r+1):
            r+=1
        return r
# function to insert redundant bits into the data
    def pos_redundant_bits(data, r):
        j=0
        k=0
        m=len(data)
        res=''
# adding redundant bit at positions 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
# function to calculate parity bits
    def calc_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)

# function to detect and correct errors
def detect_and_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(f"Error detected at position: {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 position: {res}")
        else:
            print("Error position out of range. No
                  correction performed.")
        corrected_data = "".join(data)
        return corrected_data
    else:
        print("No error detected.")
        return data

# function to remove redundant bits
def remove_redundant_bits(data, r):
    j = 0
```

```
original_data = ''
for i in range(1, len(data)+):
    if i == 2**j:
        j += 1
    else:
        original_data += data[i-1]

return original_data

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

# sender program
def sender(text):
    binary_data = text_to_binary(text)
    m = len(binary_data)
    r = calc_redundant_bits(m)
    ans = pos_redundant_bits(binary_data, r)
    ans = calc_parity_bits(ans, r)
    print(f"sender output (binary with redundant bits): {ans}")

    return ans

# receiver program
def receiver(data):
    r = calc_redundant_bits(len(data))
    corrected_data =
    detect_and_correct(data, r)
# remove redundant bits
    original_data = remove_redundant_bits(corrected_
                                          data, r)
ascii_output = binary_to_text(original_data)
print(f"Decoded text: {ascii_output}")
```

```
#Main program
if __name__ == "__main__":
    input_list = input("Enter the list to be encoded: ")
# Sender side
    channel_data = sender(input_list)
# Simulate introducing an error in the channel data (for testing)
# for example introduce an error at position 2
    corrupted_data = introduce_error(channel_data, 2)
# receiver reads and corrects the data
    receiver(corrupted_data)
```

Input:

Enter the list to be encoded : dieksha

Output

Input list in binary : 011001000110010101100101011010110011010100001110000

Redundant bits added : position 1:0; position 2:0 ; position 4:0; position 8:0; position 16:0; position 32:0

Total redundant bits added : 6

Sender output (binary with redundant bits): 000110101000111001010110010101110101110011010100001110000

Introduced error at position : 2

Binary data with error introduced:

0101101010001110010101100101011101011100110101000011100001

Error detected at position : 2

Error corrected at position : 2

corrected binary data: 000111010101010001110010101011001

1010111010110011010110100001100001

Decoded text : dieksha.

Result:

Thus the program to implement error detection and correction using hamming code concept is executed successfully and verified.

Aim: Write a p
data link cla
Simulate the
another.

program should
can make it
receives it s

create a se
1. Input wi
2. Input a t
3. Considers 1
4. Create a
DAT
5. Send the
6. Wait for
7. Reades a
8. Check AC
9. If the ac
send new

Create a rec
1. Reades a
2. check the
3. If the fra
Ack no. th
Else wit

Program:

import
import
class f
def

practical - 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.

program should achieve at least below requirement. You can make it a bidirectional program where receiver is sending its data frames with acknowledgment

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
6. Wait for the acknowledgment from the receiver
7. Reads a file called Receiver_Buffer.
8. Check Ack field for the acknowledgment numbers.
9. If the acknowledgment number is as expected send new set of frames accordingly.

Create a receiver file with following features

1. Reads a file called sender_Buffer
2. check the frame no.
3. If the frame no. are as expected, write appropriate Ack no. in the receiver_Buffer file.
   Else write NACK no. in the receiver_Buffer file

Program:

```
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("\n---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
        acknowledgements...\n")

def receive_frames(frames, window_size):
    print("\n---Receiving frames---")
    for i in range(window_size):
        if i < len(frames) and not frames[i]
        .acknowledged:
            #simulate error with 20% probability
            if random.random() < 0.2:
                print(f"Received Frame {frames[i].
                    frame_no}: {frames[i].data}[ERROR]")

#send NACK
            frames[i].acknowledged=False
            else:
                print(f"Received Frame {frames.frame_no}
                    : {frames[i].data}[OK]")

#send ACK
            frames[i].acknowledged=True

def sliding_window_protocol():
    window_size=int(input("Enter window
        size:"))
    message=input("Enter a message to send:")
#initialize frames
    frames=[frame(i, message[i]) for i in
    range(len(message))]
    base=0 #start of the window
    while base < len(frames):
```

```
#send frame
send_fram
time.slee

#receive a
the curren
receive_f

#slide the
unacknowle
while base
acknowle
base +=
if base <
print(
time

print

if-
```

Output:

```
Enter win
Enter a m
--- sendi
Sent Fram
Sent Fram
Sent Fra
Sent Fram
Sent Fram
.
Frames
```

```
# send f
send_frames (frames[base:], windowsize)
time.sleep(2) # simulate delay

# receive acknowledgements for the frames in
the current window
receive_frames (frames[base:], window_size)
# slide the window: Move base to the first
unacknowledged frame
while base < len(frames) and frames[base].
acknowledged:
    base += 1
if base < len(frames):
    print("\n Resending unacknowledged frames...\n")
    time.sleep(2)
print("\n All frames sent and acknowledged!
")
if __name__ == "__main__"
    sliding_window_protocol()
```

Output:

```
Enter window size: 20
Enter a message to send: deeksharavindran
--- sending Frames ---
Sent Frame 0: d
Sent Frame 1: e
Sent Frame 2: e
Sent Frame 3: k
Sent Frame 4: s
...
Frames sent, waiting for acknowledgments...
```

Received Frame 0 : d [Received]
Received Frame 1 : e [Received]
Received Frame 2 : e [ERROR]
Received Frame 3 : k [Received]
Received Frame 4 : s [Received]

Resending unacknowledged Frames...

--- Sending Frames ---
Sent Frame 2 : e
Sent Frame 5 : h
Sent Frame 6 : a
Frames sent, waiting for acknowledgments...

--- Receiving Frames ---
Received Frame 2 : e [Received]
Received Frame 5 : h [Received]
Received Frame 6 : a [Received]

Resending unacknowledged frames...
--- Sending Frames ---
Sent Frame 7 : r
Sent Frame 8 : a
Sent Frame 9 : v
Sent Frame 10 : i
Sent Frame 11 : n
Frames sent, waiting for acknowledgments...

--- Receiving Frames ---
Received Frame 7 : r [Received]
Received Frame 8 : a [Received]
Received Frame 9 : v [Received]
Received Frame 10 : i [Received]
Received Frame 11 : n [ERROR]

Resending unacknowledged frames...

---Sending Frames---
Sent Frame 11: n
Sent Frame 12: d
Sent Frame 13: r
Sent Frame 14: a
Sent Frame 15: n

Frames sent, waiting for acknowledgments...

---Receiving Frames---
Received Frame 11: n [Received]
Received Frame 12: d [Received]
Received Frame 13: r [Received]
Received Frame 14: a [Received]
Received Frame 15: n [Received]

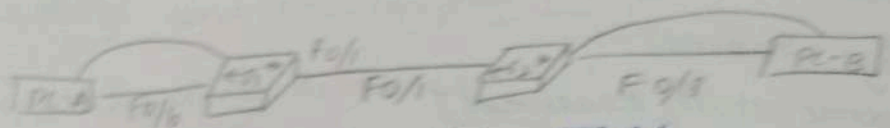
All frames sent and acknowledged!

Result:
Thus the program to implement sliding window protocol is executed successfully.

Aim: a) Simulate virtual LAN Configuration using cisco packet Tracer Simulation

Packet Tracer - Configure VLAN's and Trunking

Physical Mode Topology



Addressing Table

Device	Interface	Ip address	Subnet mask	Default Gateway
S1	VLAN 1	192.168.1.11	255.255.255.0	N/A
S2	VLAN 1	192.168.1.12	255.255.255.0	N/A
PC-A	NIC	192.168.10.3	255.255.255.0	192.168.10.1
PC-B	NIC	192.168.10.4	255.255.255.0	192.168.10.1

Objectives:

Part 1: Build the network and configure basic device settings

Part 2: Create VLANs and assign switch ports

Part 3: Maintain VLAN port assignments and the VLAN database

part 4: Configure an 802.1q trunk between the switches

Background/Scen

Modern switche
performance by
-last domains
efficiency an
communicatio
VLANs to
preserving
status physi
create VLANs
verify func
between
among hos

Instructions

Part 1:

Step 1: Build
topol

Attach
diagram,

a. Click
to the

b. Click
the Tab
them on

Background / Scenario:

Modern switches use VLAN's to enhance network performance by dividing large layer 2 broad-cast domains into smaller segments, improving efficiency and security by controlling host communication. VLAN trunks allow multiple VLANs to share a single link while preserving their identities. In a packet trace physical Mode (PTPM) activity, users will create VLANs; assign them to access ports, verify functionality and establish a trunk between switches to enable communication among hosts in the same VLAN's

Instructions

Part 1:

Step 1: Build the network as shown in the topology

Attach the devices as shown in the topology diagram, and cable as necessary

a. Click and drag both switch S1 and S2 to the Rack

b. Click and drag PC-A and PC-B to the the Table and use the power button to turn them on.

c. Provide network connectivity by connecting copper straight-through cable, as shown in the topology.

d) Connect console cable from the device PC-A to S1 & from PC-B to S2

Step 2: Configure basic settings for each switch.

a. From the desktop tab on each PC, use the terminal to console into each switch & and enable privileged EXEC mode.

b. Enter configuration mode.

c) Assign a device name to switch

d) Assign class as the privileged EXEC encrypted Password.

e. Assign cisco as the console password and enable login.

f) Assign cisco as the vty password and enable login.

g) Encrypt the plaintext passwords.

h) Create a banner that warns anyone accessing the device that unauthorized access is prohibited

g.) Configure addressing

j) shutdown

k) Set the

l) Save the configuration

Step 3: Configure From the IP configure information table.

Step 4: Test

Test network to ping devices.

Questions:

1) Can PC-

A: Yes, by to switches IP add ping PC

2. Can PC Yes, P

g.) Configure the IP addressing table for VLAN 1 on the switch.

g) shutdown all interfaces that will not be used.

k) Set the clock on each switch

l) Save the running Configuration to the startup configuration file.

Step3: Configure PC hosts.

From the desktop tab on each PC, click IP configuration and enter the addressing information as displayed in the addressing table.

Step4: Test connectivity.

Test network connectivity by attempting to ping between each of the labled devices.

Questions:

1) Can PC-A ping PC-B?

A: Yes, if both PCs are correctly connected to their respective switches and the switches have properly configured VLAN with IP addresses, PC-A should be able to ping PC-B.

2. Can PC-A ping S1?

Yes, PC-A should be able to ping switch S1 as both are within the same

Can PC-B ping S2?

A: Yes, PC-B should be able to ping switch S2 as both are within the same subnet.

4.) Can S1 ping S2?

A: Yes, S1 will be able to ping S2 since they are directly connected through same network segment.

Result: 16/11

Thus the program to implement flow control at data link layer using cisco packet simulation is executed successfully

Aim: Simple PACKE

classless allows for by allowin just de fa means the space int useful u of IP add networks

i) creating The 1st topology Network

ii) Adding Once we can be ad

Subnetti To su to pro addre the

Practical - 9

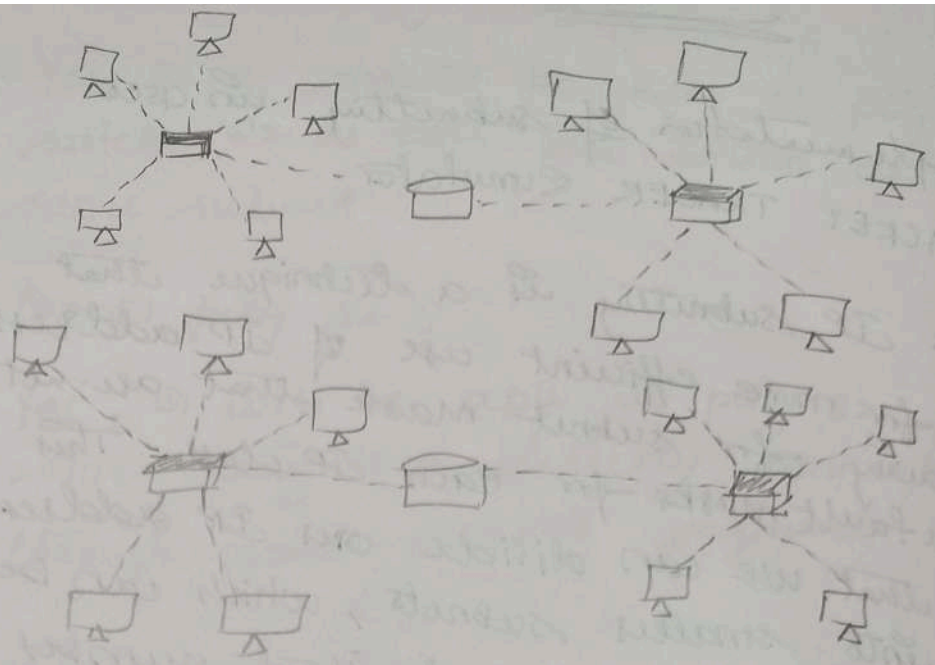
Aim: Implementation of subnetting in CISCO PACKET TRACER Simulator.

classless IP subnetting is a technique that allows for more efficient use of IP addresses by allowing for subnet mask that are not just 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 a limited number of IP addresses but need to create multiple networks.

i) creating a Network topology:
The 1st step is to create a network topology in Packet Tracer. To create, select Network and Generic.

ii) Adding the devices:
Once we have created our network topology, we can add devices to it. Here we will be adding routers, switches and PCs.

Subnetting:
To subnet the network address of 192.168.10/24 to provide enough space for at least 5 addresses for end devices, the switch and the routers we can use a /27 subnet mask.



Configuring the devices:

Now that we have added our devices and connected them, we can start configuring them. This will open the command-line interface (CLI) for the routes. In the CLI enter the following commands:

```
#enable
#configure terminal
#interface FastEthernet 0/0
#ip address {IP address}{Subnet mask}
#no shutdown
#exit
interface FastEthernet 0/1
ip address {IP address}{subnet mask}
no shutdown
exit
```

To configu
on the s
1. Right
2. Enter

enable
configure
interface Gi
ip address
no shutdow
Ctrl+t.

Testing the
Open a com
to ping
then the
can also
connectivi

Student Ob

a) Write
Subnetti
network
(subnets

b) Advanta
• Imp
• Enh
• Effi
• Re

To configure the Gigabit Ethernet interface on the routes, you can follow these steps:

1. Right-click on the routes and select CLI
2. Enter the following commands:

```
enable
configure terminal
interface GigabitEthernet 0/0
ip address {ip address} {Subnet mask}
no shutdown
Ctrl+Z
```

Testing the Network:

Open a command prompt on each PC and try to ping the other PC. If the ping is successful, then the network is functioning properly. We can also use the "ping" command to test connectivity between the routes and the PCs

Student Observation:

a) Write down your understanding of subnetting

Subnetting is the process of dividing a large network into smaller, more manageable subnetworks (subnets) Each subnet can operate independently.

b) Advantage of Implementing subnetting:

- Improved Network Management:
- Enhanced security
- Efficient IP address Utilization:
- Reduced Network traffic.

c) Subnetting Implementation in college:

Research and Mapping

To determine if subnetting is in place at your college, consult with network administration team. They can provide insights on how the IP address are segmented

Subnet Diagram and IP address list:

If subnetting is implemented, you can create a visual representation (a network diagram) and list the subnets.

AIM:

a) Enter network TRACER

In this computers copper straight network, PDU is

Procedure:

Step 1:

Configure

i) Select

ii) Press ENT

iii) Type E mode.

Step 2:

Configu

1. Assign

2. Select

select

address,

3. Assign

and gat

Step 3:

Connecting

1. Fast E

2. Fast-Ether

Practical - 10

AIM:

a) Internetworking with routers in CISCO PACKET TRACER Simulator

In this network, a router and 2 PCs are used computers are connected with router using a copper straight through cable. After forming the network, to check network connectivity a unique PDU is transfered from PC0 to PC1

Procedure:

Step 1:

Configure Router(1):

i) Select the router and open CLI
ii) Press ENTER to start configuring Router1
iii) Type Enable to activate the privileged mode.

Step 2:

Configuring PCs

1. Assign IP addresses to every PC in the network
2. Select the PC, Go to the desktop and select IP configuration and assign an IP address, default gateway, subnet mask.
3. Assign the default gateway of PC0 as 192.168.10.1 and gateway of PC1 as 192.168.20.1

Step 3:

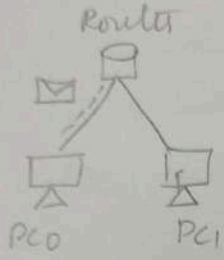
Connecting PCs with Routers

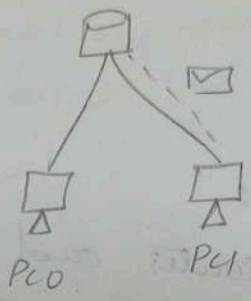
1. Fast Ethernet 0/0 port of Router1
2. Fast Ethernet 0/1 port of Router1

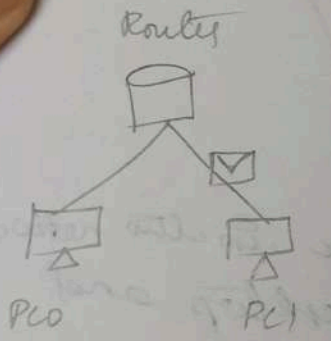
Device name	IPaddress FastEthernet 0/0	Subnet Mask	IPaddress FastEthernet 0/1
Router1	192.168.10.1	255.255.255.0	192.168.20.1

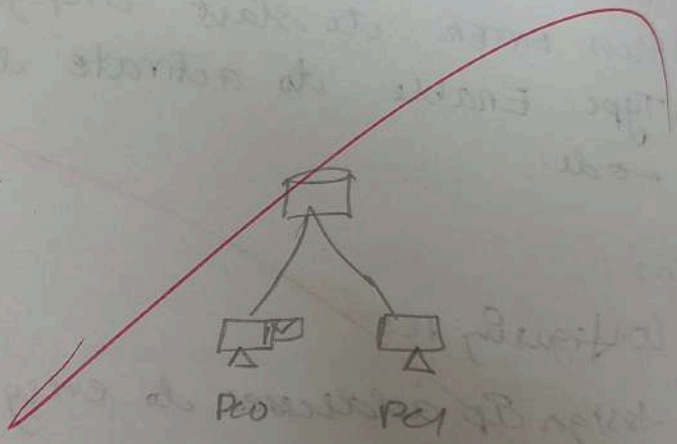
PC configuration table:

Device name	IPaddress	subnet mask	Gateway
PC0	192.168.10.2	255.255.255.0	192.168.10.1
PC1	192.168.20.2	255.255.255.0	192.168.20.1



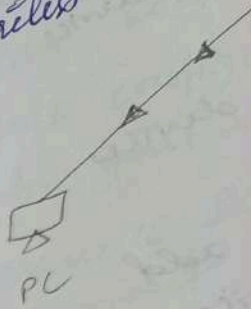






Result: Thus the implementation of subnetting is executed successfully

AIM:
b) Design and
wireless



Addressing

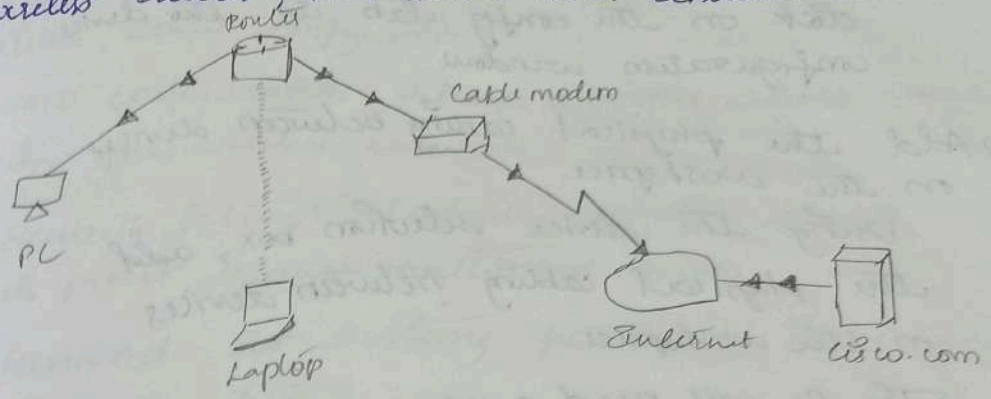
Device	IP
PC	Eth
Wireless Router	LA
Wireless Router	Bui
Cisco.com Server	Et
Laptop	W

Objectives:

Part1: Build
Topology
Step1: Laun
Step2: Build
a) Add
To
Choose

AIM:

b) Design and configure an internetwork using wireless routers, DHCP servers and Internet cloud.



Addressing Table:

Device	Interface	IP address	Subnet mask	Default gateway
PC	Ethernet 0	DHCP		192.168.0.1
Wireless Router	LAN	192.168.0.1	255.255.255.0	
Wireless Router	Internet	DHCP		
Cisco.com Server	Ethernet 0	208.61.220.220	255.255.255.0	
Laptop	Wireless	DHCP		

Objectives:

Part 1: Build a simple Network in the Logical Topology workspace

Step 1: Launch packet Tracer

Step 2: Build the Topology

a) Add network devices to the workspace

To place a device onto the workspace choose a device type from device type selection box

b) Change to the workspace.
Click on the config tab on the device configuration window.

c) Add the physical cabling between devices on the workspace
Using the device selection box, add the physical cabling between devices on the workspace.
The PC will need a copper straight-through cable to connect the wireless router

Part-2: Configure the network devices
Step-1: Configure the wireless router
a) config
Step 2: Configure the laptop
Step 3: Configure the PC
Step 4: Configure the Internet Cloud.
Step 5: Configure the Cisco.com servers.
a) Configure the Cisco.com servers as a DHCP server.
b) Configure the Cisco.com servers as a DNS server to provide domain name.
c) Configure the Cisco.com server Global setting

Part-3:
Step 1: Refresh the IPV4 setting on the PC
a) Verify that the PC is receiving IPV4 configuration information from DHCP
b) Test connectivity to the Cisco.com server from the PC

1) Write
wireless
wireless Route
• SSID config
for your
• Security
to protect
• Password
to the
• Channel
that mini
networks

2. Significance
The Dynami
servers is
it simplifies
assigning IP
• Automatic
assigns
configura
• Supports scala
to add a
large

Student Observation:

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

Wireless Router configuration

- SSID configuration: Set up a unique network name for your wireless network.
- Security Settings: Configure network security to protect against unauthorized access.
- Password: Set a strong password for connecting to the network.
- Channel selection: Choose a wireless channel that minimizes interference from other networks or devices

2. Significance of DHCP server in Internetworking:

The Dynamic Host configuration Protocol (DHCP) server is crucial in internetworking because it simplifies and automates the process of assigning IP addresses to device in a network.

- Automatic IP assignment: DHCP dynamically assigns IP addresses, reducing manual configuration and preventing IP conflicts.
- Supports Scalability: DHCP servers make it to add and manage multiple devices across large networks.

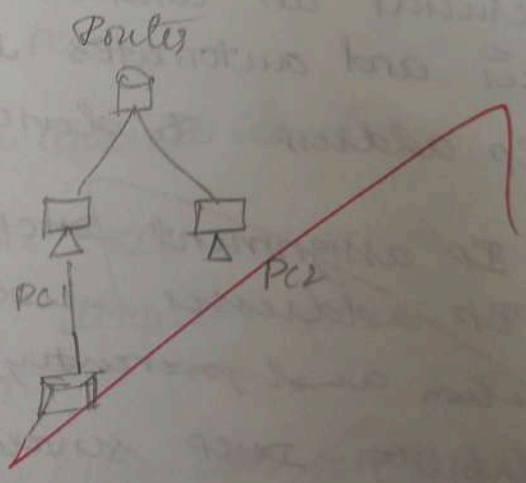
3. Design and configuration of an Ethernet network in a lab

Steps to design and configure an Ethernet

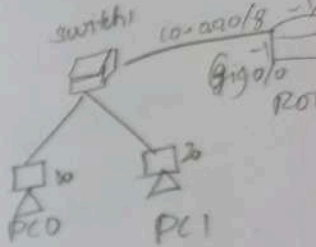
1. Hardware Requirements:
   - One switch
   - One router
   - Ethernet cables

2. Network layout
   - Router
   - switch
   - Devices

3. Configuration steps:
   i) Connect the router to switch
   ii) Configure the router's interface with an address.
   iii) Configure the DHCP server on the router
   iv) Connect PCs to the switch
   v) Test connectivity.



Aim:
a) Simulate static
Cisco Packet
static routes
add its the
process of ad
routing labs



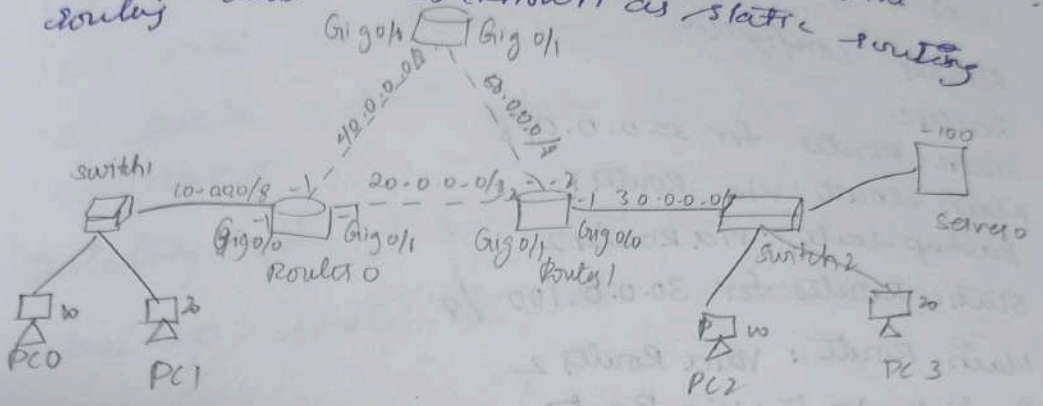
creating, adding
Routers a
networks. We
networks
routers
Router
Avail
on
Router 0
Router 1
Router 2

Practical - 11

AIM:

a) Simulate static Routing configuration using Cisco Packet Tracer

static routes are the routes you manually add to the routers routing table. The process of adding static routes to the routing table is known as static routing



creating, adding, verifying static routes

Routers automatically learn their connected networks. We only need to add routes for the networks that are not available on the routers interfaces

Router	Available networks on local interfaces	Networks available on other routers interfaces
Router 0	10.0.0.0/8, 20.0.0.0/8, 40.0.0.0/8	30.0.0.0/8, 50.0.0.0/8
Router 1	20.0.0.0/8, 30.0.0.0/8, 50.0.0.0/8	10.0.0.0/8, 40.0.0.0/8
Router 2	40.0.0.0/8 50.0.0.0/8	10.0.0.0/8, 20.0.0.0/8, 30.0.0.0/8

Static routing involves manually adding routes routing table.
The main table is prioritized over backup routes based on the fewest no. of routers or link bandwidth.

Routes configura

Routes 0
Static Routes for 30.0.0.0/8
Main route: via Routes 1.
Backup route: via Routes 2
Static Routes for 30.0.0.100 /8
Main Route: Vaia Routes 2
Backup route: via Routes 1
Static Routes for 50.0.0.0/8
Main Route: Via Routes 2
Backup route: via Routes 1

Routes 1
Static Routes for 10.0.0.0/8
Main route: via Routes 0
Backup route: via Routes 2
Static Routes for 40.0.0.0/8
Main route: Via Routes 0
Backup route: via Routes 2

Routes 2
Static Routes for 10.0.0.0 /8 and 30.0.0.0/8
Both routes are verified for correct configuration

verification of Static Routing
Verify the static routes on Routes 0 by sending ping requests and using the Tracert command to track the paths to the configured networks

To test
of main
between

Deleting
Use the
to view
To delete
command
Deleting
backup

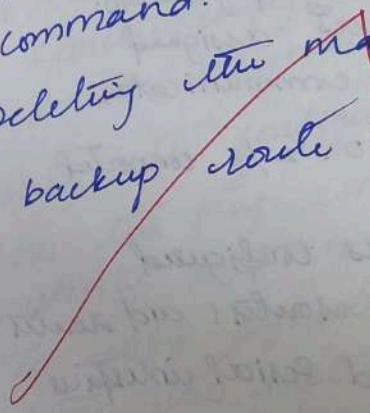
To test backup routes, simulate a failure of main route by deleting the link between Router0 and Router1

Deleting Static Routes

Use the show ip route static command to view all static routes

To delete a route, use the no ip route command.

Deleting the main route allows the backup route to take precedence



AIM:

To simulate the Routing Information Protocol (RIP) using Cisco Packet Tracer, configuring routers and PC devices to communicate effectively.

Initial configuration

Device Interfaces and IP configuration:

Each device's interface is assigned specific IP addresses for communication

PC0: Fast Ethernet 10.0.0.2/8 is connected to Router 0 Fa0/1

Router 0: Various interfaces configured for communication with router1 and router2 using both Fast Ethernet and Serial interfaces

Steps for configuration

1. Assign IP addresses to PCs

Access the PCs desktop and navigate to the IP configuration section to assign IP addresses as per the table provided.

2. Configure Router Interfaces

Enter the command line Interface (CLI) for each router to assign IP addresses to their respective interfaces using commands like:

Router> enable

Router# configure terminal

Router(config) interface fastEthernet 0/0

Router(config-if) ip address [IP address] [Subnet m

Router(config-if) no shutdown

3. Setup
Addi
serial
clak
Ro
Rou

4. Enable
configu
to shar
Rout
Route

Testing
Use the
connectivity
To verif
between
failure
traffic

Result:

The conf
a network
ensuring
adapts
transmis

3. Setup Serial Interfaces:
Addition commands are required for serial interfaces, including setting the clock rate and bandwidth at DCE end:
Router (config-if) clock rate [rate]
Router (config-if) bandwidth [value]

4. Enable RIP Routing Protocol
Configure the RIP protocol to allow routers to share routing information.
Router (config) router rip
Router (config-router) network [Network Address]

Testing the configuration
Use the ping command from PC1 to test connectivity with PC0
To verify dynamic routing remove the cable between two routers to simulate a route failure and observe how RIP reroutes traffic automatically.

Verified

Result:
The configuration successfully establishes a network that utilizes RIP for dynamic routing ensuring that if one route fails. The system adapts by finding alternative paths for data transmission

9) AIM:

Implument echo client serves using TCP/UDP sockets.

Algorithm:

TCP serves algorithm:

```
import socket
def tcp_echo_serves(host='127.0.0.1', port=65432):
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                                        as serves_socket:
        serves_socket.bind((host, port))
        serves_socket.listen()
        print(f"tcp echo serves listening on {host}:{port}")
        while true:
            conn, addr = serves.socket.accept()
            with conn:
                print(f"connected by {addr}")
                while true:
                    data = conn.recv(1024)
                    if not data:
                        break
                    conn.sendes(data)

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

9.

TCP Ech
import soc
def tcp
with
clien
mess

client_se
data =
print(

if __nam
to

Output:-
Enter mes
Received b

Result:
Implem
TCP/UDP
is ver

Practical - 12

a) AIM:

Implement echo client servers using TCP/UDP sockets.

Algorithm:

TCP servers algorithm:

```
import socket
def tcp_echo_servers(host='127.0.0.1', port=65
    with socket.socket(socket.AP_INET, socket.
                                    -STREA
                                    as servers_socket:
        servers_socket.bind((host, port))
        servers_socket.listen()
        print(f"tcp echo server listening on {host
                                                {port}
        while true:
            conn. addr = servers.socket.accept()
            with conn:
                print(f"connected by {addr}")
                while true:
                    data = conn.recv(1024)
                    if not data:
                        break
                    conn.sendall(data)

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

TCP Ech
import soc
def tcp
with
clie
mes

client_s
data =
print(

if __na

Output:-
Enter me
Recived d

Result:
Implu
TCP/UDP
is

TCP Echo client:

```
import socket
def tcp_echo_client(host='127.0.0.1', port=
                    65432)
    with socket.socket as client
    client_socket.connect((host, port))
    message = input("enter message:")

client_socket.sendall(message.encode())
data = client_socket.recv
print(f"Recived from server: {data.decode()}")
if __name__ == "__main__":
    tcp.echo_client()
```

Output:-

Enter message to echo: hello

Recived from server: hello

Result:

Implemented echo client server using TCP/UDP sockets successfully & output is verified.

12. b)

AIM:

Implement chat client server using TCP/UDP sockets.

Algorithm:

- Create a socket using TCP (sock.stream)
- Connect the sockets IP address
- Take input from the user
- Print received a response from the source
- Print server's response on client side.

TCP server code

```
import socket
ser.host = '127.0.0.1'
servs.port = 12345
servs_Soket = socket.socket(socket.AF_INET)
servs = soket.bind((servs_host, servs_port))
while True:
    Message = client_socket_ recv(1024).decode
    if message.lower() == "quit":
        print("client has ended the chat.")
        break
    print(f"client: {message}")
    response = input("servs: ")
    client_socket.send(response, encode())
    if response.lower() == "quit":
```

```
client_socket.close()
server_socket.close()
```

TCP client code

```
import socket
server_host = '127.0.0.1'
server_port = 12345
client_socket = socket.socket(socket.AF_INET)
client_socket.connect((server_host, server_port))
while True:
    message = input("chat:")
    client_socket.send(message.encode())
if message.lower() == 'quit':
    print('client ended chat')
    break
response = client ended chat")
break
response = client_socket.recv(1024).decode()
print(f"server: {response}")
client_socket.close()
```

Output:

Enter message for server: "Hello server"

Reply: Hello client.

Result: 16/11

Thus the implementation of TCP is successfully

Practical 13

AIM:

Implement your own ping program

Algorithm:

- Open a raw socket to send ICMP
- Create the ICMP Echo request packet including a header and data.
- Send Packet. Send the ICMP request to target host
- calculate the time
- Show response

Code:

```
import os
import struct
import socket
import time
import select

ICMP_Echo_request = 8
ICMP_Echo_Reply = 0

def checksum(data):
    sum = 0
    count_to = (len(data) // 2) * 2
    count = 0
    while count < count_to:
        this_val = source_String[count + 1] * 256
        sum += this_val
```

```
        sum2 = 0xffffffff
        count += 2
    if count_to < len(source_string):
        sum += source_string[-1]
        sum2 = 0xffffffff
    sum = (sum >> 16) + (sum2 & 0xffff)
    sum += (sum >> 16)
    answer = ~sum
    answer2 = 0xffff
    answer = answer >> 8 | (answer << 8 & 0xff00)
    return answer

def create_packet(id):
    header = struct.pack("bbHHh", ICMP_
                         ECHO_REQUEST, 0, 0, id, 1)
    data = struct.pack("d", time.time())
    return struct.pack("bbHHh", ICMP_ECHO_
                       REQUEST, 0, socket.htons(
                       checksum(header + data)

def do_one_ping(dest_addr, timeout=1):
    sock = socket.socket(socket.AF_INET, socket.
                         SOCK_RAW, socket.getprotobyname("icmp"))
    packet_id = os.getpid() & 0xFFFF
    sock.sendto(create_packet(packet_id), (dest_addr, 1))
    start_time = time.time()
    sock.settimeout(timeout)

def ping(host, count=4):
    dest_addr = socket.gethostbyname(host)
    for i in range(count):
```

```
    delay = do_one_ping(dest_addr)
    if delay is None:
        print("Request timed out")
    else:
        print(f"Reply from {dest_addr}:
              time={delay*1000.2f}ms")

    time.sleep(1)

if __name__ == "__main__":
    target_host = "google.com"
    ping(target_host)
```

Output:

Enter host to ping: google.com

Reply: time = 21.45ms

Result:

Thus ping program is successfully run and output is verified successfully

16/1

AIM:

write a co
implement

Algorithm:

- creati
- continu
  using
- parse
  sour
- close
  capture

code:

```
from scapy.a
from scap
def pac
    if
        p
        p
```

Practical 14

AIM:
Write a code using RAW sockets to implement packet sniffing.

Algorithm:
- create a raw socket
- continiously capture incoming packets using recvfrom().
- parse and display information like the source and destinated IP addresses & and protocol type
- close the socket after finishing the capture process.

Code:

```
from scapy.all import sniff
from scapy.layers.inet 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:
```

```
        print( f"protocol : {protocol_name}")
        print (f"source IP: {src_ip}")
        print(f"Destination IP: {dst_ip}")
        print("-" * 50)

def main():
    sniff (prn=packet_call back, filter = "ip",
                                 store = 0)

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

Output:

```
protocol : TCP
Source IP : 192.168.1.2
Destination IP : 93.184.216.34
--------------------------------------------------
Protocol : TCP
Source IP: 192.168.1.2
Destination IP: 172.217.14.206
--------------------------------------------------
protocol : UDP
Source IP : 192.168.1.2
Destination IP : 8.8.8.8
```

Result:

Thus the program to implement packet sniffing using Raw socket is implemented successfully.

AIM:
To analy
logs

Procedure
step 1:
step 2:
step 3:

clear Logfiles

Input
Logfile:
C:\Users\

Target
C:\Users
Clear exis
Delete al

Practical – 15

AIM:

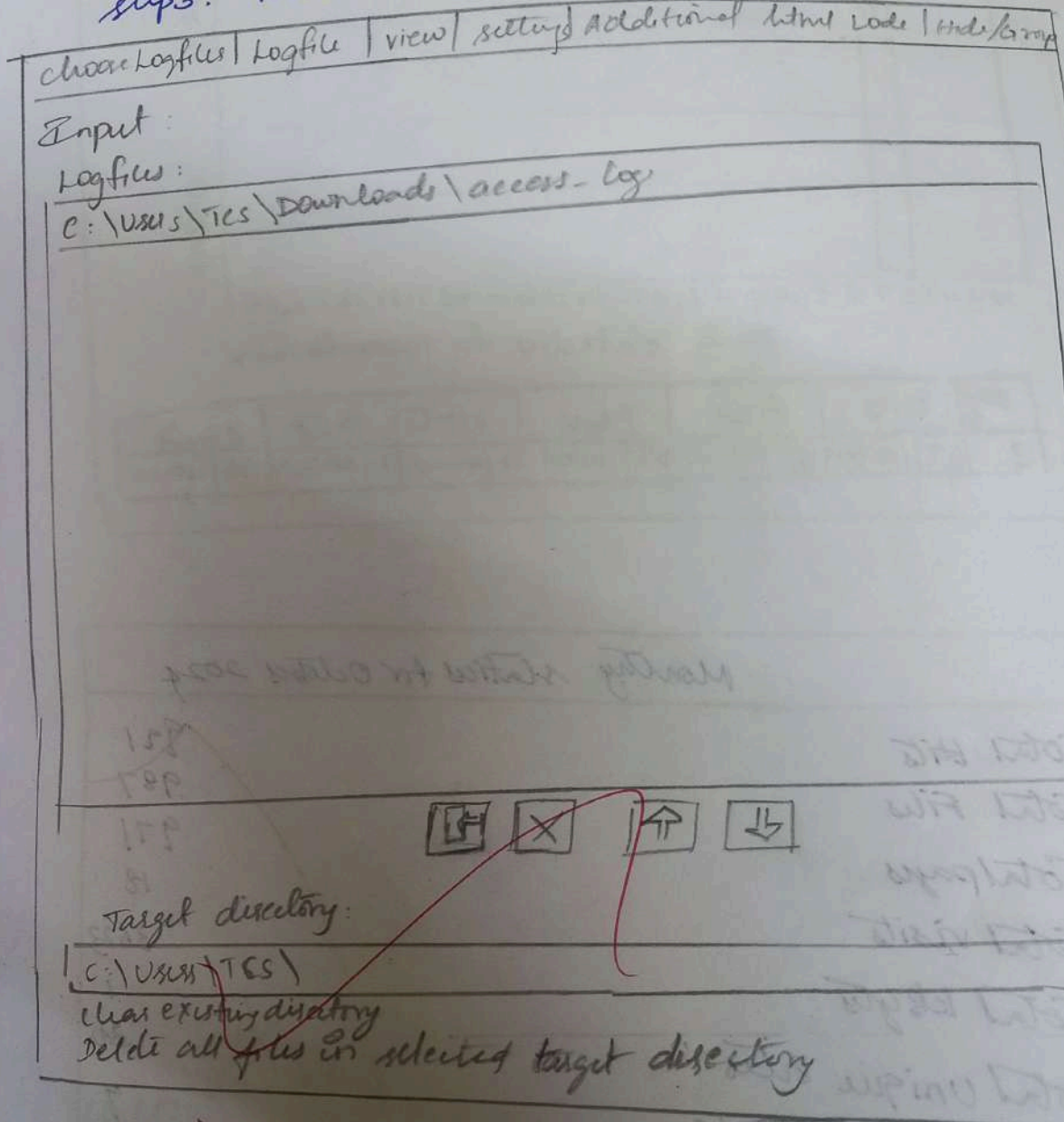
To analyze the different types of web logs using webalizer tool.

Procedure:

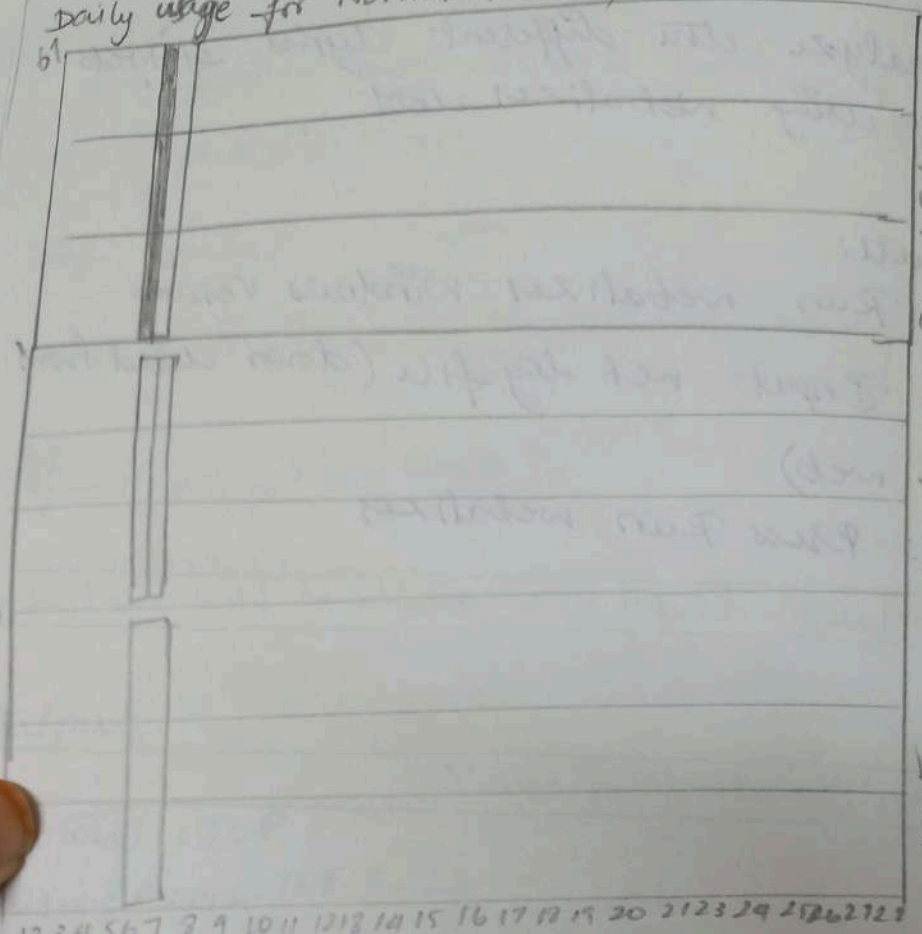
step 1: Run webalizer Windows version

step 2: Input web log file (download from web)

step 3: Press Run webalizer



Daily usage for November 2024



Daily statistics for November 2024

Day	Hits		Files		Pages		Visits		Sites		KBytes	
3	67	100.00%	65	100.00%	67	100.00%	1	100.00%	1	100.00%	139	100.00%

Monthly statistics for October 2024

Total Hits	971
Total Files	987
Total pages	971
Total visits	18
Total KBytes	3623
Total Unique Sites	1
Total Unique URLs	30
Total Unique Referrers	7
Total Unique User Agents	3

Hits per Hour
Hits per Day
Files per Day
pages per Day
sites per Day
visits per D
KBytes per D

Result:

Thus

	Avg	Max
Hits per Hour	2	180
Hits per Day	49	247
Files per Day	49	245
Pages per Day	48	245
Sites per Day	0	1
Visits per Day	0	5
KBytes per Day	181	906

Result: 16/11

Thus the different types of web logs using webalizer tool is executed successfully.

Completed

16/11