VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

COMPUTER NETWORKS

Submitted by

DHANUSH S(1BM22CS086)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019

Sep 2024-Jan 2025

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**COMPUTER NETWORKS**" carried out by **DHANUSH S(1BM22CS086)** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Computer Networks Lab - (23CS5PCCON)** work prescribed for the said degree.

Dr. Nandhini Vineeth
Associate Professor,
Department of CSE,
BMSCE, Bengaluru

Dr. Kavitha Sooda
Professor and Head,
Department of CSE
BMSCE, Bengaluru

INDEX

Sl. No.	Date	Experiment Title	Page No.
1	09-10-24	Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping messages.	1-9
2	09-10-24	Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.	9-18
3	16-10-24	Configure default route, static route to the Router .	19-25
4	13-11-24	Configure DHCP within a LAN and outside LAN.	21-29
5	20-11-24	Configure RIP routing Protocol in Routers .	30-33
6	20-11-24	Demonstrate the TTL/ Life of a Packet.	34-37
7	27-11-24	Configure OSPF routing protocol.	38-41
8	18-12-24	Configure Web Server, DNS within a LAN.	40-43
9	18-12-24	To construct a simple LAN and understand the concept and operation of Address Resolution Protocol (ARP).	44-46
10	18-12-24	To understand the operation of TELNET by accessing the router in the server room from a PC in the IT office.	47-49
11	18-12-24	To construct a VLAN and make the PC's communicate among a VLAN.	50-54
12	18-12-24	To construct a WLAN and make the nodes communicate wirelessly.	55-57

INDEX

CYCLE 2

Sl. No.	Date	Experiment Title	Page No.
1	18-12-24	Write a program for error detecting code using CRC-CCITT (16-bits).	58-62
2	18-12-24	Write a program for congestion control using Leaky bucket algorithm.	63-64
3	18-12-24	Using TCP/IP sockets, write a client-server program to make the client send the file name and the server to send back the contents of the requested file if present.	65-68
4	18-12-24	Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.	69-72
5	18-12-24	Wireshark	73

LABORATORY PROGRAM – 1

Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message.

Date 9.10.24

Routers

Pre explaination
→ Generic Router : 2 IP's
→ Place PC's – Different IP's network ids
→ Manual Router Config

Commands
enable
#Config terminal
interface fast ethernet 0/0 : 1|1
ip address 10.0.0.1 255.0.0.0
no shutdown
exit

Aim:- To Configure IP address to routers in packet tracer. To demonstrate and understand the working of routers in connecting devices of two different networks

Procedure:-
1. Add two PC's and one generic router Configure end devices : 10.0.0.10 and 20.0.0.10 and mention/define gateway 10.0.0.1 and 20.0.0.1
2. Connect the PC's to the router via Copper cross over
click on Router → CLI → Manual Configuration
Commands:-
Router > enable
Router # Config terminal
Router (config) # interface fastethernet 0/0
Router (Config-if) # ip address 10.0.0.1 255.0.0.0

Ping Statistics for 20.0.0.10 :
Packets: Sent: 4, Recieved = 4, Lost = 0 (0% loss)
Approx round trip times in milli seconds:
Minimum = 4ms, Maximum = 4ms, Average = 4ms.

C 10.0.0.0/8 is directly connected, fastethernet 0/0
C 20.0.0.0/8 is directly connected, fastethernet 1/0

10.0.0.5
10.0.0.6
2960 IOS15
Switch
PC
10.0.0.1
PC
10.0.0.2
PC
10.0.0.3

Screenshot:

Hub-PT
HUB
PC-PT
10.0.0.1
PC-PT
10.0.0.2
PC-PT
10.0.0.3

10.0.0.1
Physical
Config
Desktop
Programming
Attributes
Command Prompt
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.3
Pinging 10.0.0.3 with 32 bytes of data:
Reply from 10.0.0.3: bytes=32 time=20ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Ping statistics for 10.0.0.3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 20ms, Average = 5ms

METRO

Page:

Date: 25/9/24

Lab2

Assignment Problem:-

Ping 10.0.0.10 from 10.0.0.2

1. Data Packet(DPU) → 10.0.02 → Hub1
2. Hub1 broadcasted the DPU to the Rest 4 connections [3 PC's and Hub4] → 'X' mark
   - Hub1 also sent DPU to Switch
3. Then Hub4 broadcasted DPU to PC's 10.0.0.5 & 10.0.0.6 → 'X' mark
4. Switch proceed DPU and sent it to Hub 3 to which 10.0.0.10 was connected [destination]
5. Hub 3 broadcasted DPU to 10.0.0.10 & 10.0.0.11 → 'X' mark for 10.0.0.11 & 10.0.0.10 was the right destination.
6. 10.0.0.10 sent the DPU back to Hub 3, Hub3 set DPU to Switch and 10.0.0.11 → 'X'
7. Switch proceeded it & sent the DPU to Hub1 to which the Source is connected (10.0.0.2)
8. Hub broadcasted DPU to all the 5 Connections.
9. 10.0.0.2 displayed '✓' mark, indicating the right device, Others showed 'X' mark.
10. Hub 4 broadcasted DPU to 10.0.0.5 & 10.0.0.6, but again it should 'X' mark

Date 9/10/24

Routers

Pre explaination
→ Generic Router : 2 IP's
→ Place PC's - Different IP's network ids
→ Manual Router Config

Commands
enable
#Config terminal
interface fast ethernet 0/0 : 1/1
ip address 10.0.0.1 255.0.0.0
no shutdown
exit

Aim:- To Configure IP address to routers in packet tracer. To demonstrate and understand the working of routers in Connecting devices of two different networks.

Procedure:-
1. Add two PC's and one generic router Configure end devices : 10.0.0.10 and 20.0.0.10 and mention/define gateway 10.0.0.1 and 20.0.0.1
2. Connect the PC's to the router via Copper cross over
click on Router → CLI → Manual Configuration
Commands:-
Router > enable
Router # Config terminal
Router (config) # interface fastethernet 0/0
Router (config-if) # ip address 10.0.0.1 255.0.0.0

Router(config-if) # no shutdown
Router(config-if) # exit

Repeat for other PC: fastethernet 1/0

(5) After Successfull Configuration, the connection turns green.

(6) click on PC 10.0.0.10 → Desktop → Command prompt

(7) ping 20.0.0.10 → To Send data packet to other device from the other networks.

Topology :-

Observations:-

(1) Data packet was Sent from 10.0.0.10 to router

(2) The router Sent the packet to PC 20.0.0.10

(3) Data packet bash to router → back to PC 10.0.0.10 and a tick mark is blinked

Reply from 20.0.0.10: bytes=32 time=4ms TTL=127

IP route was observed as:

Router # show ip route

2960-24TT
Switch
PC-PT
10.0.0.1
PC-PT
10.0.0.2
PC-PT
10.0.0.3

10.0.0.1
Physical
Config
Desktop
Programming
Attributes
Command Prompt
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.3
Pinging 10.0.0.3 with 32 bytes of data:
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Ping statistics for 10.0.0.3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms

LABORATORY PROGRAM – 2

Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.

Date 9.10.24

Routers

Pre explaination
→ Generic Router : 2 IP's
→ Place PC's – Different IP's network ids
→ Manual Router Config

Commands
enable
Config terminal
interface fast ethernet 0/0 : 1/1
ip address 10.0.0.1 255.0.0.0
no shutdown
exit

Aim:- To Configure IP address to routers in packet tracer. To demonstrate and understand the working of routers in Connecting devices of two different networks.

Procedure:-
① Add two PC's and one generic router Configure end devices : 10.0.0.10 and 20.0.0.10 and mention/define gateway 10.0.0.1 and 20.0.0.1
② Connect the PC's to the router via Copper cross over
click on Router → CLI → Manual Configuration
Commands:-
Router > enable
Router # Config terminal
Router (config) # interface fastethernet 0/0
Router (config-if) # ip address 10.0.0.1 255.0.0.0

Router(config-if) # no shutdown
Router(config-if) # exit

Repeat for other PC: fastethernet 1/0

(5) After Successfull Configuration, the connection turns green.

(6) click on PC 10.0.0.10 → Desktop → Command prompt

(7) ping 20.0.0.10 → To Send data packet to other device from the other networks.

Topology :-

Observations:-

(1) Data packet was Sent from 10.0.0.10 to router

(2) The router Sent the packet to PC 20.0.0.10

(3) Data packet bash to router → back to PC 10.0.0.10 and a tick mark is blinked

Reply from 20.0.0.10: bytes=32 time=4ms TTL=127

IP route was observed as:

Router # show ip route

Ping Statistics for 20.0.0.10 :
Packets: Sent: 4, Recieved = 4, Lost = 0 (0% loss)
Approx round trip times in milli seconds:
Minimum = 4ms, Maximum = 4ms, Average = 4ms.

C 10.0.0.0/8 is directly connected, fastethernet 0/0
C 20.0.0.0/8 is directly connected, fastethernet 1/0

Screenshots:

```
2 Low-speed serial(sync/async) network interface(s)
32K bytes of non-volatile configuration memory.
63488K bytes of ATA CompactFlash (Read/Write)


         --- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: no


Press RETURN to get started!



Router>enable
Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface fastethernet0/0
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#no shutdown

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

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up

Router(config-if)#exit
Router(config)#interface fastethernet1/0
Router(config-if)#ip address 20.0.0.1 255.0.0.0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet1/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to up

Router(config-if)#exit
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console

Router#
```

```
C:\>ping 20.0.0.10

Pinging 20.0.0.10 with 32 bytes of data:

Reply from 20.0.0.10: bytes=32 time<1ms TTL=127
Reply from 20.0.0.10: bytes=32 time<1ms TTL=127
Reply from 20.0.0.10: bytes=32 time<1ms TTL=127
Reply from 20.0.0.10: bytes=32 time<1ms TTL=127

Ping statistics for 20.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

20.0.0.10

Physical | Config | Desktop | Programming | Attributes

Command Prompt

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.10

Pinging 10.0.0.10 with 32 bytes of data:

Reply from 10.0.0.10: bytes=32 time<1ms TTL=127
Reply from 10.0.0.10: bytes=32 time<1ms TTL=127
Reply from 10.0.0.10: bytes=32 time<1ms TTL=127
Reply from 10.0.0.10: bytes=32 time<1ms TTL=127

Ping statistics for 10.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

16/10/24

Lab No-3 (Experiment 2)

2 Routers

Aim: To configure ip address to routers in packet Tracer. Explore following messages: ping responses, destination unreachable, request timed out, reply

Topology:

Def gateway 10.0.0.2

Def gateway 20.0.0.2

Procedure:-

1. Add two PCs and two given Routers.
2. Configure end devices 10.0.0.10 & 20.0.0.10 and define gateways
3. Connect PC to Router Using copper cross over connect the routers to each other using Serial DCE
4. Configure the routers:
   click on Router → CLI
   → Configure to end devices similar to last experiment.

→ Connecting the routers:

Commands:

Router>enable

Router # Config terminal

Router(config) # interface Serial 2/0

Router(config-if) # ip address 30.0.0.1

Router(config-if) # no shut

5. Give appropriate naming
6. Ping the device 20.0.0.1 from 10.0.0.1

OBSERVATIONS:-

1. The connections were done properly and green lights were displayed.
2. But when the device 10.0.0.1 pinged 20.0.0.1 the output showed that the host was unreachable
3. Since the networks are not direct neighbours directly connected, the data sent was failed.
4. Pinging the same networks was a Sucess since they can directly connected, pinging 30.0.0.1 for 10.0.0.1 was a Success.

Screenshots:

10.0.0.1
30.0.0.1
30.0.0.2
20.0.0.1
Router1
PC-PT
10.0.0.10
Def Gateway 10.0.0.1
PC-PT
20.0.0.10
Def Gateway 20.0.0.1
Router1
Physical
Config
CLI
IOS Command Line Interface
Router>enable
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
* - candidate default, U - per-user static route, o - ODR
P - periodic downloaded static route
Gateway of last resort is not set
C 10.0.0.0/8 is directly connected, FastEthernet0/0
C 30.0.0.0/8 is directly connected, Serial2/0
Router#
Copy
Paste

LABORATORY PROGRAM – 3

Configure default route, static route to the Router.

(5) Default Routing

Router R→0 (Config)# ip route 0.0.0.0 0.0.0.0 30.0.0.1

Router R→2 (Config) # ip route 0.0.0.0 0.0.0.0 40.0.0.2

METRIC

Date: / /

Router(Config-if) # ip address 10.0.0.2 255.0.0.0
Router(Config-if) # no shutdown
Router(Config-if) # exit

Repeat for other PC: FastEthernet 1/0

(6) After Successfull Configuration, the Connection turns green

(7) Click on PC 10.0.0.10 → Desktop → command prompt

(8) Connecting the Routers Using the Commands

Router > enable
Router # Config terminal
Router (config) # interface Serial 2/0
Router (config-if) # IP address 30.0.0.1
Router (Config-if) # no Shut

(9) Give appropriate naming for the 3 Routers

(10) ping 20.0.0.10 → to Send data packets to other devices from the other networks.

Observations :-

(1) The Connections were done properly and green lights were displayed.

(2) IP route before setup:

C 30.0.0.0/8 is directly Connected, Serial 2/0
C 40.0.0.0/8 is directly connected, Serial 3/0

(3) IP Route after setup:-

S 10.0.0.0/8 [1/0] via 30.0.0.1
S 20.0.0.0/8 [1/0] via 40.0.0.2
C 30.0.0.0/8 is directly Connected, Serial 2/0
C 40.0.0.0/8 is directly Connected, Serial 3/0

(4) Ping from one pc to another PC is Successfull

(5) So the middle router (router 1) is Setup 2 next hops.

(6) Default Router : To Transfer when no other router is available

(7) Static Route = Define Route with assigned destination.

Ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

8/8

23/10

Reply from 20.0.0.1 : bytes = 32 time = 9ms TTL = 125

Reply from 20.0.0.1 : bytes = 32 time = 6ms TTL = 125

Reply from 20.0.0.1 : bytes = 32 time = 7ms TTL = 125

Reply from 20.0.0.1 : bytes = 32 time = 7ms TTL = 125

Ping Statistics for 20.0.0.1 :

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

Approximate round trip times in milli-Seconds

Minimum = 6ms, Maximum = 9ms, Average = 7ms

Screenshots:

10.0.0.1
20.0.0.1
Se2/0
Fa0/0
Router-PT
Router0
20.0.0.2
Se2/0
30.0.0.1
Se3/0
Router-PT
Router1
30.0.0.2
Se2/0
40.0.0.1
Fa0/0
Fa0
PC-PT
10.0.0.10
Gateway 10.0.0.1
Fa0
PC-PT
40.0.0.10
Gateway 40.0.0.1
Router0
Physical
Config
CLI
IOS Command Line Interface
Router(config-if)#
%LINK-5-CHANGED: Interface Serial2/0, changed state to up
%SYS-5-CONFIG_I: Configured from console by console
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up
Router(config-if)$exit
Router(config)$exit
Router$show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
* - candidate default, U - per-user static route, o - ODR
P - periodic downloaded static route
Gateway of last resort is 20.0.0.2 to network 0.0.0.0
C 10.0.0.0/8 is directly connected, FastEthernet0/0
C 20.0.0.0/8 is directly connected, Serial2/0
S* 0.0.0.0/0 [1/0] via 20.0.0.2
Router#
Copy
Paste

10.0.0.1
20.0.0.1
Se2/0
Fa0/0
Router-PT
Router0
20.0.0.2
Se2/0
30.0.0.1
Se3/0
Router-PT
Router1
30.0.0.2
Se2/0
40.0.0.1
Fa0/0
Fa0
PC-PT
10.0.0.10
Gateway 10.0.0.1
Fa0
PC-PT
40.0.0.10
Gateway 40.0.0.1
Router1
Physical
Config
CLI
IOS Command Line Interface
%SYS-5-CONFIG_I: Configured from console by console
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to up
Router(config-if)$exit
Router(config)$exit
Router$show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
* - candidate default, U - per-user static route, o - ODR
P - periodic downloaded static route
Gateway of last resort is not set
S 10.0.0.0/8 [1/0] via 20.0.0.1
C 20.0.0.0/8 is directly connected, Serial2/0
C 30.0.0.0/8 is directly connected, Serial3/0
S 40.0.0.0/8 [1/0] via 30.0.0.2
Router#
%SYS-5-CONFIG_I: Configured from console by console
Copy
Paste

10.0.0.1
20.0.0.1
Se2/0
Fa0/0
Router-PT
Router0
20.0.0.2
Se2/0
30.0.0.1
Se3/0
Router-PT
Router1
30.0.0.2
Se2/0
40.0.0.1
Fa0/0
Router2
Fa0
PC-PT
10.0.0.10
Gateway 10.0.0.1
Fa0
PC-PT
40.0.0.10
Gateway 40.0.0.1
Router2
Physical
Config
CLI
IOS Command Line Interface
Gateway of last resort is not set
C 40.0.0.0/8 is directly connected, FastEthernet0/0
Router#
%LINK-5-CHANGED: Interface Serial2/0, changed state to up
%SYS-5-CONFIG_I: Configured from console by console
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
* - candidate default, U - per-user static route, o - ODR
P - periodic downloaded static route
Gateway of last resort is 30.0.0.1 to network 0.0.0.0
C 30.0.0.0/8 is directly connected, Serial2/0
C 40.0.0.0/8 is directly connected, FastEthernet0/0
S* 0.0.0.0/0 [1/0] via 30.0.0.1
Router#
Copy
Paste

10.0.0.1
20.0.0.1
Se2/0
Fa0/0
Router-PT
Router0
20.0.0.2
Se2/0
30.0.0.1
Se3/0
Router-PT
Router1
30.0.0.2
Se2/0
40.0.0.1
Fa0/0
Fa0
PC-PT
10.0.0.10
Gateway 10.0.0.1
Fa0
PC-PT
40.0.0.10
Gateway 40.0.0.1
10.0.0.10
Physical
Config
Desktop
Custom Interface
Command Prompt
Pinging 30.0.0.1 with 32 bytes of data:
Reply from 30.0.0.1: bytes=32 time=4ms TTL=254
Reply from 30.0.0.1: bytes=32 time=4ms TTL=254
Reply from 30.0.0.1: bytes=32 time=4ms TTL=254
Reply from 30.0.0.1: bytes=32 time=4ms TTL=254
Ping statistics for 30.0.0.1:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 4ms, Maximum = 4ms, Average = 4ms
PC>ping 30.0.0.2
Pinging 30.0.0.2 with 32 bytes of data:
Reply from 30.0.0.2: bytes=32 time=8ms TTL=253
Reply from 30.0.0.2: bytes=32 time=6ms TTL=253
Reply from 30.0.0.2: bytes=32 time=5ms TTL=253
Reply from 30.0.0.2: bytes=32 time=7ms TTL=253
Ping statistics for 30.0.0.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 5ms, Maximum = 8ms, Average = 6ms
PC>

10.0.0.1
20.0.0.1
Se2/0
Fa0/0
Router-PT
Router0
20.0.0.2
Se2/0
30.0.0.1
Se3/0
Router-PT
Router1
30.0.0.2
Se2/0
40.0.0.1
Fa0/0
Fa0
PC-PT
10.0.0.10
Gateway 10.0.0.1
Fa0
PC-PT
40.0.0.10
Gateway 40.0.0.1
10.0.0.10
Physical
Config
Desktop
Custom Interface
Command Prompt
Pinging 30.0.0.2 with 32 bytes of data:
Reply from 30.0.0.2: bytes=32 time=8ms TTL=253
Reply from 30.0.0.2: bytes=32 time=6ms TTL=253
Reply from 30.0.0.2: bytes=32 time=5ms TTL=253
Reply from 30.0.0.2: bytes=32 time=7ms TTL=253
Ping statistics for 30.0.0.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 5ms, Maximum = 8ms, Average = 6ms
PC>ping 40.0.0.1
Pinging 40.0.0.1 with 32 bytes of data:
Reply from 40.0.0.1: bytes=32 time=9ms TTL=253
Reply from 40.0.0.1: bytes=32 time=7ms TTL=253
Reply from 40.0.0.1: bytes=32 time=9ms TTL=253
Reply from 40.0.0.1: bytes=32 time=8ms TTL=253
Ping statistics for 40.0.0.1:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 7ms, Maximum = 9ms, Average = 8ms
PC>

10.0.0.1
20.0.0.1
Se2/0
Fa0/0
Router-PT
Router0
20.0.0.2
Se2/0
30.0.0.1
Se3/0
Router-PT
Router1
30.0.0.2
Se2/0
40.0.0.1
Fa0/0
Fa0
PC-PT
10.0.0.10
Gateway 10.0.0.1
Fa0
PC-PT
40.0.0.10
Gateway 40.0.0.1
10.0.0.10
Physical
Config
Desktop
Custom Interface
Command Prompt
Pinging 40.0.0.1 with 32 bytes of data:
Reply from 40.0.0.1: bytes=32 time=8ms TTL=253
Reply from 40.0.0.1: bytes=32 time=7ms TTL=253
Reply from 40.0.0.1: bytes=32 time=9ms TTL=253
Reply from 40.0.0.1: bytes=32 time=8ms TTL=253
Ping statistics for 40.0.0.1:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 7ms, Maximum = 9ms, Average = 8ms
PC>ping 20.0.0.2
Pinging 20.0.0.2 with 32 bytes of data:
Reply from 20.0.0.2: bytes=32 time=4ms TTL=254
Reply from 20.0.0.2: bytes=32 time=4ms TTL=254
Reply from 20.0.0.2: bytes=32 time=3ms TTL=254
Reply from 20.0.0.2: bytes=32 time=4ms TTL=254
Ping statistics for 20.0.0.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 3ms, Maximum = 4ms, Average = 3ms
PC>

LABORATORY PROGRAM – 4

Configure DHCP within a LAN and outside LAN

Procedure :-

1. Place 3 Routers (Generic), 3 Generic Switch to 6 PCs.
2. Connect the Routers to the Corresponding Switches, Then Connect 2PCs to one Switch. Use Automatically choose Connection Type.
3. Configure the end devices and define gateways
4. Configure the Routers using CLI and check for Green lights for all Connections

for Router0 :-

```
Router > enable
Router # configterminal
Router(config-router) # network 10.0.0.0
Router(config-router) # network 40.0.0.0
Router(config-router) # exit
```

Observations:-

① IP address were allocated dynamically

② When we ping from one PC to another data was sent successfully

10.0.0.2 ③ PC> ping 10.0.0.4

Pinging 10.0.0.4 with 32 bytes of data

Reply from 10.0.0.4 : bytes=32 time=0ms TTL=128
Reply from 10.0.0.4 : bytes=32 time=0ms TTL=128
Reply from 10.0.0.4 : bytes=32 time=0ms TTL=128
Reply from 10.0.0.4 : bytes=32 time=0ms TTL=128

ping Statistics for 10.0.0.4:
Packets: Sent=4, Recieved=4, Lost=0
Approximate round trip times in ms
minimum Outside LAN

Procedure (2):-

① Dynamically enable ip address for end Devices

CLI Commands

Router>enable
Router#Configterminal
Router(Config-if) #interface fast ethernet
Router(config-if) #ipaddress 10.0.0.1 255.0.0.
Router(config-if) #ip-helper address 10.0.0.2

10.0.0.1
dg-10.0.0.0
Fa0
Server-PT
Server0
DHCP server
Fa3/1
Fa0/1
Fa1/1
Fa2/1
Fa0
PC-PT
PC0
Fa0
PC-PT
PC1
Fa0
PC-PT
PC2

Router0
10.0.0.2
dg1-10.0.0.1
dg2=20.0.0.1
Server-PT
Server0
DHCP server
PC-PT
PC0
10.0.0.5
PC-PT
PC1
10.0.0.6
PC-PT
PC2
10.0.0.3
PC-PT
PC3
20.0.0.3
PC-PT
PC4
20.0.0.4
PC-PT
PC5
20.0.0.5

PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.4
Pinging 10.0.0.4 with 32 bytes of data:
Reply from 10.0.0.4: bytes=32 time=1ms TTL=128
Reply from 10.0.0.4: bytes=32 time=0ms TTL=128
Reply from 10.0.0.4: bytes=32 time=0ms TTL=128
Reply from 10.0.0.4: bytes=32 time=0ms TTL=128
Ping statistics for 10.0.0.4:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 1ms, Average = 0ms
PC>

LABORATORY PROGRAM – 5

Configure RIP routing Protocol in Routers

Lab No. 6

(Experiment 5)

20/11/2024

Problem Statement:- Configure Routing information protocol in Routers (RIP)

Procedure:-

1. Place 3 Routers (Generic), 3 Generic Switch to 6 PCs.
2. Connect the Routers to the Corresponding Switches, Then Connect 2PCs to one Switch Use Automatically choose Connection Type.
3. Configure the end devices and define gateways
4. Configure the Routers using CLI and check for Green lights for all Connections

for Router0:-

```
Router > enable
Router # configterminal
Router(config-router) # network 10.0.0.0
Router(Config-router) # network 40.0.0.0
Router(Config-router) # exit
```

Router1:-
Router>enable
Router#config terminal
Router(Config-router)#network 40.0.0.0
Router(Config-router)#network 50.0.0.0
Router(Config-router)#network 20.0.0.0
Router(Config-router)#exit.

Router2:
Router>enable
Router#Config terminal
Router(Config-router)#network 50.0.0.0
Router(Config-router)#network 30.0.0.0
Router(Config-router)#exit.

Q Demonstrate the TTL/Life of a packet.

→ Observation:
① Before Routing information Protocol:-
In Router 2:-
(Show IP route)
C 30.0.0.0/8 is directly connected
C 50.0.0.0/8 is directly connected
R 10.0.0.0/8 [120/2] via 50.0.0.1

Procedure:-
① Demonstrate TTL using Same Topology
② Send a Simple PDU for end Device 10.0.0.2 to 30.0.0.3 in Simulation Mode
③ Create the Topology as Shown in figure ~~before~~ and also Configure the Routers before Demonstrating using TTL

Observation:-

① Router0:
Inbound Details: TTL: 255
Outbound Details: TTL: 254

② Router1
Inbound Details: TTL: 254
OutBound Details: TTL: 253

③ Router 2
Inbound Details: TTL: 253
OutBound Details: TTL: 252

TTL reduces after passing through every router

Topology:-

40.0.0.1
Se2/0
Router0
Fa0/0
10.0.0.1
40.0.0.2
Se2/0
50.0.0.1
Se3/0
Router1
Fa0/0
20.0.0.1
50.0.0.2
Se2/0
Router2
Fa0/0
30.0.0.1
Fa0/1
Fa2/1
Fa0
PC-PT
PC0
10.0.0.2
PC-PT
PC1
10.0.0.3
PC-PT
PC2
20.0.0.2
PC-PT
PC3
20.0.0.3
PC-PT
PC4
30.0.0.2
PC-PT
PC5
30.0.0.3

PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
Pinging 30.0.0.2 with 32 bytes of data:

Request timed out.
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Reply from 30.0.0.2: bytes=32 time=6ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125

Ping statistics for 30.0.0.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 7ms, Average = 6ms

PC>ping 30.0.0.2

Pinging 30.0.0.2 with 32 bytes of data:

Reply from 30.0.0.2: bytes=32 time=4ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125
Reply from 30.0.0.2: bytes=32 time=7ms TTL=125

Ping statistics for 30.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 7ms, Average = 6ms

PC>

LABORATORY PROGRAM – 6

Configure OSPF routing protocol

Procedure:-

① Place three routers & 2 end devices, connect them as per topology & assign ip address.

② Set IP address to routers:-

Router 1:-

Router(config)# interface Serial 2/0
Router(config-if) # ip address 20.0.0.1 255.0.0.
Router(config-if) # encapsulation ppp
Router(config-if) # ~~clock~~ rate 64000

Similarly Set for other 2 routers.

③ OSPF routing:-

Router 1:
Router(config) # router ospf 1
Router(config-router) # ~~router-id~~ 1.1.1.1
Router(config-router) # network 10.0.0.0
0.255.255.255 area 3
Router(config-router) # network 20.0.0.0
0.255.255.255 area 1

Similarly Set for other 2 routers i.e 2.2.2.2 & 3.3.3.3

④ Loopback

Router 1:
Router(config) # interface loopback 0
Router(config-if) # ip address 172.16.1.252
255.255.0.0
Router(config-if) # no shutdown

Similarly do for other 2 routers
172.16.1.253
172.16.1.254

(5) Ping device 40.0.0.10 from 10.0.0.10

OBSERVATION:-

(1) Routing Table:-

for Router 8:

Router # show ip route

O IA 10.0.0.0/8 via 30.0.0.1

O IA 20.0.0.0/8 via 30.0.0.1

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

C 30.0.0.0/8 is directly connected, Serial 2/0

C 30.0.0.1/32 is directly connected, Serial 2/0

C 40.0.0.0/8 is directly connected, fa 0/0

C 172.16.0.0/16 is directly connected, loopback

- All the routers know about each other
- Thus Connection Successfully established through OSPF protocol.

(2) Pinging:-

→ Pinging was Successfull: The data packets were successfully sent from device 10.0.0.10 to 40.0.0.10 as connection was established through OSPF protocol.

Se2/0
Fa0/0
Router-PT
Router0
10.0.0.1
20.0.0.1
20.0.0.2
Se2/0
Se3/0
Router-PT
Router1
30.0.0.1
30.0.0.2
Se2/0
Router-PT
Fa0/0
40.0.0.1
Fa0
PC-PT
PC0
10.0.0.10
def GW:10.0.0.1
Fa0
PC-PT
PC1
40.0.0.10
def GW:40.0.0.1

PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Pinging 40.0.0.10 with 32 bytes of data:
Request timed out.
Reply from 40.0.0.10: bytes=32 time=7ms TTL=125
Reply from 40.0.0.10: bytes=32 time=7ms TTL=125
Reply from 40.0.0.10: bytes=32 time=8ms TTL=125
Ping statistics for 40.0.0.10:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 7ms, Maximum = 8ms, Average = 7ms
PC>ping 40.0.0.10
Pinging 40.0.0.10 with 32 bytes of data:
Reply from 40.0.0.10: bytes=32 time=9ms TTL=125
Reply from 40.0.0.10: bytes=32 time=7ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Ping statistics for 40.0.0.10:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 6ms, Maximum = 9ms, Average = 7ms
PC>

LABORATORY PROGRAM – 7

Demonstrate the TTL/ Life of a Packet

Q) Demonstrate the TTL/Life of a packet.

→ Observation:

① Before Routing information Protocol:-

In Router 2:-

(Show IP route)

C 30.0.0.0/8 is directly connected

C 50.0.0.0/8 is directly Connected

R 10.0.0.0/8 [120/2] via 50.0.0.1

Procedure:-

① Demonstrate TTL using Same Topology.

② Send a Simple PDU for end Device 10.0.0.2 to 30.0.0.3 in Simulation Mode

③ Create the Topology as Shown in figure ~~before~~ And also Configure the Routers before Demonstrating using TTL

Observation:-

① Router0:
Inbound Details: TTL: 255
Outbound Details: TTL: 254

② Router1
Inbound Details: TTL: 254
OutBound Details: TTL: 253

③ Router 2
Inbound Details: TTL: 253
OutBound Details: TTL: 252

TTL reduces after passing through every router

Topology:-

PDU Information at Device: Router0
OSI Model
Inbound PDU Details
Outbound PDU Details
At Device: Router0
Source: PC0
Destination: PC3
In Layers
Layer7
Layer6
Layer5
Layer4
Layer 3: IP Header Src. IP: 10.0.0.2, Dest. IP: 20.0.0.3 ICMP Message Type: 8
Layer 2: Ethernet II Header 000A.41E3.E33A >> 0010.11A0.4697
Layer 1: Port FastEthernet0/0
Out Layers
Layer7
Layer6
Layer5
Layer4
Layer 3: IP Header Src. IP: 10.0.0.2, Dest. IP: 20.0.0.3 ICMP Message Type: 8
Layer 2: HDLC Frame HDLC
Layer 1: Port(s): Serial2/0
1. FastEthernet0/0 receives the frame.
Challenge Me
<< Previous Layer
Next Layer >>

PDU Information at Device: Router0
OSI Model
Inbound PDU Details
Outbound PDU Details
PDU Formats
Ethernet II
0 4 8 14 19 Byt
PREAMBLE: 101010...1011
DEST MAC: 0010.11A0.4697
SRC MAC: 000A.41E3.E33A
TYPE: 0x800
DATA (VARIABLE LENGTH)
FCS: 0x0
IP
0 4 8 16 19 31 Bits
4
IHL
DSCP: 0x0
TL: 28
ID: 0xa
0x0
0x0
TTL: 255
PRO: 0x1
CHKSUM
SRC IP: 10.0.0.2
DST IP: 20.0.0.3
OPT: 0x0
0x0
DATA (VARIABLE LENGTH)
ICMP
0 8 16 31 Bits
TYPE: 0x8
CODE: 0x0
CHECKSUM

PDU Information at Device: Router0

OSI Model | Inbound PDU Details | Outbound PDU Details

PDU Formats

HDLC

0	8	16	32	32+x	48+x 56+x
FLG: 0111 1110	ADR: 0x8f	CONTROL: 0x0	DATA: (VARIABLE LENGTH)	FCS: 0x0	FLG: 0111 1110

IP

0	4	8	16	19	31 Bits
4	IHL	DSCP: 0x0	TL: 28		
ID: 0xa			0x0	0x0	
TTL: 254		PRO: 0x1	CHKSUM		
SRC IP: 10.0.0.2					
DST IP: 20.0.0.3					
OPT: 0x0				0x0	
DATA (VARIABLE LENGTH)					

ICMP

0	8	16	31 Bits
TYPE: 0x8	CODE: 0x0	CHECKSUM	
ID: 0x5		SEQ NUMBER: 10	

LABORATORY PROGRAM – 8

Configure Web Server, DNS within a LAN.

18/12/24

Lab No. 11

DNS (Domain name System)

Aim:- Configure Web Server, DNS within a LAN

Topology:-

Procedure:-

① Set up the LAN as per the topology mentioned above & Configure the devices.

② Go to Server → Services → DNS :
Name: bmsce [Domain name]
Address: 10.0.0.2
Add the mapping of domain name to address

③ Go to PC → Config → Global → Setting → DNS Server: 10.0.0.2
[The Server that provides DNS mapping]

④ Go to PC → Desktop → Web Browser
Type the URL: http://bmsce

Fa0/1
Switch-PT
Switch0
Fa1/1
Fa0
PC-PT
PC0
10.0.0.1
Fa0
Server-PT
Server0
10.0.0.2

PC0
Physical
Config
Desktop
Custom Interface
Web Browser
X
<
>
URL
http://bmsce
Go
Stop
Cisco Packet Tracer
Welcome to my page. Opening doors to new opportunities. Mind Wide Open.
Quick Links:
A small page
Copyrights
Image page
Image

LABORATORY PROGRAM – 9

To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP)

18/12/2024

Lab No 10

ARP (Address Resolution Protocol)

Aim:- To Construct Simple LAN and understand the concept and Operation of address Resolution protocol (ARP)

Topology:

Procedure:

1. Create the topology as shown above
2. Configure the PC's and Server.
3. Click on Inspect mode (🔍), then click on the end devices and open ARP tables.
4. Send a data packet from any end device say server to other end devices say 10.0.0.3 PC.
5. Open simulation mode to capture each step of data transfer.

OBSERVATIONS:

① The ARP tables of all end devices are initially empty.

② When the data packet from Server arrives at Switch, Since the Source MAC address is unknown, it sends a broadcast message to all devices.

③ The device with the IP address present in the destination address of the data packet responds to the message

④ The Server and the PC Update their ARP Tables matching IP address to MAC address.

⑤ Overtime, the ARP tables grows as datapackets are set.

⑥ The MAC Table of the Switch which was initially empty Updates its mac table gradually too

ARP Table for 10.0.0.4:-

IP address	Hardware Address	Interface
10.0.0.3	0001.C726.A7E5	FastEthernet0

⑦ Similarly other ARP tables are Updated.

Logical
[Root]
New Cluster
Move Object
Set Tiled Background
ARP Table for PC0
IP Addre: Hardware Address Interface
10.0.0.2 0003.E490.6097 FastEthernet0
ARP Table for PC1
IP Addre: Hardware Address Interface
10.0.0.1 0004.9A10.2391 FastEthernet0
ARP Table for PC2
IP Addre: Hardware Address Interface
ARP Table for Server0
IP Addre: Hardware Address Interface
ARP Table for Switch0
IP Addre: Hardware Address Interface
PC-PT PC0
10.0.0.1
PC-PT PC1
10.0.0.2
PC-PT PC2
10.0.0.3
Server-PT Server0
Switch0
Physical
Config
CLI
IOS Command Line Interface
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/1, changed state to up
%LINK-5-CHANGED: Interface FastEthernet2/1, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet2/1, changed state to up
%LINK-5-CHANGED: Interface FastEthernet3/1, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet3/1, changed state to up
Switch>show mac address-table
Mac Address Table
Vlan Mac Address Type Ports
1 0003.e490.6097 DYNAMIC Fa1/1
1 0004.9a10.2391 DYNAMIC Fa0/1
1 000a.41b0.b710 DYNAMIC Fa3/1
Switch>
Copy
Paste
Simulation Panel
Event List
Reset Simulation
Constant Delay

PC1
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>arp -a
No ARP Entries Found
PC>

LABORATORY PROGRAM – 10

To understand the operation of TELNET by accessing the router in server room from a PC in IT office.

18/12/24

Lab No 12
TELNET

Aim:- To understand the operation of TELNET by accessing the router in server room from a PC in IT office.

Topology:-

Procedure:-

① Create the topology as given above and Configure the devices.

② Commands in Router:-

```
Router>enable
Router# Config Terminal
Router(config)# hostname R1
R1(Config)# enable secret 1234
R1(Config)# interface fastethernet 0/0
R1(Config-if)# ip address 10.0.0.2 225.0.0.0
R1(Config-if)# no shut

R1(Config-if)# line Vty 0 3
R1(Config-line)# login
% Login disabled on line 194, untill 'password' is set
R1(Config-line)# password 4321
R1(Config-line)# exit.
R1(Config)# exit
```

enable password

~~User access~~ verification password

R1 # wr
Building Configuration...
[OK]
Note:- Vty 0 3 : first four visited terminal lines for Telnet access.

③ In PC: Command Prompt:
→ first try pinging to see if devices are connected

PC> telnet 10.0.0.2
Trying 10.0.0.2 .... open

User access Verification
Password: 4321
Password: 4321
R1 > enable
Password: 1234
R1 # show ip route

C 10.0.0.0/8 is directly connected, fastethernet 0/0
R1#

OBSERVATIONS:
① The admin in PC is able to run commands as run in router CLI and See results from PC.
② Telnet allows users to establish a remote session with another device like router, Over a TCP/IP network.
③ Using telnet, we can access and control the remote devices CLI as if you were physically connected to it.

Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>telnet 10.0.0.2
Trying 10.0.0.2 ...Open


User Access Verification

Password:
R1>enable
Password:
R1#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, FastEthernet0/0
R1#
```

LABORATORY PROGRAM – 11

To construct a VLAN and make the PC's communicate among a VLAN

18/12/24

lab No. 9

VLAN

Aim: To construct a VLAN and make the PC's communicate among a VLAN.

Topology:-

Procedure:-

① Place a 1841 Router, a Switch and PCs

② Connect the four PCs to the Switch Via fasthethernet.

③ Since only 4 fasthethernet ports are available in the Switch, we have add an ethernet port.

④ Switch of the power button of the Switch
→ Add the ethernet port to the Switch
→ Switch on the power button
→ Connect the router to the Switch via ethernet 6/1

⑤ In the Switch, go to Config Tab and
→ Select VLAN Database
→ Give VLAN number (Say 2)
→ Give VLAN name (Say 'cseise')
→ Add it to the Database

⑥ Select the Switch:
→ Go to Config
→ Go to ethernet 6/1 i.e Connected to router
→ Make it the trunk

⑦ Configure the PCs as shown in the Topology

⑧ Select Switch:
→ Go to Config
→ Go to fastethernet 2/1
→ Set VLAN number as 2 i.e 'cseise'
→ Similarly set VLAN 2 for fastethernet 3/1 interface

⑨ Configure the router:
Router(Config) # inter
Router(Config-if) # ip address 192.168.1.1 255.255.255.0
Router(Config-if) # no Shut
Router(Config-if) # exit

Now to configure the router's VLAN interface:
Router(Config) # interface fastethernet 0/0.1
Router(Config-Subif) # encapsulation dot1q 2
Router(Config-Subif) # ip address 192.168.2.1 255.255.255.0
Router(Config-subif) # noshut
Router(Config-Subif) # exit

(10) Ping devices within the Same VLAN and to devices of different VLAN.

OBSERVATIONS :

① When devices ar pinged within Same VLAN:
→ Pinging 192.168.1.3 from 192.168.1.2
→ data packet doesnt go the router
→ The Switch forwards the packets without the need of the router.

② When a device pings a device of another VLAN
→ Pinging 192.168.2.3 from 192.168.1.2
→ The data packets journey is as follows
192.168.1.2 → Switch → Router → Switch → 192.168.2.3

③ VLANs divide a Single Switch into multiple logical Switches
→ Devices in one VLAN connot directly communicate with devices in another VLAN without a router

④ Traffic isolation:
→ Each VLAN maintains its own broadcast domain
→ BroadCasts Sent by devices in one VLAN donot reach devices in another VLAN.

⑤ VLAN trunking allows Switches to forward frames from different VLANs over a single link called trunk.
→ This is done by adding an additional header information Called tag to the ethernet frame - VLAN Tagging

192.168.1.1
Fa0/0/41
Router0
192.168.2.1
Eth6/1
Fa0/1
Fa1/1
Fa3/1
Fa2/1
Fa0
PC-PT
PC0
192.168.1.2
dg:192.168.1.1
Fa0
PC-PT
PC1
192.168.1.3
dg:192.168.1.1
Fa0
PC-PT
PC2
192.168.2.2
dg:192.168.2.1
Fa0
PC-PT
PC3
192.168.2.3
dg:192.168.2.1

Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Request timed out.
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=4ms TTL=127

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 4ms, Average = 1ms

PC>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127
Reply from 192.168.2.2: bytes=32 time=2ms TTL=127
Reply from 192.168.2.2: bytes=32 time=0ms TTL=127

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms

PC>ping 192.168.2.3

Pinging 192.168.2.3 with 32 bytes of data:

Request timed out.
Reply from 192.168.2.3: bytes=32 time=3ms TTL=127
Reply from 192.168.2.3: bytes=32 time=2ms TTL=127
Reply from 192.168.2.3: bytes=32 time=1ms TTL=127

Ping statistics for 192.168.2.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 3ms, Average = 2ms

PC>ping 192.168.2.3

Pinging 192.168.2.3 with 32 bytes of data:

Reply from 192.168.2.3: bytes=32 time=0ms TTL=127
Reply from 192.168.2.3: bytes=32 time=0ms TTL=127
Reply from 192.168.2.3: bytes=32 time=2ms TTL=127
Reply from 192.168.2.3: bytes=32 time=0ms TTL=127

Ping statistics for 192.168.2.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms

PC>
```

LABORATORY PROGRAM – 12

To construct a WLAN and make the nodes communicate wirelessly

18/12/24

Lab No 13

WLAN (Wireless LAN)

Aim:- To construct a Wireless LAN and make the nodes communicate wirelessly.

Initial Topology:

Procedure:-

① Create the Topology as given above and configure the devices.

② Configure Access Point:

click Accesspoint → Config → Port 1:

SSID: bmsce

Select ⊙ WEP

Set key: 1234567890

③ Configure PC & Laptop with wireless standards.

→ Switch off Device

→ Drag the existing PT-HOST-NM-LAM to the component listed in the LHS of Physical

→ Drag WMP300N wireless interface to the empty pot.

→ Switch on the device

④ In the config Tab, a new wireless interface was added.

(5) configure the device by entering SSID, WEP, WEP key, IP address and Gateway.

Topology after wireless Configuration:-

(6) Ping from every device to every other device to check for connection.

Observation:-

(1) we were able to ping from every device to every other device

(2) #Access point:
→ Creates bridge between wired and wireless devices
SSID broadcasting → announces the wireless network's name (SSID) to allow devices to connect using WEP, WPA or WPA2

(3) WMP300N wireless interface;
→ Wireless network adapter that enables devices to communicate with access point using wireless signals

(4) Pinging: 10.0.0.1 to 10.0.0.3:
10.0.0.1 → Switch → AccessPoint → 10.0.0.3
This is after the ARP tables are updated after broadcasting

(5) Pinging: 10.0.0.3 to 10.0.0.1
10.0.0.3 → Accesspoint → Switch → 10.0.0.1

Fa0/0
Router-PT
Router0
Fa1/1
Fa2/1
Fa0/1
Switch-PT
Switch0
Access Point-PT
Access Point0
Fa0
PC-PT
PC0
10.0.0.2
PC-PT
PC1
10.0.0.3
Laptop-PT
Laptop0
10.0.0.4

PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.3
Pinging 10.0.0.3 with 32 bytes of data:
Reply from 10.0.0.3: bytes=32 time=22ms TTL=128
Reply from 10.0.0.3: bytes=32 time=6ms TTL=128
Reply from 10.0.0.3: bytes=32 time=3ms TTL=128
Reply from 10.0.0.3: bytes=32 time=7ms TTL=128
Ping statistics for 10.0.0.3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 3ms, Maximum = 22ms, Average = 9ms
PC>ping 10.0.0.4
Pinging 10.0.0.4 with 32 bytes of data:
Reply from 10.0.0.4: bytes=32 time=19ms TTL=128
Reply from 10.0.0.4: bytes=32 time=5ms TTL=128
Reply from 10.0.0.4: bytes=32 time=6ms TTL=128
Reply from 10.0.0.4: bytes=32 time=7ms TTL=128
Ping statistics for 10.0.0.4:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 5ms, Maximum = 19ms, Average = 9ms
PC>

\

LABORATORY PROGRAM – 1

Write a program for error detecting code using CRC-CCITT (8-bits).

Cycle 2

Programs

(P1) CRC

Write a Program for error detecting code using CRC-CCITT(16-bits)

Python Code:-

```
def crc_ccitt(data):
    polynomial = 0X1021
    initial_crc = 0XFFFF
    crc = initial_crc
    for byte in data.encode():
        crc ^= (byte<<8)
        for _ in range(8):
            if crc & 0X8000:
                crc = (crc<<1) ^ polynomial
            else:
                crc<<=1
            crc &= 0X ffff
    return crc
data = input('Enter data to calculate CRC-CCITT:')
print(f'checksum: {CRC_ccitt(data): 04X}')
```

→ if MSB is 1, sumulate division else move to next

→ Marking: ensures CRC remains 16 bit

Output:-

Enter data to Calculate CRC-CCITT: 1234

CRC-CCITT checksum: 5349

Python Code:-

```
def crc_ccitt(data):
    polynomial = 0X1021
    initial_crc = 0XFFFF
    crc = initial_crc
    for byte in data.encode():
        crc ^= (byte<<8)
        for _ in range(8):
            if crc & 0X8000:
                crc = (crc<<1) ^ polynomial
            else:
                crc<<=1
            crc &= 0X ffff
    return crc
data = input('Enter data to calculate CRC-CCITT:')
print(f'checksum: {CRC-ccitt(data): 04X}')
```

→ if MSB is 1, sumulate division else move to next

→ Marking: ensure CRC remains 16 bit

Output:-

Enter data to Calculate CRC-CCITT: 1234

CRC-CCITT checksum: 5349

```
    return crc
data = input("Enter data to calculate CRC-CCITT: ")
print(f"checksum: {CRC_CCitt(data):04X}")
```

→ if MSB is 1, sumulate division else move to next

→ Marking: ensures CRC remains 16 bit

Output:-

Enter data to Calculate CRC-CCITT: 1234

CRC-CCITT checksum: 5349

Code

```
def xor(dividend, divisor):
    """Perform XOR operation between dividend and divisor."""
    result = ''
    for i in range(1, len(divisor)):
        result += '0' if dividend[i] == divisor[i] else '1'
    return result

def crc(data, gen_poly):
    """Compute the CRC check value using CRC-CCITT (8-bit)."""
    data_length = len(data)
    gen_length = len(gen_poly)

    # Append n-1 zeros to the data
    padded_data = data + '0' * (gen_length - 1)
    check_value = padded_data[:gen_length]

    for i in range(data_length):
        if check_value[0] == '1':
            # XOR operation if the first bit is 1
            check_value = xor(check_value, gen_poly)
        else:
            # Retain original check value if first bit is 0
            check_value = check_value[1:]

        # Shift left and add the next data bit
        if i + gen_length < len(padded_data):
            check_value += padded_data[i + gen_length]

    return check_value[1:]  # Remove the leading bit

def receiver(data, gen_poly):
    """Simulate the receiver side to check for errors."""
    print("\n--------------------")
    print("Data received:", data)

    # Perform CRC computation on received data
    remainder = crc(data, gen_poly)

    # Check if the remainder is all zeros
    if '1' in remainder:
        print("Error detected")
    else:
        print("No error detected")

if __name__== "__main__":
    # Input data and generator polynomial
    data = input("Enter data to be transmitted: ")
    gen_poly = input("Enter the Generating polynomial: ")
```

```
# Compute CRC check value
check_value = crc(data, gen_poly)
print("\n----------------------------")
print("Data padded with n-1 zeros:", data + '0' * (len(gen_poly) - 1))
print("CRC or Check value is:", check_value)

# Append check value to data for transmission
transmitted_data = data + check_value
print("Final data to be sent:", transmitted_data)
print("----------------------------\n")

# Simulate the receiver side
received_data = input("Enter the received data: ")
receiver(received_data, gen_poly)
```

Output

```
Enter data to be transmitted: 1001100
Enter the Generating polynomial: 100001011

----------------------------------------
Data padded with n-1 zeros: 100110000000000
CRC or Check value is: 0100010
Final data to be sent: 10011000100010
----------------------------------------

Enter the received data: 10011000100011

------------------------------
Data received: 10011000100011
Error detected
```

LABORATORY PROGRAM – 2

Write a program for congestion control using Leaky bucket algorithm.

(P2)

Write a program for congestion control using Leaky bucket Algorithm

- Imagine a bucket with a Small hole at the bottom.
- Water(packets) can be added to the bucket, but it can only leave through the hole at constant rate (Output rate)
- If the bucket is full and more water is added, the excess water overflows (packets are dropped)

Python Code:-

```
import time
import random

NUM_Packets = 5
def leaky_bucket(Output_rate, bucket_size):
    packet_sizes=[random.randint(1,100) for_in range
                (NUM_Packets)]
    print("Incoming packets:")
    for i, packet in enumerate(packet_sizes):
        print(f"Packet[{i}]: {packet} bytes")
    remaining_bytes=0
    for i, packet in enumerate(packet_sizes):
        print(f"Processing Packet[{i} of Size{packet}
                                bytes...")

        if packet > bucket_size:
            print(f"PacketSize{packet} exceed bucket
                capacity {bucket_size}
                - Packet Rejected" }

        Continue
```

Code

```
# Getting user inputs
storage = int(input("Enter initial packets in the bucket: "))
no_of_queries = int(input("Enter total no. of times bucket content is checked: "))
bucket_size = int(input("Enter total no. of packets that can be accommodated in the bucket:
"))
input_pkt_size = int(input("Enter no. of packets that enters the bucket at a time: "))
output_pkt_size = int(input("Enter no. of packets that exits the bucket at a time: "))

for i in range(no_of_queries):  # space left
    size_left = bucket_size - storage
    if input_pkt_size <= size_left:
        # update storage
        storage += input_pkt_size
    else:
        print("Packet loss =", input_pkt_size)

    print(f"Buffer size = {storage} out of bucket size = {bucket_size}")

    # as packets are sent out into the network, the size of the storage decreases
    storage -= output_pkt_size
```

Output

```
Enter initial packets in the bucket: 0
Enter total no. of times bucket content is checked: 4
Enter total no. of packets that can be accommodated in the bucket: 10
Enter no. of packets that enters the bucket at a time: 4
Enter no. of packets that exits the bucket at a time: 1
Buffer size = 4 out of bucket size = 10
Buffer size = 7 out of bucket size = 10
Buffer size = 10 out of bucket size = 10
Packet loss = 4
Buffer size = 9 out of bucket size = 10
```

LABORATORY PROGRAM – 3

Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.

(P3)

Using TCP/IP rockets, write a client-Server program to make client Sending the filename and the Server to Send back the contents of requested file if present.

① Client TCP.py:

```
from socket import *
ServerName = '127.0.0.1'
ServerPort = 12000
clientSocket = Socket(AF_INET, SOCK_STREAM)
clientSocket.connect((ServerName, ServerPort))
Sentence = input ("\n Enter file name:")
clientsocket.send(Sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print("\nfromServer:\n")
print(filecontents)
clientSocket.close()
```

Output:

python ServerTCP.py	python Client
① The Server is ready to recv	③ Enter file name: Sample.txt
② Connection established with ('127.0.0.1', 60477)	⑤ from Server: This is a Sample Text file
④ Sent content of Sample.txt	⑥ # close
⑦ Server is ready to reciove	

Code: Client.py

```
from socket import *
serverName = "127.0.0.1"  # Server address (localhost)
serverPort = 12000  # Port number where the server listens

# Create TCP socket
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))  # Connect to server

# Ask user for file name to request
sentence = input("Enter file name: ")

# Send file name to server
clientSocket.send(sentence.encode())

# Receive file contents from server
filecontents = clientSocket.recv(1024).decode()
print('From Server:', filecontents)

# Close the connection
clientSocket.close()
```

Page:
Date: / /

(2) ServerTCP.py

```
from Socket import *
ServerName='127.0.0.1'
ServerPort=12000
ServerSocket=Socket(AF_INET, SOCK_STREAM)
ServerSocket.bind((ServerName, ServerPort))
ServerSocket.listen(1)
while 1:
    print('The Server is ready to recieve')
    Connection Socket, addr=serverSocket.accept()
    print(f'Connection established with {addr}')
    Sentence=connectionSocket.recv(1024).decode()
    file=open(Sentence,'r')
    l=file.read(1024)
    connectionSocket.Send(l.encode())
    print(f'Sent contents of {Sentence}')
    file.close()
    ConnectionSocket.close()
```

#1024 - no. of bytes that the Server will try to send at once from Socket.

Code: Server.py

```
from socket import *
serverName = "127.0.0.1"  # Server address (localhost)
serverPort = 12000  # Port number to listen on

# Create TCP socket
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))  # Bind socket to the address and port
serverSocket.listen(1)  # Listen for 1 connection
print("The server is ready to receive")

while True:
    # Accept a connection
    connectionSocket, addr = serverSocket.accept()

    # Receive the file name from the client
    sentence = connectionSocket.recv(1024).decode()

    # Try opening the file
    try:
        file = open(sentence, "r")  # Open file in read mode
        fileContents = file.read(1024)  # Read file content (up to 1024 bytes)
        connectionSocket.send(fileContents.encode())  # Send file contents to client
        file.close()
    except FileNotFoundError:
        # Send error message if file not found
        connectionSocket.send("File not found".encode())

    # Close the connection
    connectionSocket.close()
```

Output

LABORATORY PROGRAM – 4

Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.

(P4)

(4) Using UDP sockets, write a client Server program to make client Sending the file name and Server to Send back Contents of requested file if present.

(1) Client UDP.Py :-

```
from socket import *
ServerName = '127.0.0.1'
ServerPort = 12000
clientSocket = Socket(AF_INET, SOCK_DGRAM)
Sentence = input('Enter file name')
clientSocket.send to (bytes(sentence, 'utf-8'),
                        ServerName, Serverport))
file contents, serverAddrress = clientSocket.recvfrom
                                        (2048)
print(Reply from Server:)
print(fileconten.decode('utf-8'))
for i in file contents :
        print(str(i), end = ' ')
clientSocket.close()
```

Code: ClientUDP.py

```
from socket import *
serverName = "127.0.0.1"  # Server address (localhost)
serverPort = 12000  # Port number where the server listens

# Create UDP socket
clientSocket = socket(AF_INET, SOCK_DGRAM)

# Ask user for file name to request
sentence = input("Enter file name: ")

# Send the file name to the server using UDP
clientSocket.sendto(sentence.encode("utf-8"), (serverName, serverPort))

# Receive file contents from the server
fileContents, serverAddress = clientSocket.recvfrom(2048)

# Print the file contents received from the server
print("From Server:", fileContents.decode())

# Close the UDP socket
clientSocket.close()
```

(2) Server UDP.Py:

```
from socket import *
ServerName = '127.0.0.1'
ServerPort = 12000
ServerSocket = Socket(AF_INET, SOCK_DGRAM)
ServerSocket.bind(('127.0.0.1', ServerPort))
print('The server is ready to recieve')

while 1:
    sentence, clientAddress = ServerSocket.recvfrom
                                        (2048)

    Sentence = Sentence.decode ('utf-8')
    file = open(Sentence, 'r')
    con = file.read(2048)
    ServerSocket.sendto(bytes (con, 'utf-8'),
                                clientAddress).
    print(f'Sent contents of {Sentence}')
    for i in Sentence:
        print(str(i), end = '')
    file.close()
```

Observations:

→ No connections is established.

Code: ServerUDP.py

```
from socket import *
serverPort = 12000  # Port number to listen on

# Create UDP socket
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))  # Bind the socket to the server address and port

print("The server is ready to receive")

while True:
    # Receive file name from the client
    sentence, clientAddress = serverSocket.recvfrom(2048)

    # Try opening the file
    try:
        file = open(sentence.decode(), "r")  # Open file in read mode
        fileContents = file.read(2048)  # Read file content (up to 2048 bytes)
        serverSocket.sendto(fileContents.encode("utf-8"), clientAddress)  # Send file contents to
client
        file.close()
    except FileNotFoundError:
        # Send error message if file not found
        serverSocket.sendto("File not found".encode("utf-8"), clientAddress)
```

Output

Wireshark

WIRESHARK

Summary:-

① Wireshark is a powerfull open Source network protocol analysers used for capturing and inspecting network traffic in real time

② features:

→ Packetcapture displaypackets detailed formats
→ filtering - Isolate protocols for analysis
→ Protocol analysis breakdown for details.
→ Packet reassembly - fragmentation & reassembly analysis for application layer communication,

③ Hands on usage:-

→ Captured live traffic.
→ Used filters lets http and ip.src==<IP> to narrow down packets of interest.
→ Examined packet details such as Source/dest IPs, points, payload data & timestamps
→ Saved captured traffic to .pcap file for future analysis.

④ Learnings:-

→ Gained insight into how data is transmitted across a network
→ Understood Structure of common network protocols & their role in communication

Wireshark provide valuable insights into network behaviour, helps diagonse issues, and enhances Understanding of protocol level Communication.