INDEX

NAME: Barath raj STD.: CSE SEC.: A ROLL NO.: 038 SUB.: CN-LAB

S. No.	Date	Title	Page No.	Teacher's Sign / Remarks
1.	13/9/24	study of various network commands used in Linux and windows		
2.	27/7/24	study of different type of Network cable		
3.	30/7/24	Experiments on CISCO Packets Tracer (Simulation Tool).		
4.	17/8/24	Setup and configure LAN using switch & ethernet cable		
5.	20/8/24	experiment : Packet capture food Wie. shark		
6.	20/8/24	Hamming code		
7.	24/8/24	Sliding window Protocol		
8	4/10/24	LAN config using CISCO		
9.	8/10/24	Subnetting in CISCO		
10	15/10/24	Inter networking in CISCO		
11	18/10/24	Routing at Network layer		
12	22/10/24	End-End communication		
13	25/10/24	Ping Program		
14	29/11/24	Packet sniffing		
15	5/11/24	Types of serialization tool.		

completed
23/11

Study of various Network commands used in Linux and Windows

Ex:NO:1

AIM:
study of various Network commands used in Linux and Windows

Basic network commands:

arp-a: display the IP address of your computer along with the IP and MAC address of your router.

output: Interface : 192.168.209.1 ---- 0.5

Internet address	Physical address	Type
192.168.209.255	ff-ff-ff-ff-ff	static
224.0.22	01-00-5e-00-00-16	static
224.0.0.251	01-00-5e-00-00-fb	static

hostname: TCP/IP command that display the name of your computer

output: LAPTOP-LMTODIS2

ipconfig/all: display details TCP/IP configuration information, including Router, Gateway, DNS, DHCP and Ethernet adapter info

output: windows IP configuration

Host Name . . . . . : LAPTOP-LMTODIS2
Primary DNS Suffix . . . . :
Node Type . . . . . : mixed
IP Routing Enabled . . . . : NO
WINS Proxy Enabled . . . . : NO

netstat -a: helps solve problem with Net BIOS name resolution (Net BIOS over TCP/IP)

output: NBTSTAT [-a Remote Name] [-A IP Address] [-c] [-n] [-r] [-R] [-RR] [-s] [-S] [interval]

netstate: display statistics about active TCP/IP connections, including network connections, routing table and interface statistics

output: Active Connection

Proto	local address	Foreign address	state
TCP	127.0.0.1:49674	3ca52zn0mj:6500	ESTABLISHED
TCP	127.0.0.1:49675	3ca52znumj:49690	ESTABLISHED
TCP	127.0.0.1:49676	3ca52znumj:49675	ESTABLISHED

nslookup: Tool used to perform DNS lookup in Linux, displaying details such as IP addresses, MX records and NS servers of a domain.

output: Server: dns.google.com Address: 8.8.8

Non-authoritative answer:

Name: google.com

Addresses: 2404:6800:4007:81b::200e

142.250.182.78

Pathping: combines Ping and Tracert, tracing the route to a destination and testing each router along the way to gather data loss statistics.

output: usage: pathping [-g host-list] [-h maximum-hops] [-i address] [-P period] [-q num-queries] [-w timeout] [-4] [-6] target

Ping: test connection between two nodes using ICMP (Internet control message Protocol) and can be passed with a host-name / IP address or fully qualified domain name.

output:

Ping statistics for 142.250.182.78:

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

Approximate round trip times in milli-seconds:

Minimum = 34ms, Maximum = 1955 ms, Average = 831 ms

route: shows / manipulates the IP routing table and is used to set up static routes to specific hosts or networks via an interface

output:

Route [-f] [-p] [-4] [-6] command [destination]

[MASK Netmask] [gateway] [METRIC metric] [IF Interface]

Some Important LINUX commands

1. **ip**: Essential for administrators, used to show address information, manipulation routing, and display network devices, interface and tunnels

Command Syntax: ip <options> <object> <command>

a) Show IP address assigned to an interface ip address show.

output:

ens 33: <BROADCAST, MULTICAST, UP, LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000 link/ether 00:0C:29:eb:03:40 brd ff:ff:ff:ff:ff:ff

altname enp2s1

inet 192.168.209.130/24 brd 192.168.209.255 scope

global dynamic noprefix route CNS 33

valid - lft forever preferred - lft forever

b) Assign an IP to an interface : ip address 192.168.209.130/24 dev CNS 33

c) delete an IP to an interface : ip address del 192.168.209.130/24 dev CNS 33

d) Bring an interface online : ip link set CNS 33 up

e) Bring an interface offline : ip link set CNS 33 down

f) Enable promiscuous mode for an interface : ip link set CNS 33 up

g) display the route taken for a specific IP : ip route get 10.10.1.4

output :

10.10.1.4 via 192.168.209.2 dev CNS 33 192.168.209.130 uid 0

cache

2) ifconfig : stable for configuring and trouble shooting networks, has been replaced by the 'ip' command

output : CNS: 33: flags = 4419 <UP, BROADCAST, RUNNING, PROMISC, NOTRAILERS>

mtu 192.168.209.130 netmask 255.258.255.0 broadcast

inet 6 fe 80::20c:29ff:fec6:346 prefixlen 64 scope 0x60>

ether 00:0c:29:e6:03:46 txqueuelen 1000 (Ethernet)

RX packets 935 41 bytes 1410749603 (141.0 MB)

RX errors dropped over runs 0 frame 0

TX packet 258 71 bytes 159187 (1.5 MB)

TX errors 0 dropped 0 over runs 0 carrier 0 collisions 0

3. mtr : combines the functionality of ping and trace route, providing detailed statistics about each hop, including response time and packet loss, to diagnose network issues.

Command syntax : mtr <options> hostname / IP

a) show statistics including each hop with time and loss %.

mtr google.com

output :

(192.168.209.130) → google.com (142.250.182.78)

Host	Packets				Ping		
	loss %	snt	last	Avg	Best	wrst	stdev
1- gateway	0.0	261	0.7	0.6	0.3	10.7	1.6
2. maa05g20-in-f14.1e100.net	0.0	294	6.3	7.1	5.2	269.1	17.1

b) show numeric IP address instead of host name:

'nslookup google.com'

c) show both numeric IP address and host name:

'nslookup -d google.com'

d) set the number of pings to send:

nslookup -c 10 google.com

4. tcp dump: designed for capturing and displaying packets

a) Install 'tcp dump': 'sudo apt-get install tcpdump

output:

Reading Package lists ... Done
Reading dependency tree ... Done
Building dependency tree .... Done
Reading state information ..... Done
tcp dump is already the newest version (4.99.4-3ubuntu0.1)
tcp dump set to manually installed
0 upgraded 0 newly installed 0 to remove and 0 not upgraded.

b) List all available interface for capturing: 'tcp dump -D'

output:

1. eth0 33 [up, running, connected]
2. lo [up, running, loop back]

c) Capture traffic on 'eth 33': 'tcp dump -i eth 9'

output:

12 packets captured
12 packets received by filter
0 packets dropped by kernel

5. Ping: Verifies IP-level connectivity by sending ICMP Echo message and displaying Echo reply message with round-trip messages.

output:

ping google.com

--- google.com ping statistics
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 10.364/30.444/44.225/12.518ms

Configuring An ethernet connection 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. List connection profiles: 'nmcli connection show'
2. Add a new ethernet connection (or skip if modifying existing):
   'nmcli connection add con-name "DNA" if name ens33 type ethernet
3. optionally rename the connection profile:
   'nmcli connection modify 'wired connection 1"
4. display current settings:
   'nmcli connection show'
5. configure IPv4 settings
   'nmcli connection modify 'wired connection 1" ipv4.method auto
6. configure IPv6 settings:
   'nmcli connection modify 'wired connection 1' ipv6.method auto
7. Activate the profile:
   'nmcli connection up wired connection 1

Verification:

1. Display IP settings:
   'ip address show ens33
2. Display IPv4 default gateway
   'ip route show default
3. Display DNS settings:
   'cat /etc/resolv.conf
4. Display IPv6 default gateway
   'ip -6 route show default'
5. Verify connectivity with 'Ping':
   'ping google.com'

nmcli

Student observation:

1. which command is used to find the reachability of a host machine from your device? - 'Ping'
2. which command will give the details of hops taken by a packet to reach its destination? - 'trace route'
3. which command display the ip configuration of your Mac - 'ip config'
4. which command display the TCP port status in your machine - 'net stat'
5. write the modify the ip configuration in a Linux machine - 'nmcli connection modify "wired connection 1"

31/8/22

Result:

Thus the study of various network commands used in Linux and Windows and executed successfully

Study of different type of Network cable

Exp No: 2

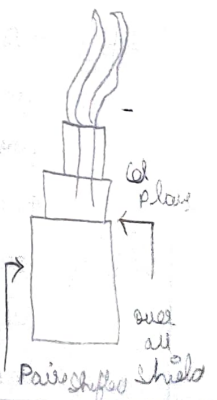
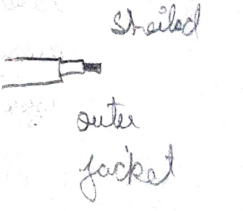
Date:

Aim: Study of different type of Network cable

a) Understand different types of network cables:

Different type of cable 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	Advantage / disadvantage	Application / user	Image
UTP	category 3 category 5 category 5e	10 mbps up to 100mbps	Advantage: - cheaper in cost - easy to install as they have small overall diameter disadvantage: - more prone to electromagnetic interface and noise (EMI)	10Base-T ethernet Fast Ethernet Gigabit ethernet Fast ethernet Gigabit ethernet	
STP SSTP	category 6, 6a category 7	10 Gbps 10 Gbps	Advantages: - Shielded - Faster than UTP - Less susceptible to noise and interface disadvantages: - Expensive - Greater installation effort	Gigabit Ethernet 10 Gb ethernet (55m) widely used in data centers Gigabit ethernet 10G Ethernet (100m)	
Coaxial cable	RG-6 RG-59 RG-11	10-100mbps	Advantage: - High bandwidth - Immune to interface - Low loss bandwidth - Versatile disadvantage: - Limited distance - cost - Size is bulky	Speed of signal is 500m Cable television network High speed internet connection	

fiber optics	Single Mode multi mode	100Gbps	Advantage: - High speed - High Bandwidth - High Security - long distance disadvantage: - Expensive - Requires Skilled installers	Maximum distance of fiber optics cable is around 100 meter	

b) make own ethernet cross-over cable / straight cable

Tools and Part needed:

- ethernet cabling: CAT 5e is ratified for gigabit support, but CAT 6 cabling works or well, just over short distance
- A Crimming tool: This is an all-in one networking tool that pushed down the pins in the plug and strip and cut the shielding of the cables.
- Two RJ45 plugs.
- optional two plug shields.

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

Step 2: Next, strip approximately 1.5cm of cable shielding from both ends. the Crimming 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. Referring 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 further than others, slightly back to create an even level. the difficult aspect is placing these into the RJ45 plug without messing up the order. To do so, hold the plug with the lip side facing away from you and have the gold pins facing towards you.

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

much shielding. Simply snip the cables back a little more

Step 6: After the wires are securely sitting inside the plug, insert into the crimping tool and push down. It should be placed correctly, but pushing too hard can crack the fragile plug.

Step 7: Lastly Repeat for the other end using Diagram B (to make cross over cables) / using diagram A (to make straight through cable).

- To test it, plug it in and attempt to connect two devices exactly.

Student observation:

1. what is the different between cross cable and straight cable ?
cross cables have crossed wiring for connecting similar devices, while straight cable are parallel wiring for connecting different devices.
2. which type of cable is used to connect two Pc? (straight / cross cable)
cross cable
3. which type of cable is used to connect a router/ switch to your Pc?
straight cable. (straight / cross cable)
4. Find out the category of twisted pair cable used in your lab to connect the Pc to the net work socket?
the category is typically cat 5e or cat 6.
5. write down your understanding / challenges faced and output noted while making a twisted pair cross / straight socket.
making a twisted pair cable involves arranging the wires in the correct order. facing challenges with precise wiring and crimping and ensure reliable network connections upon testing.

3/8/24

Result:

Thus the cable connection is done and executed Sucessfully.

Experiments On CISCO Packets Tracer (Simulation tool)

Ex. No: 3 Date: 30/9/24

Aim: To study the Packet Tracer tool installation and user interface overview.

a) To understand environment of CISCO Packets Tracer to design simple

Introduction:

A simple as the name suggests, simulates networks devices and its environment. Packet Tracer is an exciting network design/ simulation and modelling tool.

* It allow you to model complex system without the need for dedicated equipment.
* It helps you to practice your network configuration and trouble shooting skills via computer or an Android or iOS based mobile devices.
* It is available for both the linux and windows desktop environment.
* Protocols in Packet Tracer are coded to work as to have in the same way as they would on a real hardware.

Installing Packet Tracer:

To download Packet Tracer, go to https:// www. netacad and log in 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 in your laptop).

Windows: Installation in windows is pretty simple and straight forward; the step comes in a single file named PacketTracer - setup 6.exe. Open this file to begin the setup wizard, accept the license agreement choose a location and start the installation.

Linux: Linux user with an Ubuntu / Debian distribution should download the for Ubuntu, and those using Fedora/ Redhat / can must download the file for fedora. Or root executable permission to this file for fedora. Grant executable permission to this file by using chmod and execute it to begin the installation

User Interface overview:

the layout of the Packet Tracer is divided into several components. the components of the Packet Tracer interface are as follows:

1. menu bar: this is a common menu found in all software applications it is used to open, save, print, change preferences and so on.

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

3. logical/Physical workshop 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 tool bar: this tool bar has sides 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. Buttons are also provided to control the time and to capture the packets.

7. Network components box: this component contains all of the networks and devices available with Packet Tracer and is further divided into two areas.

(i) Area Ta: Device-Type selection box - contains Device categories

(ii) Area Tb: 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:- user can create highly-customized packets to test their topology from this area and the results are displayed as a list.

b) Analyse the behaviours of network devices using CISO Packet TRACER simulation.

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

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. Similarly connect remaining 3 PCs to the HUB

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 required as there are only two end devices in the network.

4. click on the PDU (message icon) from the common tool box. Drag and drop it on one of PC (source machine) and then drop it on one of PC (source machine) and then drop it on another PC (destination machine) connected to the HUB.

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

6. Repeat step 3 and step 5 for the PCs connected to the switch.

7. Observe how HUB and Switch are forwarding the PDU and write your observation and conclusion about the behaviour of Switch and HUB.

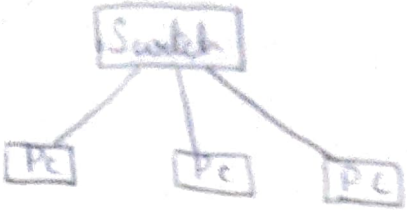
Switch observation

a) From your observation, write down the behavior of switch and hub in terms of forwarding the packet received by them.

A hub broadcasts packets to all ports, while a switch forwards packets only to the destination port based on MAC address.

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

A star topology is implemented where each PC is connected to a central switch or hub.



31/8

Result:

Thus the Packet Tracer tool is used and executed successfully.

Setup and configure LAN using Switch & ethernet cable

EX.NO: 4 Date:

Aim:

Setup and configure a LAN (Local area network) using Switch and ethernet cables in our labs.

What is a LAN?

A Local Area Network (LAN) refers to a Network that connects devices within a limited area, such as Office building, School or home. It enables users to share resources including data, Printers and internet access.

How to setup LAN:

① - Plan & design an appropriat network topology that into account network requirement.

② - Take 4 computers, a Switch with 8, 16 ports which is Sufficient for network of their sizes and 4 ethernet cables.

③ - Connect your computer to network switch via an ethernet cable.

④ - Assign IP address to your PCs admin.

↳ log on to the client computer as admin

↳ click network and Internet connections.

↳ Right click local area connector ethernet Go to properties → Select Internet protocol.

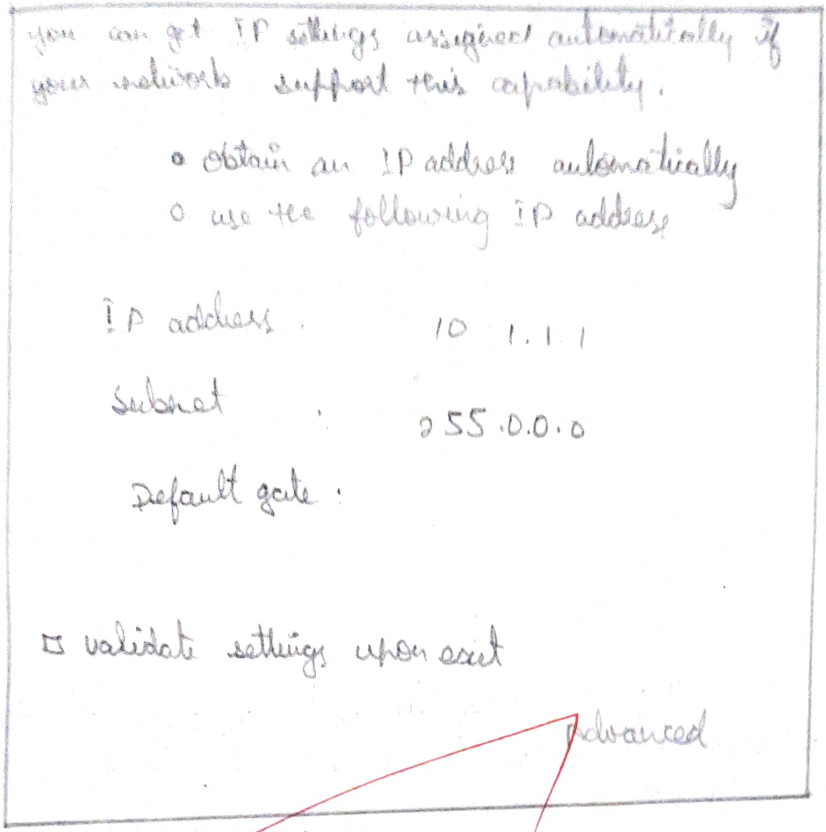
⑤ - configure a network switch.

↳ connect four computer to the switch to switch's web Interface, you will need to connect your computer to switch.

⑥ Check the connectivity between switch and other machine by using Ping command.

(7) - select a folder → go to properties → click share it with everyone.

(8) Try to access the shared folder from other computer of the network.



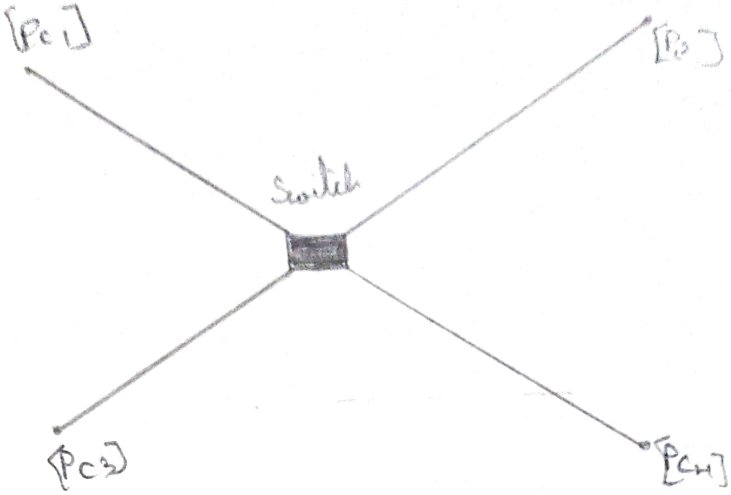
Result :-

the above experiment has been executed successfully.

3/8.

Student observation

Draw a neat diagram of the LAN in the configuration observation task that you have implemented in your lab. Write the IP configuration of each and every device.



outcome:

Lan was successfully set up and all devices could communicate with each other using their assigned IP address. Shared resources like folder were accessible.

challenges faced:

* ensuring each Pc has a unique IP address to avoid conflicts

* Initial difficulty accessing the switch web due to incorrect IP address.

Ex.No:05

Date:

Aim: experiments on packet capture tool: wireshark

Packet sniffer:

→ sniffs messages being sent / received from / by your computer.

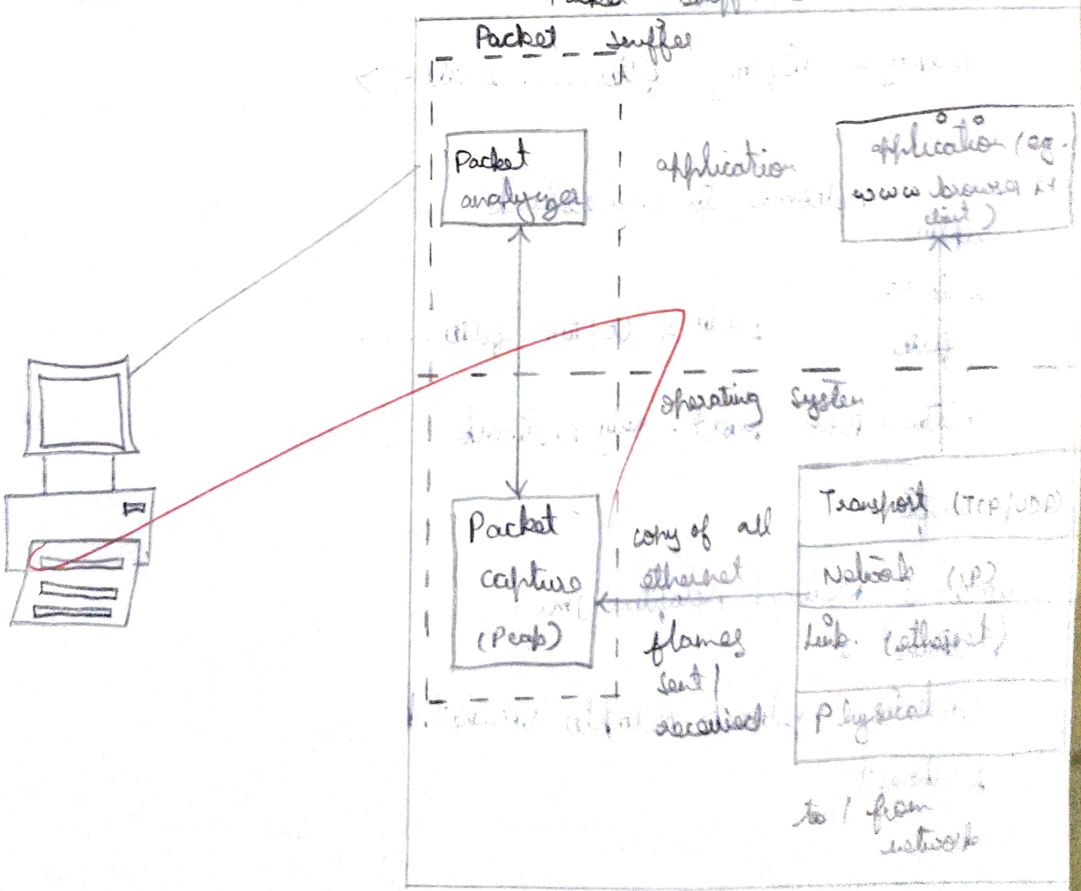
→ store and display the contents of the various Protocol fields in the message

→ Passive program
- never sends packets it self.
- no packets addressed to it.
- receives a copy of all packets (sent / received)

Packet sniffer structure Diagnostic tools:

- Tcpdump - Eg: Tcpdump -src host 10.129.41.2
- Wireshark - wireshark -r exe 3.out.

Packet sniffer structure



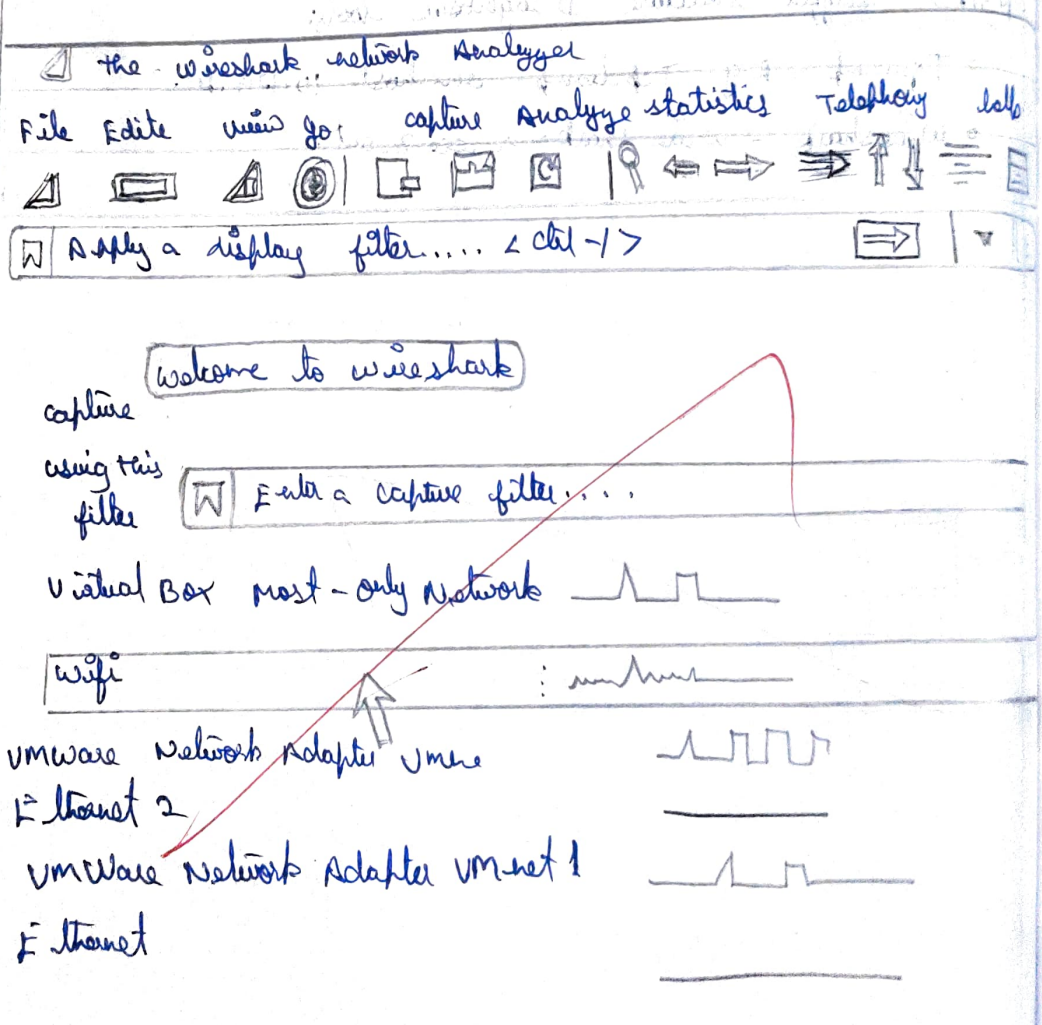
WIRESHARK:

Function:
captures and analyzes network traffic in real-time

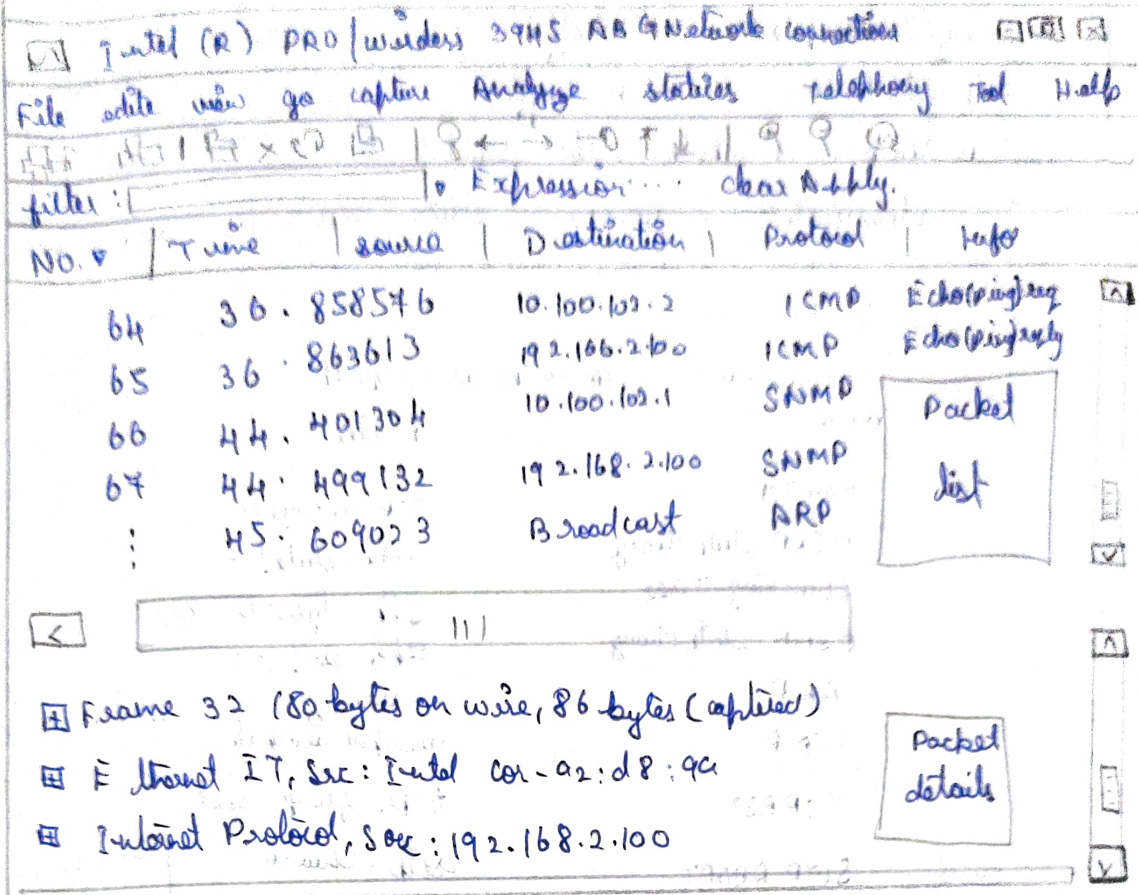
features:
→ capture and decode packets
→ Apply filter
→ view statistic
→ Troubleshoot and analyze network issues

uses:
→ Network trouble shooting
→ security analysis
→ Protocol debugging
→ learning network protocols.

installation:
- Windows / macos: official website.
- Linux : Package repositories.



usage:
start capturing by selecting a network interface
Promiscuous mode captures all packets.



Click the red "stop" button near the left of corner of the window wirer you want to stop capturing traffic

Packet List Pane:
- Displays all Packets in the capture file
- Selecting a wire show detailed info in the other fanes.

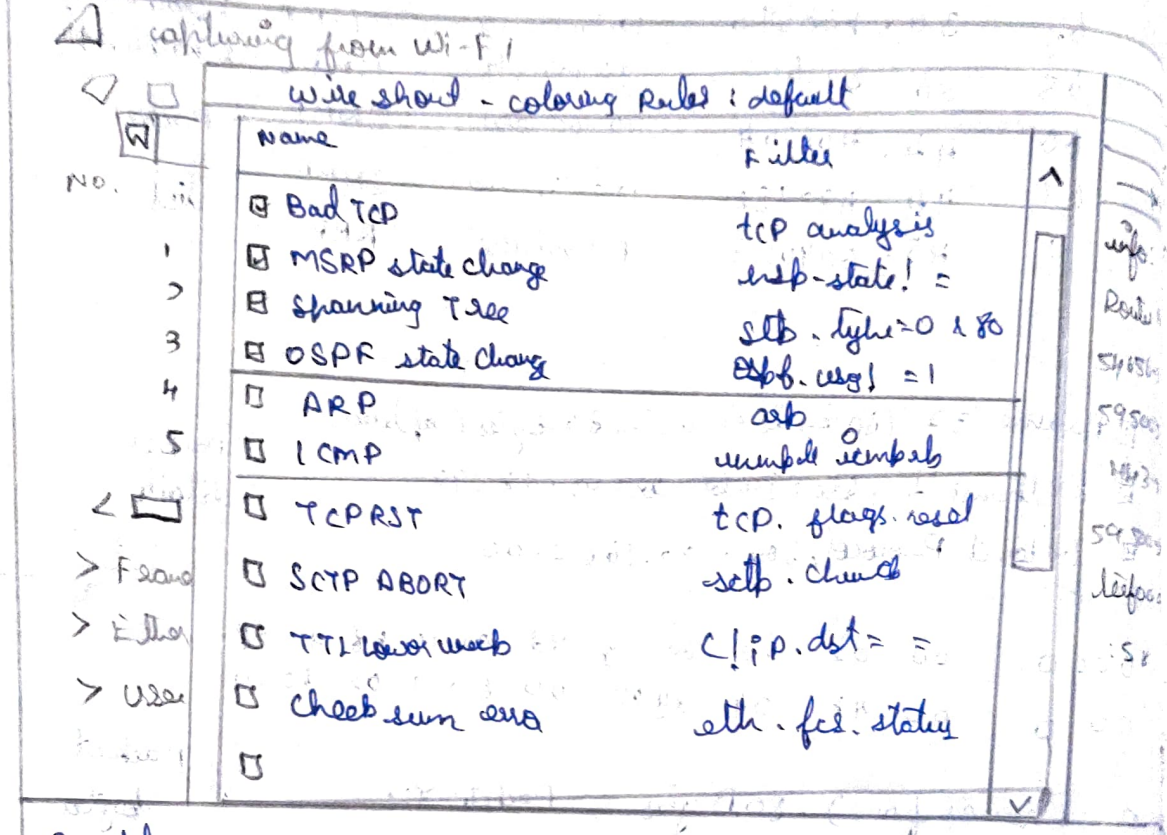
Packet details Pane:
- shows detailed info of the decled Packet including Protocols and fields in a tree format

Packet bytes Pane:

- displays the Packet data in a hexadecimal style.

color coding.

- light Purple - TCP traffic
- Light Blue - UDP traffic
- Black - Packets with errors
- customization - view or modify color rules under



Sample:

→ use sample files to practice in wireshark for via

→ save your captures with files > Save for late

filtering Packets:

→ Apply filter to focus on specific network traffic

→ use other apps to isolate traffic for analyses

Type a filter, press Enter, eg: 'dns' for DNS Packet

wire shark auto completes.

* wifi

File Edit view go capture Analyze statistics

No.	Time	source	destination	Protocol	length	Info
→ 305	5.248733	2601:1c0	DNS	90	standard	
306	5.249092	2601:1c0	DNS	90	standard	
307	5.249967	2601:1c0	DNS	118	standard	
← 308	5.270325	2601:1c0	DNS	106	standard	

use Analyze > Display filters to Pick or save filter. See the docs for more info Right click a Packet, choose follow & TCP stream to see the full conversation. use follow for other protocols too.

capturing and analysis Packets using wire shark Tool:-

To filter, view, capture, Packets, capture 100 packet from the ethernet: IEEE 802.3 LAN interface and save it.

Procedure:

(i) Select local Area connection.
(ii) go to capture → option
(iii) select stop capture automatically after 100 Packets
(iv) then click start capture
(v) save the packets

1. creat a filter to display only TCP/UDP Packets inspect the Packets and Provide the flow graph.

Procedure:-

(i) Select local area connection
(ii) go to capture → option
(iii) Select stop capture after 100 Packet
(iv) click start capture.
(v) search TCP Packet
(vi) To see flow graph click statistics → flow graph

Result:-

thus the wireshark is executed and done successfully.

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 header? Explain.

NO, ARP packets do not have a transport layer because it operates at the data link layer of the OSI model & is used to map IP addresses 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 larger responses

4) what is the port number used by HTTP protocol?

the default port number used by HTTP protocol is 80. for HTTPS, the secure version of HTTP, the default port is 443.

5) what is a broadcast IP address?

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

Implement error detection and correction using Hamming code concepts.

EXP: 6

Aim: Write a program to implement error 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 for error correction.

Create 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 output in a file called channel.

Create a

```
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 calculate redundant bits needed for error correction.
        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 detection
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 bits 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 1 = 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 1 = int (data [j-1])
if (Parity != 0):
        res += position
if res != 0
    print ("error detected at position: {res}")
    data = list (data)
# correct the error
if res <= n:
        data [res-1] = '0' if data [res-1] == '1'
    print (f"error corrected at position: {res}")
else:
    print ("error position out of range
        No correction performed.")
```

```
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)+1):
        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 = calculate_redundant_bits(m)
```

arr = pos - redundant - bits (binary - data, r)

print (f" Sender output (binary with redundant bits): {arr}

return arr

Receiver Program

def receiver (data):

r = calc - redundant - bits (len (data))

corrected - data = detect - and - correct

(data, r)

Redundant bit Removal

original - data = remove - redundant - bits

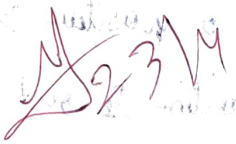
(corrected - data, r)

Print (f" decoded text: {ascii - output}")

main program:

if __name__ == "__main__":

Result: the program for hamming code is executed successfully



EX NO 7 Sliding window protocol

Aim:
Write a Program to implement flow control at data link layer using sliding window protocol simulate the flow of frames from on node to another.

Procedure:

1) The slidingwindow protocol controls the flow of data between the sender and receiver with a fixed window size n

2) the sender can send upto N frames without waiting for acknowledgement

3) After Sending, the window "slides" forward as acknowledgements are received.

4) the receiver process frames in sequence and acknowledges them.

5) Lost or corrupted frames are retransmitted, ensuring reliable and ordered delivery.

```
import time
import random
class Frame:
    def - - - init - - (self, frame-no, data):
    self.frame-no = frame-no
    self.data a = data
    self.acknowledged = False

def send-frames( frames, window-size ):
    print ("\n--sendingFrames--")
    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}[OK]")

frames[i].acknowledged = True

def sliding_window_Protocol():

window_size = int(input("enter window Size:"))

message = input("enter a message to used:")

frames = [Frame(i, message[i]) for i in range(len(message))]

base = 0

while base < len(frames):

send_frame(frames[base:], window_size)

time.sleep(2)

receive_frame(frames[base:], window_size)

while base < len(frames) and frame[base] acknowledged:

base += 1

if base < len(frames):

Print("\n Resending in acknowledged frames.")

time.sleep(2)

Print("\n frames send and unacknowledged

if __name__ == "__main__":
    slidingwindowprotocol()

output:

Enter window size: 5

Enter a message to send: HELPP

. . . . . sending Frames . . .

sent Frame 0: H
Sent Frame 1: E
Sent Frame 2: L
sent Frame 3: P
sent Frame 4: P

Frame sent, waiting for acknowledgement

. . . Receiving Frames . . . .

Received frame 0: H [received]
Received frame 1: E [Received]
Received frame 2: L [Received]
Received frame 3: P [Received]
Received frame 4: P [ERROR]
~~Received frame~~

- - - Resending Frames - - - -

Received Frame 4: A [Received]

All frames are sent and acknowledged.

Result: Thus the code for flow control (sliding control) is executed successfully.

Ex NO: 8 (a)

Date:

Virtual LAN

Aim: To simulate a Virtual LAN (VLAN) configuration using CISCO Packet Tracer, follow these step-by-step instructions

Step 1: Setup the topology

1. Open cisco packet tracer
2. Drag and Drop switches (cisco 2969) and 2 PCs into the workspace.
3) use copper straight through cables to connect the PC's to the switch using their Fast ethernet Ports.

Step 2: Assign IP addresses to PC's

1. click on each PC, go to the desktop tab and click IP configuration
2. Assign IP addresses to each PC:

- PC_1 - IP - 192.168.1.10 Subnet mask: 255.255.255.0
- PC_2 - IP - 192.168.2.10 Subnet mask: 255.255.255.0

Step 3: VLAN configuration on the Switch:

1. click on the Switch and open the CLI (command line interface)
2. Enter the following commands to create VLANS and assign the two ports to different VLAN's.

Switch (config-if) # Switchport access vlan 20

Switch (config-if) # exit

Step 4 Verify VLAN configuration :-

1) Run the following command to confirm VLAN's are properly configured.

Switch # show VLAN brief.

Step 5 Test connectivity :

1) Go to PC1 open the command prompt and ping PC2's IP address (192.168.2.10)

2) Since PC1 and PC2 are in different VLANs, they should not be to communicate with each other (you will receive timeouts)

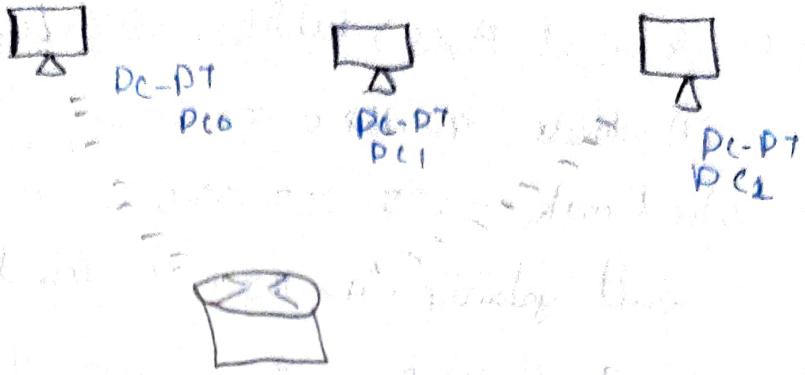
Step 6 save the configuration

Finally save the configuration using,

Switch # copy running-config start up config.

Result :-

b) Configuration of wireless LAN using a CISCO Packet Tracer



1) click network devices down click wireless devices select wireless Router in enter

2) Add 8 PC's click end device

3) click router GUI tab check given up IP address as (192.168.0.1) click disable & save setting

4) come up again click wireless, wireless settings change network save settings.

5) come up, click wireless settings, Security mode change to WEP give key 1as 0 1 2 3 4 5 6 7 8 9, Save

6) close tab

7) click PC(1) desktop, IP configuration

IP address: 192.168.0.5

Subnet mass: 255.255.255.0

default gateway: 192.168.0.1 close tab

8) Go to next PC(2) desktop, IP configuration

3) Now PC(2) again PC wireless (Repeat step 10 11 12)

4) PC(3) ⇒ (Repeat step (10, 11, 12))

5) to check the connectivity whether all is connected check any PC command prompt any IP address given during process Ping 192.168.0.6 enter.

output

VLAN	Name	status	Ports
1	default	active	Fa 0/6, Fa 0/7 Fa 0/8 Fa 0/24, Gig 0/1 Gig 0/2
10	Marketing	active	Fa 0/1, Fa 0/2
20	Sales	active	Fa 0/3, Fa 0/4
1002	fddi default	active	
	token ring default	active	

Ping 192.168.0.6

Pinging 192.168.0.6 with 32 bytes of data Request timed out.

Ping statistics for 192.168.0.6

Packets: sent = 4 Received = 0 Lost = 4 (100% loss)

23/11

Result:- Tools to simulate virtual LAN configuration using CISCO packet simulator and configuration of wireless LAN using CISCO Packet Trace is done and executed successfully.

3) configure SSID:

Set your network name.

Enable SSID board cast.

4) Set wireless Security:

- Choose WPA2-PSK or WPA3 for security
- Set a password.

5) configure wireless channel:

- Set the channel to Auto or manually select a channel.
- choose the band 2.4 GHz or 5 GHz

6) Enable DHCP

- Enable DHCP if required to assign IP address.

7) Save Settings and restart the AP

8) Test connection by connecting devices to the SSID using the Password.

Note book Summary:

1) SSID : Lab - WLAN

2) Security : WPA2-Psk

3) key : Lab@12345

4) channel : Auto

5) Band : 2.4GHz or 5 GHz

6) DHCP : Enable.

Ex No: 9
date: 9/10/24

Implementation of Subnetting

Aim:
Implementation of Subnetting 'IS CISCO PACKET TRACER simulation.

Steps:

1) Create a network topology
   - Open a Cisco Packet tracer
   - click on New, Network > Generic to create a blank topology.

2) Add devices
   - add the following devices
     - 2 routers (R_1, R_2)
     - 2 Switches (S_1, S_2)
     - 10 PCs (5 for each Subnet)
   - Connect the devices
     - use the appropriate cables to connect
       - R_1 to S_1
       - R_1 to R_2
       - S_2 to R_2

3) Subnetting configuration:
   - Network address : 192.168.1.0/24
   - Subnet mask : 127 (2 routers & Subnet has addresses each)

4) IP addressing scheme:
   - Router R_1
     - Gigabit Ethernet 0/0 : 192.168.1.1
     - Gigabit Ethernet 0/1 : 192.168.

- open CLI on Router R1 and router enable
  - configure terminal
  - Interface Gigabit ethernet 0/0
  - IP address 192.168.2.1 → 255.255.255.254
- NO shut down
  - Exit

- Switch configuration
  - open CLI on Switch S1 & enter

enable
configure terminal
Interface fastethernet 0/1
Interface mode access
switch port mode access
Exit.

interface fact ethernet 0/2
Switch port mode access
exit.

- Pc configuration
  - Right click on each Pc and select config

Enter:
- IP address, subnet mask (255.255.255.254) default gateway (Router CID)

b) Testing the Networks.
- open the command prompt on each PC
- List the ping command to check connectivity between PC's and the router Ping 192.168.1.x (for PC's in the first subnet) Ping 192.168.3.x

(for PC's in the second subnet)

*) conclusion:

If all things are successfull your subnetting and networks configuration in CISCO packet tracer is functioning correctly.

Student observation:

Write down your understanding of subnetting

Subnetting is the pratice of diveding a larger network into smaller, manageable sub-networks to improve performance and security. It uses a subnet mask to define the network and host portions of an IP address.

what is the advantage of implementing subnetting within a network?

1) Improved performance:

EX.NO:10 Routing in CISCO Packet Tracer simulator

date

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 transferred from PC0 to PC1.

Procedure:

step-1 (configuring Router):

1. Select the router and open CLI
2. Press ENTER to start configuring Router1.
3. Type enable to activate the privileged mode.

step-2 (configuring PC1):

1. Assign IP addresses to every PC in the network
2. select the PC.G10 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
4. Assign the default gateway of PC1 as 192.168.20.1

step-3 (connecting PC1 with Router):

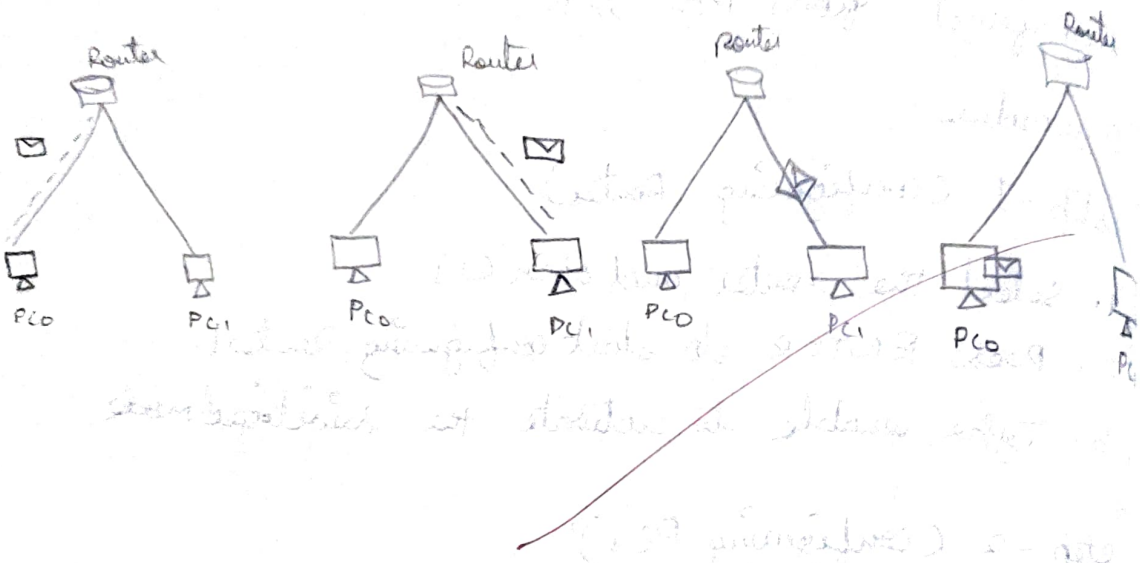
1. Fast Ethernet 0/0 Port of Router1
2. Fast Ethernet 0/1 Port of Router1

Router configuration Table:

Device Name	IP address Fast Ethernet 0/0	Subnet Mask	IP address Fast Ethernet 0/1
Router1	192.168.10.1	255.255.255.0	192.168.20.1

Pc configuration table:

device Name	IP address	subnet mask	Gateway
Pc0	192.168.10.2	255.255.255.0	192.168.10.1
Pc1	192.168.10.2	255.255.255.0	192.168.10.1



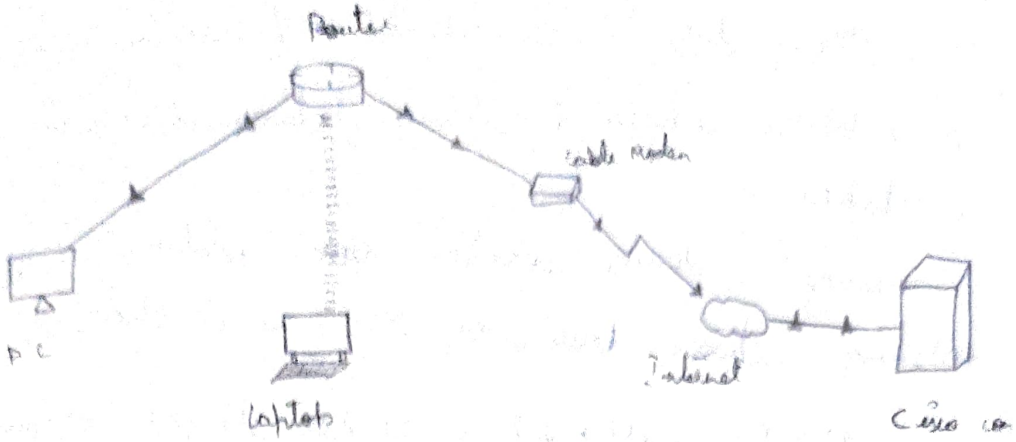
Result:

Have the CISCO Packet tracer simulator created successfully.

Wireless router DHCP router

Aim:

To Design and configure an internetwork using wireless router, DHCP server 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 Server	Internet	DHCP		
cisco.com Server	Ethernet 0	208.67.220.220	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, first choose a device type from the

b. change display names of the network devices to fit workspace

To change the display name of the network devices on the Packet Tracer logical work space, we click the config tab in 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 work space.

the Pc will need a copper straight - through cable to connect to the wireless router.

Part 1: configure the Network Devices

step-1: configure the wireless router

a. Locate the wireless router network on the wireless router.

b. click on the same setting tab.

step-2: configure the laptop.

a. configure the laptop.

step -3: configure the Pc

a. configure the Pc for the wired network

step-4: configure the Internet cloud

a. Install network modules if necessary

b. Identify the form and to modify

c. Identify the type of provider.

step-5: configure the web - can server.

a) configure the [illegible]
b) configure the [illegible]
c) configure the [illegible]
d) configure the [illegible]
e) configure the [illegible]
f) configure the [illegible]

before adding

Packet transmission:

1. Write down the key factors of configuring wireless routers and DHCP server

Wireless Router configuration:

- SSID configuration: set up a unique network name (SSID) for your wireless network to allow devices to identify and connect.
- Security settings: configure network security to protect against unauthorized access.
- Password settings: set a strong password for connecting to the network
- Channel selection: choose a wireless channel that minimizes interference from other networks or devices

Frequency: 2.4GHz or 5GHz band.

- 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 devices in a network.

- Network Scalability: DHCP server make it easy to add and manage multiple devices across large networks, as essential feature in

3. Design and configuration an Inter-network in a

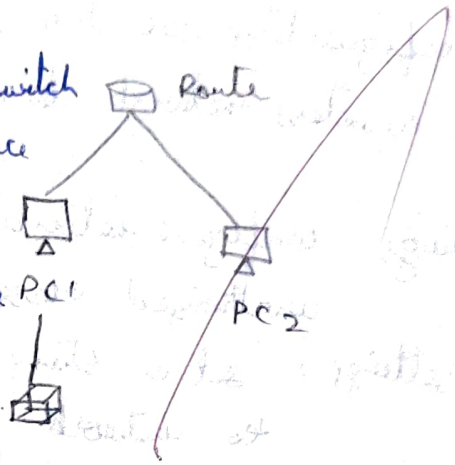
Step to design and configure an Inter-network.

1. Hardware Requirements:
   - One Switch
   - one router
   - Ethernet cables
2. Network layout:
   - Router
   - Switches
   - Devices
3. configuration

i) connect the router to switch

ii) configure the router's interface with an IP address

iii) configure the DHCP server PC1 as the router.



Result :-

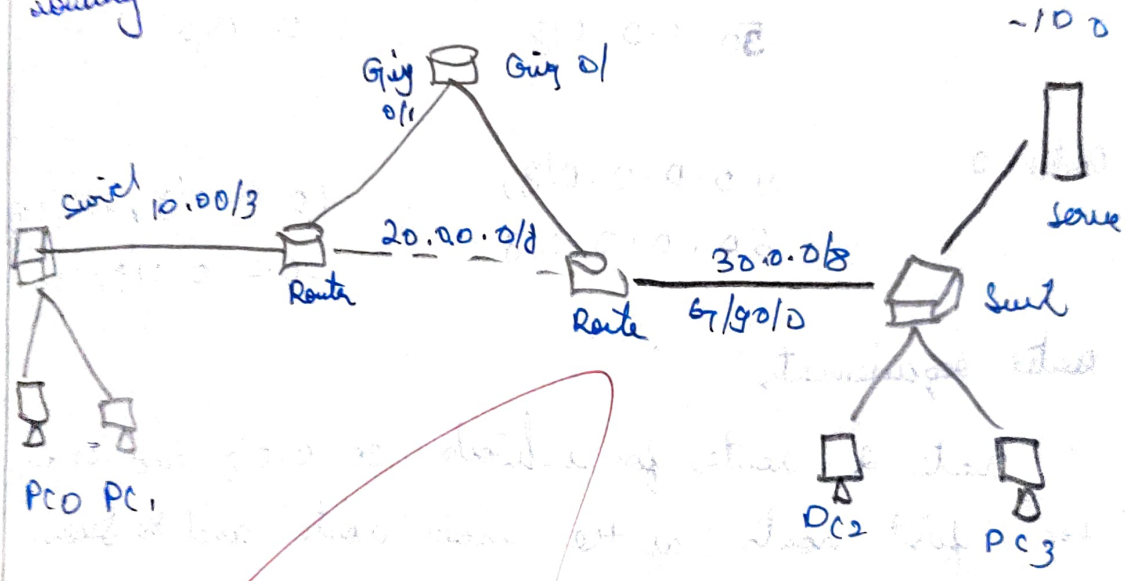
hence the router connected executed successfully

EXP.NO:11
date:

Routing at network error

Aim:
a) Simulate static Routing configuration using CISCO Packet tracker.

static routs are the routs you manually add to the router's routing table. the process of adding static routes to the routing table is known as static routing.



Creating, adding, Verifying static router

Routers automatically learn their connected networks, we only need to add router for the N etwork that are not available on the router interface.

Router	Available networks on local interface	Network available on other routers interface
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

Router requirements

- create a route for network 30.0.0/8 and configure the first router as the main route and the second route as a backup router.

EXP.NO: 11b
4/11/2024

Simulate RIP using CISCO Packet tracer

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 Fa0
Router 0	Fa0/1	10.0.0.1/8	PC0's Fast Ethernet
Router 0	Se0/0/1	192.168.1.254/30	Router 2's Se0/0/0
Router 0	Se0/0/0	192.168.1.249/30	Router 1's Se0/0/1
Router 1	Se0/0/1	192.168.1.246/30	Router 2's Se0/0/1
Router 1	Se0/0/0	192.168.250/30	Router 1's Se0/0
Router 2	Se0/0/1	192.168.245/30	Router 0's Se0/0
Router 2	Fa0/1	192.168.253/30	Router 1's Se0/0
Router 2	Fa0/1	192.168.1/30	Router 0's Se0/0
PC1	Fast Ethernet	20.0.0.1/30	PC1's Fast Ethernet
		20.0.0.2/30	Router's Fa0/1

Assign IP address to PCs

Double click PCs and click desktop menu item and click IP configuration. assign IP address referring the above table.

Assign the ip address for all the interface of router.

- Set the clock rate for DCE and not for the DTE end

show controller interface give whether the interface is DCE or DTE

Configure RIP routing Protocol.

Router 0:

router rip
network 10.0.0.0
network 192.168.1.252
network 192.168.1.244

Router 1:

router rip
network 192.168.1.244
network 192.168.1.248

Router 2:

router rip
network 10.0.0.0
network 192.168.1.252
network 192.168.1.244

Access the command Prompt of PC 1 and use Ping command to test the connectivity from PC0

Result:

Thus the Simulation RIP using CISCO Packet Tracer has been implemented successfully.

EXP.NO:12 End to End communication at transport layer

date:-

Aim:- Implement echo client server using TCP / UDP sockets

TCP echo client - server algorithm.

Server:

1. create a TCP Socket
2. Connect the Socket to local address and host.
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 the client.
   - else break the loop.
6. close connection.

client:

1. create a TCP socket
2. connect to the server using specified address and Port
3. Send a message to server.
4. Receive the echo message from the server.
5. Display the received message
6. Close 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")
connect, 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()
```

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 message to send : Hi, I am Sanath raj

Received from server : Hi, I am Sanath raj.

> Python tcp - server.py

TCP server is waiting for a connection

connected to ('127.0.0.1', 56873)

Received : Hi, I am Sanath raj.

23/1

Result :-

Thus, the program to implement echo client server using TCP is executed sucessfully.

5) Aim:

To implement the chat client server using TCP/UDP server.

Algorithm:

chat - Server:

1. Start the server by creating a socket, bind to a specific address and Port, listen for incoming connections.
2. When a new client connects add client to a list of a connected clients start a new process to talk to the clients.
3. For each connected clients start a new step checking for new messages.
4. If a client disconnects remove that client from the list & step 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 by creating a to listen to messages
3. keep asking for the new message.
4. keep running till the user decides to quit.

chat - client.py.

```
import client.py
import threading
def receive_message (client_socket):
    while true:
        try: message = client_socket, recv (1024),
                       decode ("utf")
        if message:
            Print (f"Server: {message}");
```

except 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_socket), do_error = True).
                                  start()

while True:
    message = input("yes:")
    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).
            if not message:
                break
            Print(f"Received message from client: {message}")

```
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, args=(client_socket,))
    client_handler.start()
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: Basith
Type from message to client: Received
> Python chat_client.py
connected to chat server.
you: client
you: server: Received
```

Result:

Thus the program to implement the client server using TCP is executed successfully.

EXP.NO:13 Implement your own Ping Program

date :

Aim: Implement your own ping program

Algorithm:

- Open a raw socket to send ICMP request
- 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.

server Script.Py:

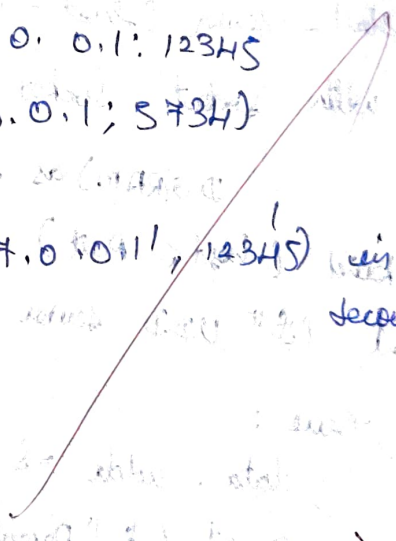
```
import socket
def start_server(host='127.0.0.1', Port=12345):
    with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as S:
        S.bind((host, Port))
        Print(f"UDP server running on {host}:{Port}")
while True:
    data, addr = S.recvfrom(1024)
    Print(f"Received message from {addr}:
    {data.decode()}")
    S.sendto(b'Pong', addr)
if __name__ == "__main__":
    start_server()
```

Client. Script. Py.

```
def start_server(host='127.0.0.1', port=12345):
    with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
        try:
            s.bind((host, port))
            print(f"UDP server running on {host}:{port}")
            while True:
                data, addr = s.recvfrom(1024)
                print(f"UDP server running at {host}:{port}")
                s.sendto(b'Pong', addr)
        except socket.timeout:
            print("Request timed out")
```

Output:-

```
> Python Ping_server.py
UDP server running on 127.0.0.1:12345
Received message from (127.0.0.1, 57324)
> Python Ping_client
Received Ping from ('127.0.0.1', 12345) in 0.000 seconds
```



Result:

Thus the Program to implement Ping Program is executed successfully.

EX.NO: 14
date :

write a code using RAW Socket

Aim: write a code using RAW Socket to implement Packet Sniffing.

Algorithm:

- create a raw Socket
- Continously capture incoming Packets using raw Socket
- Parse and display information like the Source and destinated IP address and Protocol type.
- close the Socket after finishing the capturing Process

code:

```
from scapy.all import sniff
from scapy.layers.inet import IP, TCP, UDP, ICMP
def Packet_call back (Packet):
    if IP in Packet:
        ip_layer = Packet[IP]
        Protocol = IP_layer.Protocol
        src_ip = ip_layer.src
        dest_ip = ip_layer.dst
        Protocol_name = ""
        if P_protocol == 1:
            P protocol_name = "ICMP"
        elseif Protocol == 6:
            Protocol_name = 'TCP'
else if
    Protocol_name = "UDP"
```

else:
    Protocol_name = "unknown Protocol"
  print(f"Protocol : {Protocol : {Protocol_name}}")
  Print (f" Source IP : {src_ip}")
   Print (f" Destination IP : {dest_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 : 193.184.216.34

Protocol : TCP
Source IP : 192.168.1.2
Destination IP : 172.217.14.206

Protocol : TCP
Source IP : 192.168.1.2

23/11

Result:-
thus the code using RAW sockets to implement packet sniffing is executed successfully.

EX.NO: 15

date:

Analyse various types of server using weblaliza tool.

Aim: To analyse the different types of web logs using weblaliza tool

Procedure:

S1: Run webalizer windows version

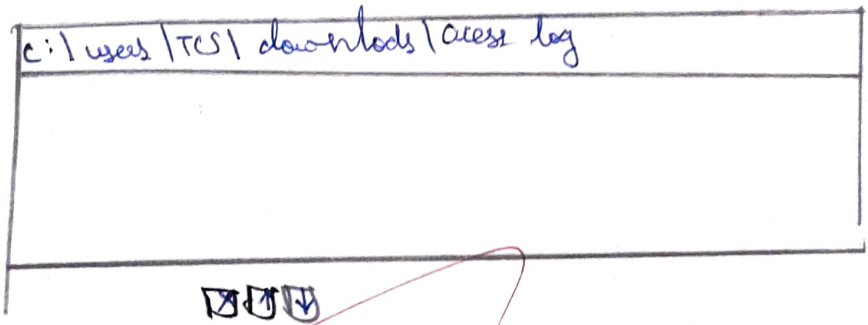
S2: Input web log file (down load from)

S3: Press run webalizer

choose	log file	view	additional HTML code	HTML

Input:

log file:

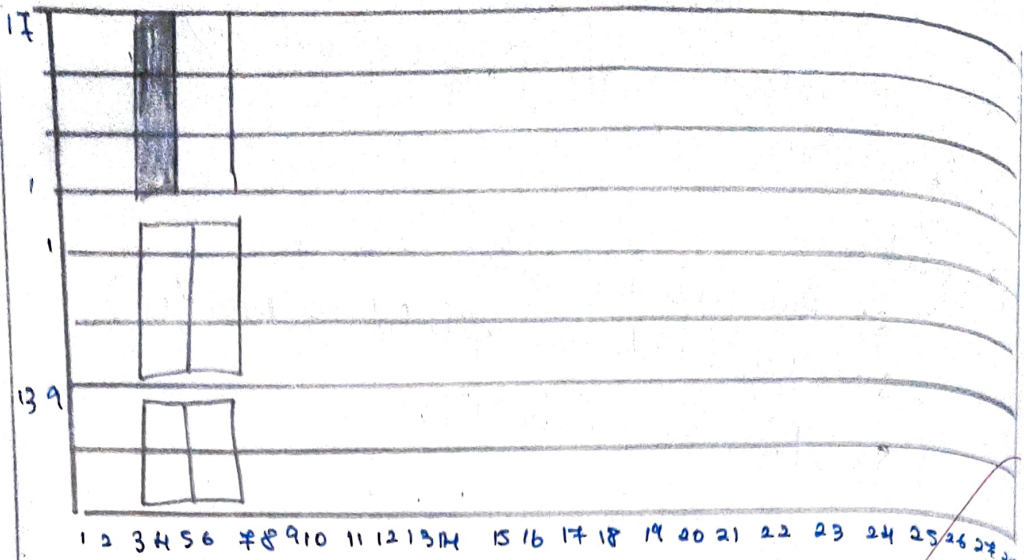


Target Directory

clear existing directory

delete all files in selected targeted directory.

daily usage for November 2024



Day	Hits	Files	Pages visits	sites	k bytes
3	67.100.1.	65100.00 -/.	1 100.00%	1100.00%.	139100.00%

23/11

Result:-

Thus, the procedure to analyse the different types of web logs using weblalizer tool is executed sucessfully.