MODERN NETWORKING

Practical file

MSc Part 1

Prof. Ismail H. Popatia
Maharashtra College

Index

Configure IP SLA Tracking and Path Control Topology

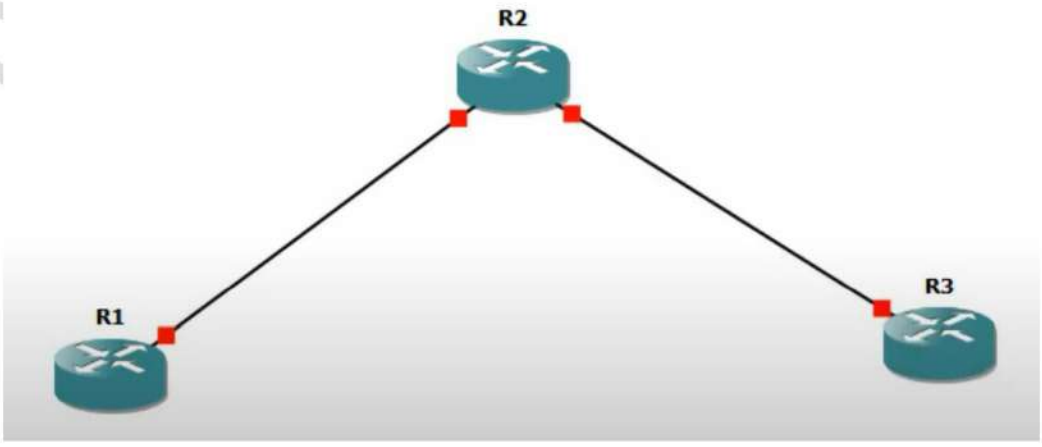
Aim: To configure and verify IP SLA (Service Level Agreement) tracking and path control on Cisco routers, enabling dynamic failover between multiple ISPs based on real-time network performance metrics.

Theory: IP SLA is a Cisco IOS feature that allows active monitoring of network performance by generating traffic and measuring parameters like delay, jitter, and packet loss. This proactive monitoring helps in assessing the quality of network paths.

Path Control involves directing traffic over specific network paths based on performance metrics. By integrating IP SLA with tracking objects, routers can make intelligent routing decisions, ensuring optimal path selection and network resilience.

Tracking Objects are used to monitor the status of IP SLA operations. If an IP SLA operation fails (e.g., due to high latency or packet loss), the tracking object reflects this state, allowing the router to adjust its routing decisions accordingly.

Topology:



Commands:

Configure IP SLA Operations
R1(config)# ip sla 1
R1(config-ip-sla)# icmp-echo 209.165.200.254 source-interface Serial0/0/0
R1(config-ip-sla-echo)# timeout 5000
R1(config-ip-sla-echo)# frequency 10
R1(config)# ip sla schedule 1 life forever start-time now
Explanation: This configuration sets up an ICMP echo operation (ping) to the IP address 209.165.200.254, using Serial0/0/0 as the source interface. The operation has a timeout of 5000 ms and a frequency of 10 seconds.

Configure Tracking Objects
R1(config)# track 1 ip sla 1 reachability
R1(config-track)# delay down 10 up 1
R1(config-track)# exit
Explanation: This command creates a tracking object (ID 1) that monitors the reachability of IP SLA operation 1. The delay parameters ensure that transient failures don't cause unnecessary route changes.

Configure Static Routes with Tracking
R1(config)# ip route 0.0.0.0 0.0.0.0 209.165.201.1 5 track 1
R1(config)# ip route 0.0.0.0 0.0.0.0 209.165.202.129 10 track 2
Explanation: These commands set up two default routes with different administrative distances. The first route uses 209.165.201.1 with a distance of 5 and is tracked by object 1. The second route uses 209.165.202.129 with a distance of 10 and is tracked by object 2.

Configure IP SLA Responder on Remote Router
R2(config)# ip sla responder
Explanation: This command enables the IP SLA responder on the remote router (R2), allowing it to respond to the ICMP echo requests sent by R1.

Verification:
To verify the IP SLA operation:
R1# show ip sla statistics

To check the status of tracking objects:
R1# show track

To view the routing table and confirm active routes:
R1# show ip route

Conclusion:
By configuring IP SLA tracking and path control, the network can dynamically adjust to changing conditions, ensuring optimal path selection and improved network resilience. This setup is particularly beneficial in scenarios where multiple ISPs are used, allowing for automatic failover in case of link degradation or failure.

For video demonstration of the above practical scan the following QR-code or type the link address

https://youtu.be/mIj40NPOH9I?si=0XWBVaZ27wcaacYt

Using the AS_PATH Attribute

Aim: To understand and demonstrate the use of the AS_PATH attribute in Border Gateway Protocol (BGP) for path selection and traffic engineering by simulating a multi-AS network topology using GNS3.

Theory: The Border Gateway Protocol (BGP) is a standardized exterior gateway protocol designed to exchange routing and reachability information among autonomous systems (ASes) on the Internet. BGP is classified as a path-vector routing protocol, and it makes routing decisions based on paths, network policies, or rule-sets configured by a network administrator .

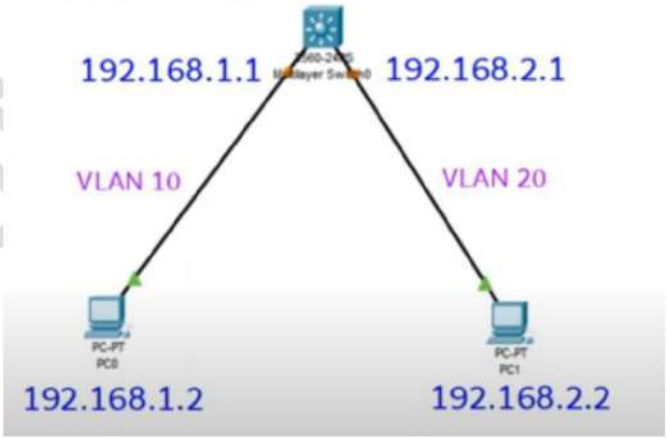
One of the key attributes in BGP is the AS_PATH attribute. This attribute lists the sequence of ASes that routing information has traversed. It serves two primary purposes:

Loop Prevention: By examining the AS_PATH, a BGP router can detect routing loops. If a router sees its own AS number in the AS_PATH of a received route, it will reject that route to prevent a loop .

Path Selection: BGP prefers routes with shorter AS_PATHs, assuming all other attributes are equal. This preference helps in selecting the most efficient path to a destination .

Additionally, network administrators can manipulate the AS_PATH attribute using a technique called AS Path Prepending. This involves adding multiple instances of an AS number to the AS_PATH to make a route less attractive, thereby influencing the path selection process.

Topology:



Code: sa
In the GNS3 simulation, we set up a network topology with three routers, each representing a different AS:

Router R1: AS 100
Router R2: AS 200
Router R3: AS 300

The goal is to advertise a network from R1 and observe how AS_PATH influences the route selection on R3.

Step 1: Configure BGP on R1
```
router bgp 100
 network 192.168.1.0 mask 255.255.255.0
 neighbor 10.0.12.2 remote-as 200
```

Step 2: Configure BGP on R2
```
router bgp 200
 neighbor 10.0.12.1 remote-as 100
 neighbor 10.0.23.3 remote-as 300
```

Step 3: Configure BGP on R3
```
router bgp 300
 neighbor 10.0.23.2 remote-as 200
```
Step 4: Verify AS_PATH on R3

On R3, use the following command to view the BGP table:
```
show ip bgp
```
We observe the route to 192.168.1.0/24 with an AS_PATH of 200 100, indicating that the route has traversed AS 200 and AS 100.

Step 5: Apply AS Path Prepending on R2
To make the path through R2 less preferred, prepend AS 200 multiple times:
```
router bgp 200
 neighbor 10.0.23.3 remote-as 300
 neighbor 10.0.23.3 route-map PREPEND out
route-map PREPEND permit 10
 set as-path prepend 200 200 200
```

After applying the route-map, R3 will see the AS_PATH as 200 200 200 100, making it longer and thus less preferred compared to other available paths.

Conclusion:

This practical demonstrates the significance of the AS_PATH attribute in BGP for both loop prevention and path selection. By manipulating the AS_PATH using techniques like AS Path Prepending, network administrators can influence routing decisions to achieve desired traffic engineering outcomes. The GNS3 simulation effectively illustrates how BGP routers use the AS_PATH attribute to determine the best path to a destination network.

For video demonstration of the above practical scan the following QR-code or type the link address

https://youtu.be/28yj646Wwro?si=WRZ4ePqwTNJQ-zWP

Configuring IBGP and EBGP Sessions, Local Preference, and MED

Aim: To configure IBGP, EBGP and set the local preference and MED

Theory:
Border Gateway Protocol (BGP) Overview

BGP (Border Gateway Protocol) is a routing protocol used to exchange routing information between different networks on the internet. It is classified into two types:

- **External BGP (EBGP)** – Used between different Autonomous Systems (AS).
- **Internal BGP (IBGP)** – Used within the same Autonomous System.

External BGP (EBGP)

- EBGP is used to exchange routes between different Autonomous Systems (ASes).
- It is commonly used by Internet Service Providers (ISPs) and large enterprises to communicate with external networks.
- The default Time-To-Live (TTL) value is 1, meaning that EBGP peers must be directly connected unless explicitly configured otherwise.
- AS-path attribute is used in EBGP to prevent routing loops.
- It prefers shorter AS paths when selecting the best route.
- **Example Scenario:**
  If AS100 wants to exchange routes with AS200, they establish an EBGP connection between their routers.

Internal BGP (IBGP)

- IBGP is used for routing within the same Autonomous System.
- It ensures that all routers in an AS have a consistent view of external routes learned via EBGP.
- Unlike EBGP, IBGP does not modify the AS-path attribute.
- IBGP requires a full mesh of connections (or Route Reflectors/Confederations to reduce overhead).
- Next-hop attribute must be reachable within the AS for proper routing.
- **Example Scenario:**
  If AS100 has multiple routers, they must use IBGP to share routes learned from EBGP peers.

Key Differences Between EBGP and IBGP

Feature	EBGP	IBGP
Used for	Between different ASes	Within the same AS
AS-Path Modification	Yes	No
Next-Hop Change	Yes	No (next-hop must be reachable)
Default TTL	1	255
Full Mesh Required?	No	Yes (or use Route Reflectors)

MED and Local Preference in BGP

BGP (Border Gateway Protocol) uses several attributes to influence routing decisions. Two important attributes that help in path selection are **MED (Multi-Exit Discriminator)** and **Local Preference**.

Multi-Exit Discriminator (MED)

Purpose: MED is used to influence the incoming traffic from an external AS by suggesting the preferred entry point into an AS when multiple links exist.

Characteristics:

- It is an **optional, non-transitive** attribute.
- A **lower MED value** is preferred.
- MED is shared only with **directly connected external neighbors** (not propagated beyond the next AS).
- It is commonly used between ISPs or between enterprise networks and ISPs.
- **Example Scenario:**
  If AS100 has two links to AS200, it can set a lower MED on one link to tell AS200 to prefer that path for incoming traffic.

Local Preference

Purpose: Local Preference is used within an AS to influence the outgoing traffic by selecting the preferred exit point when multiple paths to the same destination exist.

Characteristics:

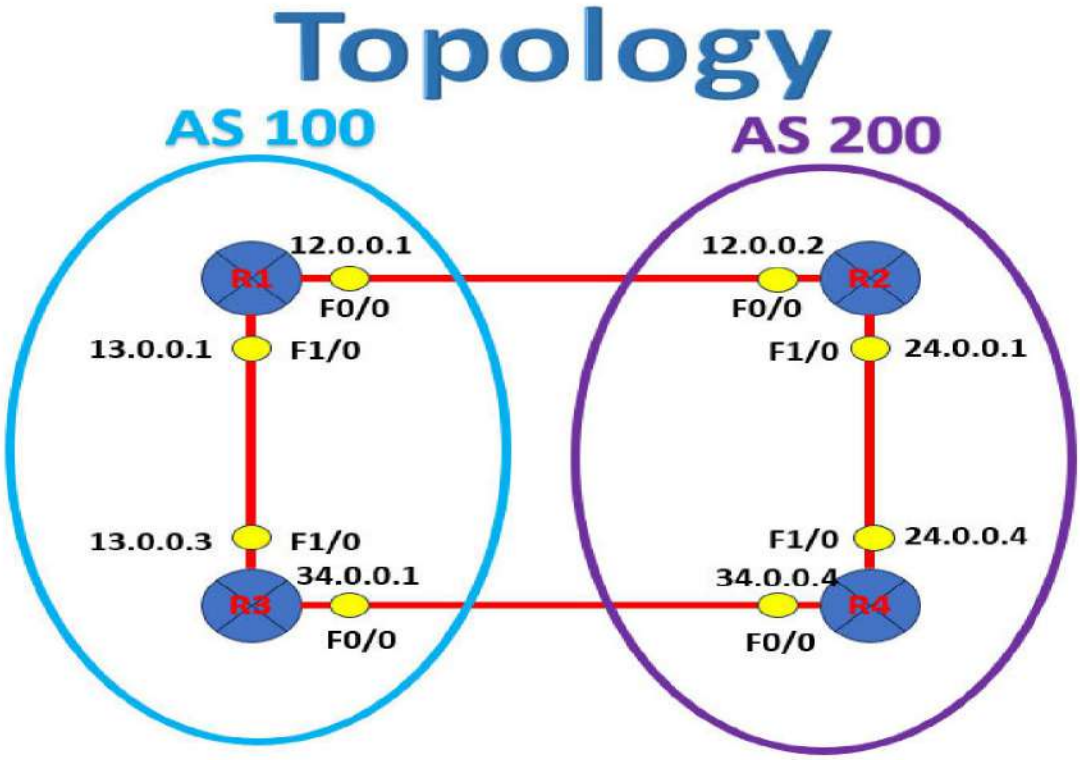
- It is a **well-known, discretionary** attribute.
- A **higher Local Preference value** is preferred.
- It is propagated within the AS to all IBGP peers.
- Used mainly by ISPs and large networks to control outbound traffic flow.
- **Example Scenario:**
  If AS100 has two exit points (R1 and R2) to AS200, setting a higher Local Preference on R1 will make all routers in AS100 prefer R1 for outgoing traffic.

Key Differences Between MED and Local Preference

Feature	MED (Multi-Exit Discriminator)	Local Preference
Function	Controls **incoming traffic** from another AS	Controls **outgoing traffic** within an AS
Preference Rule	Lower value preferred	Higher value preferred
Scope	Shared with EBGP peers but not propagated further	Propagated to all IBGP peers
Attribute Type	Optional, non-transitive	Well-known, discretionary
Used By	External ASes to choose entry points	Internal AS to choose exit points

Both attributes play a crucial role in BGP traffic engineering by influencing how traffic enters and exits an autonomous system.

We use the following topology



We do the configuration using the following steps

Step 1: Configure the IP addresses on all the Routers

Router 1

```
R1#
R1#configure terminal
R1(config)#
R1(config)#interface fastEthernet 0/0
R1(config-if)#
R1(config-if)#ip address 12.0.0.1 255.255.255.0
R1(config-if)#
R1(config-if)#no shutdown
R1(config-if)#
R1(config-if)#exit
R1(config)#
R1(config)#interface fastEthernet 1/0
R1(config-if)#ip address 13.0.0.1 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#
```

```
R1(config-if)#exit
R1(config)
```

Router 2

```
R2#
R2#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)#
R2(config)#interface fastEthernet 0/0
R2(config-if)#
R2(config-if)#ip address 12.0.0.2 255.255.255.0
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#
R2(config)#interface fastEthernet 1/0
R2(config-if)#
R2(config-if)#ip address 24.0.0.1 255.255.255.0
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#
```

Router 3

```
R3#
R3#configure terminal
R3(config)#i
R3(config)#interface fastEthernet 0/0
R3(config-if)#
R3(config-if)#ip address 34.0.0.1 255.255.255.0
R3(config-if)#no shutdown
R3(config-if)#
R3(config-if)#exit
R3(config)#
R3(config)#interface fastEthernet 1/0
R3(config-if)#
R3(config-if)#ip address 13.0.0.3 255.255.255.0
R3(config-if)#no shutdown
R3(config-if)#exit
R3(config)#
```

Router4

```
R4#
R4#configure terminal
R4(config)#
```

```
R4(config)#interface fastEthernet 0/0
R4(config-if)#
R4(config-if)#ip address 34.0.0.4 255.255.255.0
R4(config-if)#no shutdown
R4(config-if)#exit
R4(config)#
R4(config)#interface fastEthernet 1/0
R4(config-if)#
R4(config-if)#ip address 24.0.0.4 255.255.255.0
R4(config-if)#no shutdown
R4(config-if)#exit
R4(config)#
```

Step 2: Set IBGP and EBGP on each router

Router 1

```
R1(config)#router bgp 100
R1(config-router)#
R1(config-router)#neighbor 13.0.0.3 remote-as 100
R1(config-router)#neighbor 13.0.0.3 update-source fastEthernet 1/0
R1(config-router)# network 12.0.0.0 mask 255.255.255.0
R1(config-router)#neighbor 12.0.0.2 remote-as 200
R1(config-router)#network 13.0.0.0 mask 255.255.255.0
R1(config-router)#
R1(config-router)#exit
R1(config)#exit
R1#
```

Router 2

```
R2(config)#router bgp 200
R2(config-router)#
R2(config-router)#neighbor 24.0.0.4 remote-as 200
R2(config-router)#neighbor 24.0.0.4 update-source fastEthernet 1/0
R2(config-router)#network 12.0.0.0 mask 255.255.255.0
R2(config-router)#neighbor 12.0.0.1 remote-as 100
R2(config-router)#network 24.0.0.0 mask 255.255.255.0
R2(config-router)#exit
R2(config)#
```

Router 3

```
R3(config)#router bgp 100
R3(config-router)#
R3(config-router)#neighbor 13.0.0.1 remote-as 100
R3(config-router)#neighbor 13.0.0.1 update-source fastEthernet 1/0
```

```
R3(config-router)#network 34.0.0.0 mask 255.255.255.0
R3(config-router)#neighbor 34.0.0.4 remote-as 200
R3(config-router)#network 13.0.0.0 mask 255.255.255.0
R3(config-router)#
```

Router 4

```
R4(config)#
R4(config)#router bgp 200
R4(config-router)#
R4(config-router)#neighbor 24.0.0.1 remote-as 200
R4(config-router)#neighbor 24.0.0.1 update-source fastEthernet 1/0
R4(config-router)#network 34.0.0.0 mask 255.255.255.0
R4(config-router)#neighbor 34.0.0.3 remote-as 100
R4(config-router)#network 24.0.0.0 mask 255.255.255.0
R4(config-router)#exit
R4(config)#
```

Step 4: Verify the BGP protocol by pinging from Router1 to all interfaces

R1#ping 12.0.0.2

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 12.0.0.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/29/36 ms
```

R1#ping 13.0.0.3

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 13.0.0.3, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/26/36 ms
```

R1#ping 24.0.0.4

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 24.0.0.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 56/60/64 ms
```

R1#ping 34.0.0.4

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 34.0.0.4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 52/56/60 ms
```

Step 5: Configure Local Preference (Influencing Outbound Traffic from AS 100)

```
R1#
R1#configure terminal
R1(config)#
R1(config)#route-map smile_ip permit 10
R1(config-route-map)#
R1(config-route-map)#set local-preference 200
R1(config-route-map)#
R1(config-route-map)#exit
R1(config)#router bgp 100
R1(config-router)#
R1(config-router)#neighbor 13.0.0.3 route-map smile_ip in
R1(config-router)#exit
R1(config)#exit
```

Step 6: Configure MED (Influencing Inbound Traffic to AS 100)

```
R2(config)#
R2(config)#route-map set_med permit 10
R2(config-route-map)#set metric 50
R2(config-route-map)#exit
R2(config)#
R2(config)#router bgp 200
R2(config-router)#
R2(config-router)#neighbor 12.0.0.1 route-map set_med out
R2(config-router)#exit
R2(config)#
```

Step 7: Verification

```
R1#
R1#show ip bgp summary
BGP router identifier 13.0.0.1, local AS number 100
BGP table version is 7, main routing table version 7
4 network entries using 576 bytes of memory
6 path entries using 480 bytes of memory
3/3 BGP path/bestpath attribute entries using 408 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 1488 total bytes of memory
BGP activity 4/0 prefixes, 6/0 paths, scan interval 60 secs

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
12.0.0.2        4   200      19      15        7    0    0 00:08:42        3
13.0.0.3        4   100      12      15        7    0    0 00:07:41        2
```

R1#show ip route

```
Codes: L - local, C - connected, S - static, 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
       i - IS-IS, su - IS-IS summary, 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, H - NHRP, l - LISP
       + - replicated route, % - next hop override

Gateway of last resort is not set

      12.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        12.0.0.0/24 is directly connected, FastEthernet0/0
L        12.0.0.1/32 is directly connected, FastEthernet0/0
      13.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C        13.0.0.0/24 is directly connected, FastEthernet1/0
L        13.0.0.1/32 is directly connected, FastEthernet1/0
      24.0.0.0/24 is subnetted, 1 subnets
B        24.0.0.0 [20/50] via 12.0.0.2, 00:00:24
      34.0.0.0/24 is subnetted, 1 subnets
B        34.0.0.0 [200/0] via 13.0.0.3, 00:07:31
```

R1#show ip bgp

```
BGP table version is 7, local router ID is 13.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 r>  12.0.0.0/24      12.0.0.2                 50             0 200 i
 * i 13.0.0.0/24      13.0.0.3                  0    100      0 i
 *>                    0.0.0.0                  0          32768 i
 *>  24.0.0.0/24      12.0.0.2                 50             0 200 i
 *   34.0.0.0/24      12.0.0.2                 50             0 200 i
 *>i                   13.0.0.3                 0    100      0 i
```

R1#show ip bgp neighbors

```
BGP neighbor is 12.0.0.2,  remote AS 200, external link
 BGP version 4, remote router ID 24.0.0.1
 BGP state = Established, up for 00:09:48
 Last read 00:00:16, last write 00:00:23, hold time is 180, keepalive interval is 60 seconds
 Neighbor sessions:
  1 active, is not multisession capable (disabled)
 Neighbor capabilities:
  Route refresh: advertised and received(new)
  Four-octets ASN Capability: advertised and received
  Address family IPv4 Unicast: advertised and received
  Enhanced Refresh Capability: advertised and received
  Multisession Capability:
  Stateful switchover support enabled: NO for session 1
 Message statistics:
  InQ depth is 0
  OutQ depth is 0

                Sent     Rcvd
  Opens:           1        1
  Notifications:   0        0
  Updates:         3        7
  Keepalives:      12       12
```

For video demonstration of the given practical click on the link below or scan the QR-code

https://youtu.be/lW8dKINlkm8

Secure the Management Plane

Aim: To Secure the management plane using various strategies

Theory: The management plane is responsible for managing a network device — such as configuring settings, monitoring status, and maintaining security. It allows administrators to communicate with the router or switch through command-line interfaces (CLI) or graphical interfaces.

Examples of management plane protocols:
- Telnet (insecure, plaintext)
- SSH (secure)
- SNMP
- HTTP/HTTPS
- Console/VTY access

The management plane is a critical attack surface. If compromised, attackers can:
- View or change device configurations
- Shut down interfaces
- Redirect traffic
- Launch attacks on other devices

Hence, securing it is essential for protecting the entire network infrastructure.

Methods to Secure the Management Plane

1. Use Secure Access Protocols
   Replace insecure protocols like Telnet with SSH, which provides encrypted access to the CLI.

2. Implement User Authentication
   Configure local usernames and passwords, and set privilege levels to limit what users can do.

3. Use Access Control Lists (ACLs)
   Limit access to management interfaces by allowing only authorized IP addresses to connect.

4. Configure Strong Passwords & Encryption
   Use `enable secret`, `service password-encryption`, and avoid weak or default credentials.

5. Disable Unused Services
   Turn off services like CDP, HTTP, FTP, and bootp that are not in use to reduce the attack surface.

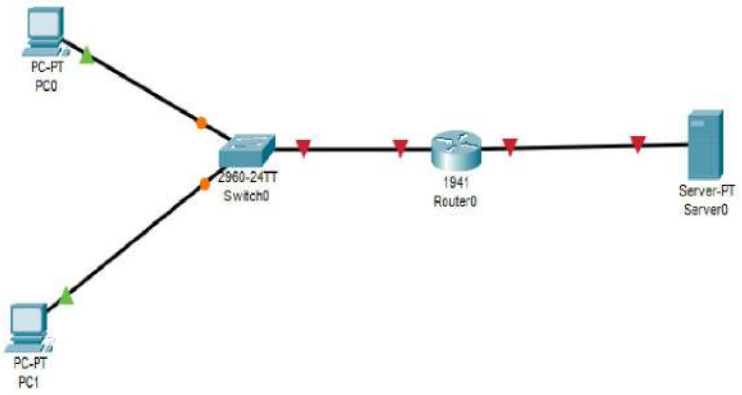
6. Display Banner Warnings
   Configure `banner motd` and `banner login` to display legal notices or warnings to deter unauthorized users.

7. Verify and Test the Setup
Always test your configuration using commands like `show running-config`, `show ip ssh`, `show access-lists`, and attempt authorized and unauthorized access.

Real-World Importance

In enterprise networks, securing the management plane ensures:
- Only authorized personnel can make changes
- Devices are not compromised remotely
- Regulatory compliance with security standards (like ISO, NIST, etc.)
- Reduced risk of configuration tampering, insider threats, and cyber attacks

Topology:
We use the following topology

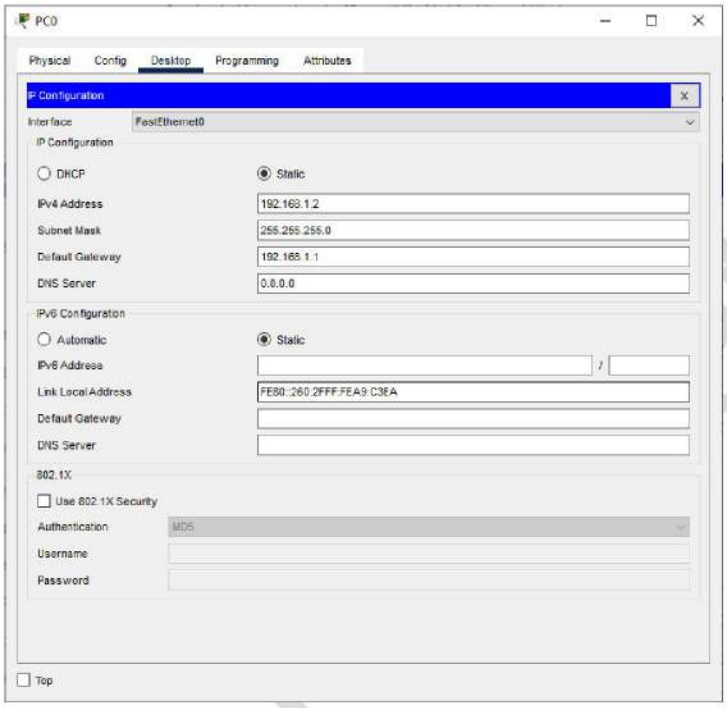


IP Addressing Table:

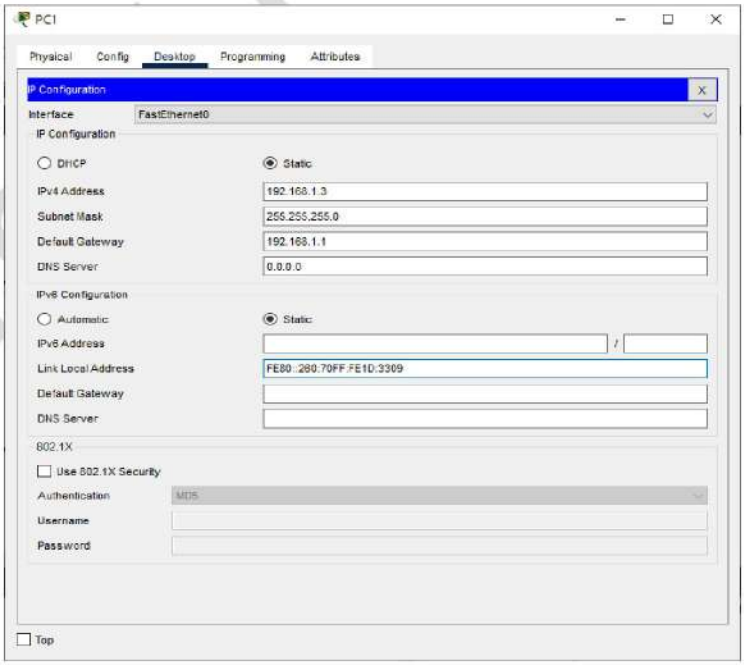
Device	Interface	IP Address	Subnet Mask	Gateway
PC0	FastEthernet0	192.168.1.2	255.255.255.0	192.168.1.1
PC1	FastEthernet0	192.168.1.3	255.255.255.0	192.168.1.1
Router	GigabitEthernet0/0	192.168.1.1	255.255.255.0	
Router	GigabitEthernet0/1	192.168.2.1	255.255.255.0	
Server	FastEthernet0	192.168.2.2	255.255.255.0	192.168.2.1

We Configure the IP addresses as follows

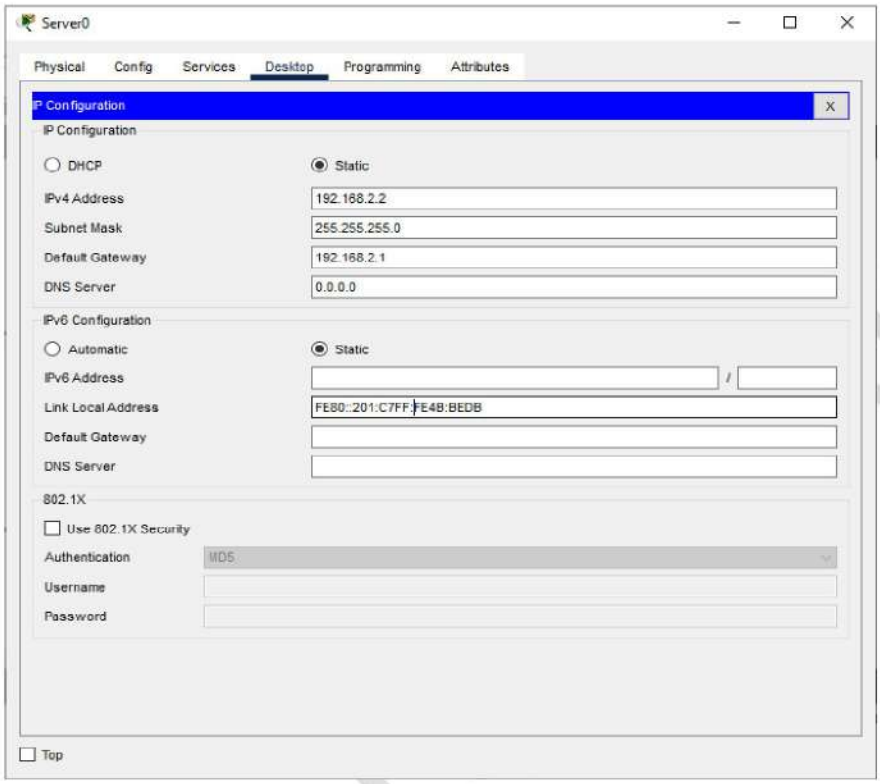
PC0 :



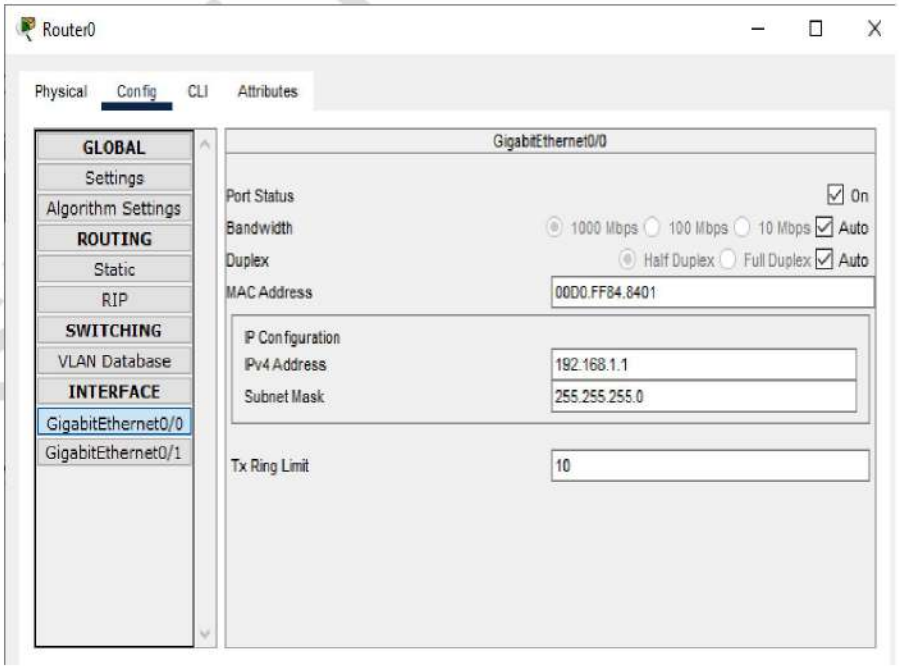
PC1 :



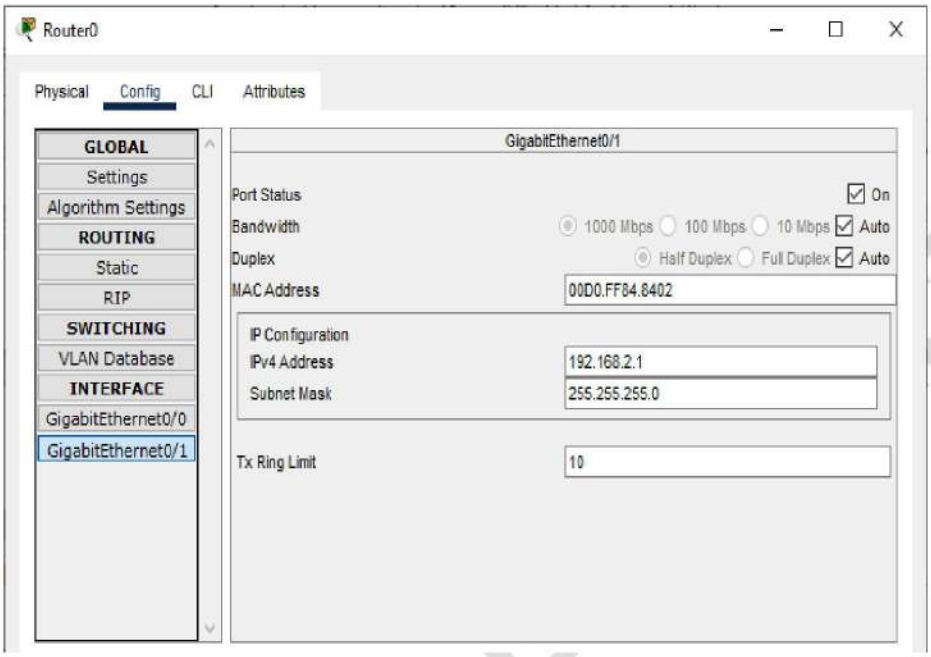
Server:



Router0: interface GigabitEthernet0/0



Router0: interface GigabitEthernet0/1



Now we secure the management plane using the following steps

Step 1: Secure access to Router0 via SSH
Router>enable
Router#
Router#configure terminal
Router(config)#
Router(config)#hostname R1
R1(config)#
R1(config)#ip domain-name ismileacademy.com
R1(config)#
R1(config)#crypto key generate rsa

The name for the keys will be: R1.ismileacademy.com
Choose the size of the key modulus in the range of 360 to 2048 for your
General Purpose Keys. Choosing a key modulus greater than 512 may take
a few minutes.

How many bits in the modulus [512]: 2048
% Generating 2048 bit RSA keys, keys will be non-exportable...[OK]

R1(config)#ip ssh version 2
*Mar 1 0:25:48.611: %SSH-5-ENABLED: SSH 1.99 has been enabled

```
R1(config)#
R1(config)#ip ssh time-out 60
R1(config)#ip ssh authentication-retries 2
R1(config)#username admin privilege 15 secret smilehp
R1(config)#line vty 0 4
R1(config-line)#
R1(config-line)#transport input ssh
R1(config-line)#
R1(config-line)#login local
R1(config-line)#exit
R1(config)#
```

Step 2: Restrict Access to Router using access control list

```
R1(config)#access-list 10 permit 192.168.1.2
R1(config)#access-list 10 deny any
R1(config)#line vty 0 4
R1(config-line)#
R1(config-line)#access-class 10 in
R1(config-line)#exit
R1(config)#
R1(config)#interface gigabitEthernet 0/0
R1(config-if)#
R1(config-if)#ip access-group 10 in
R1(config-if)#exit
R1(config)#
```

Step 3: Disable unused services in the Server

```
R1(config)#
R1(config)#ip access-list extended smile
R1(config-ext-nacl)#
R1(config-ext-nacl)#permit icmp host 192.168.1.2 host 192.168.2.2
R1(config-ext-nacl)#permit tcp host 192.168.1.2 host 192.168.2.2 eq 21
R1(config-ext-nacl)#permit tcp host 192.168.1.2 host 192.168.2.2 eq 22
R1(config-ext-nacl)#deny ip any host 192.168.2.2
R1(config-ext-nacl)#exit
R1(config)#
R1(config)#interface gigabitEthernet 0/1
R1(config-if)#
R1(config-if)#ip access-group smile out
R1(config-if)#exit
R1(config)#
```

Step 4: Configure Banner messages

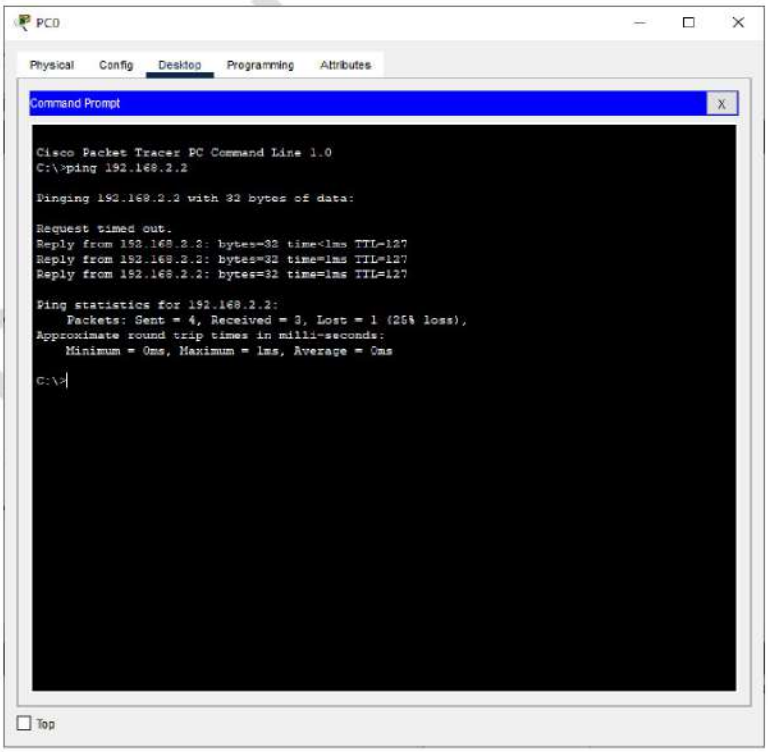
R1(config)#banner motd b
Enter TEXT message. End with the character 'b'.
****
* Unauthorized access is prohibited. Disconnect now! *

R1(config)#
R1(config)#banner login b
Enter TEXT message. End with the character 'b'.
****
* Authorized personnel only. Unauthorized users beware!*

The above steps complete the configuration part of Secure management plane, now we verify it as follows

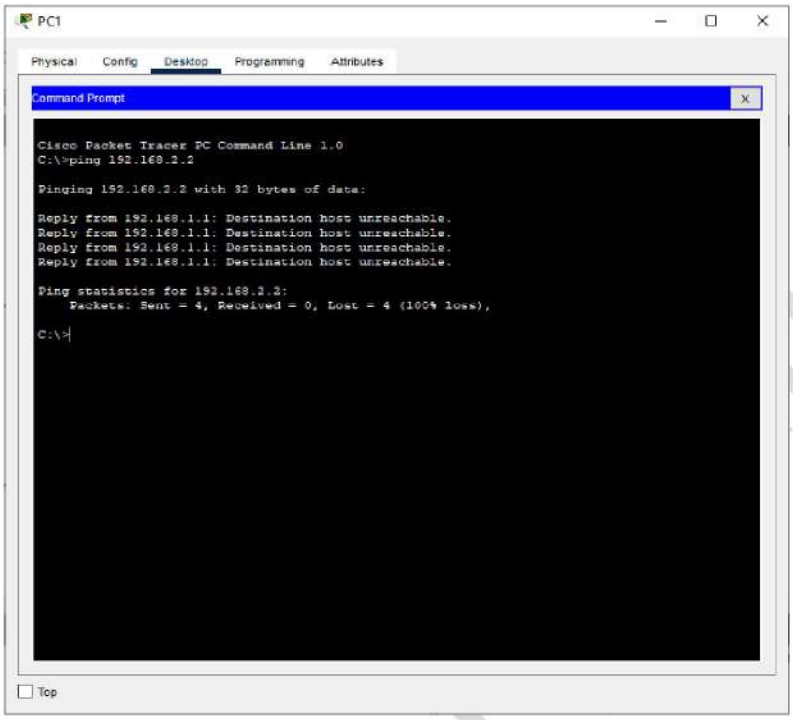
Output:

1) Checking for PING message
   PC0: Checking for ping message from PC0 to Server



It is successful

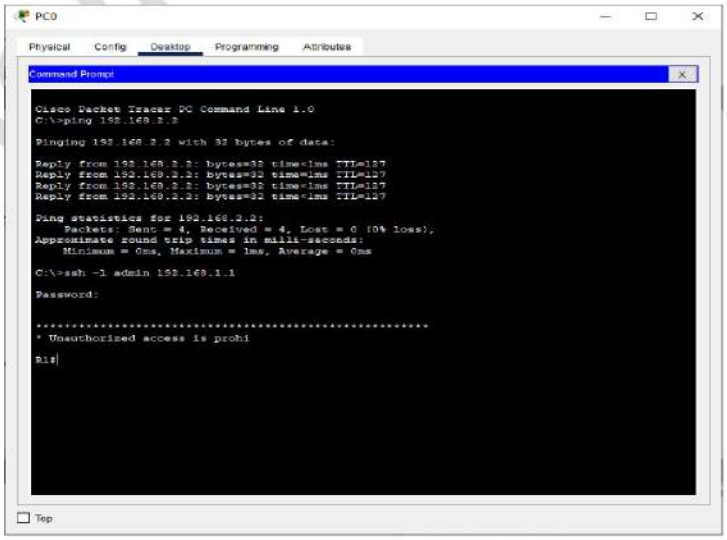
PC1: Checking for ping message from PC1 to Server



It is failure

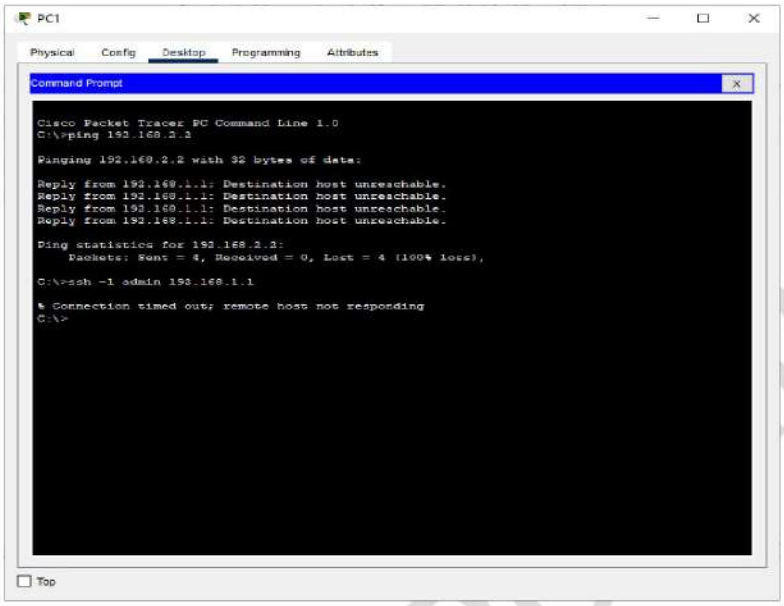
2) Checking for remote login using ssh

PCO:



It is success

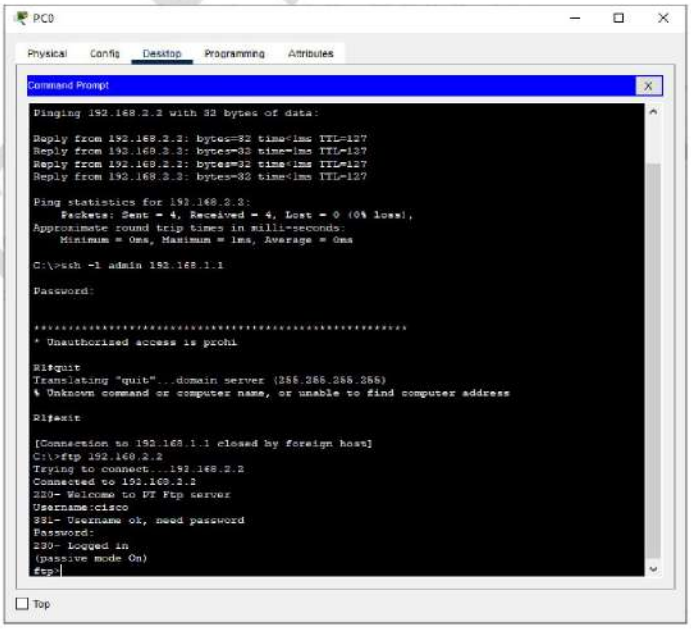
PC1:



It is failure

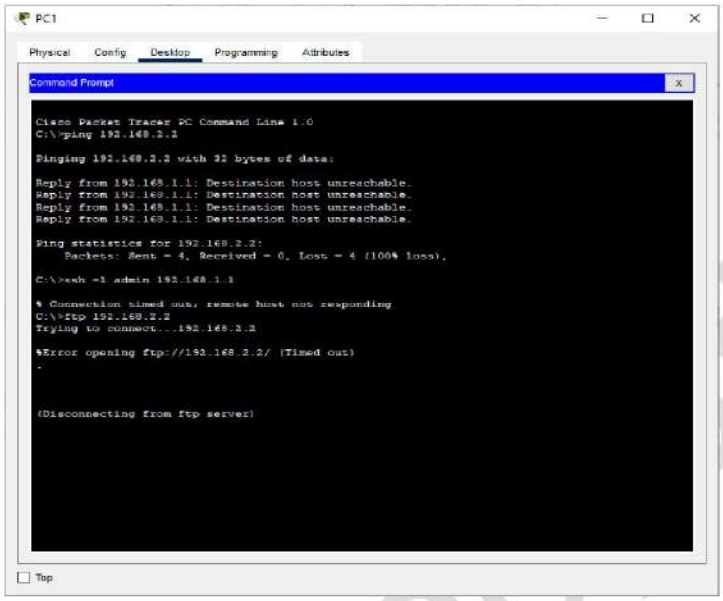
3) Checking for ftp

PC0:



It is success

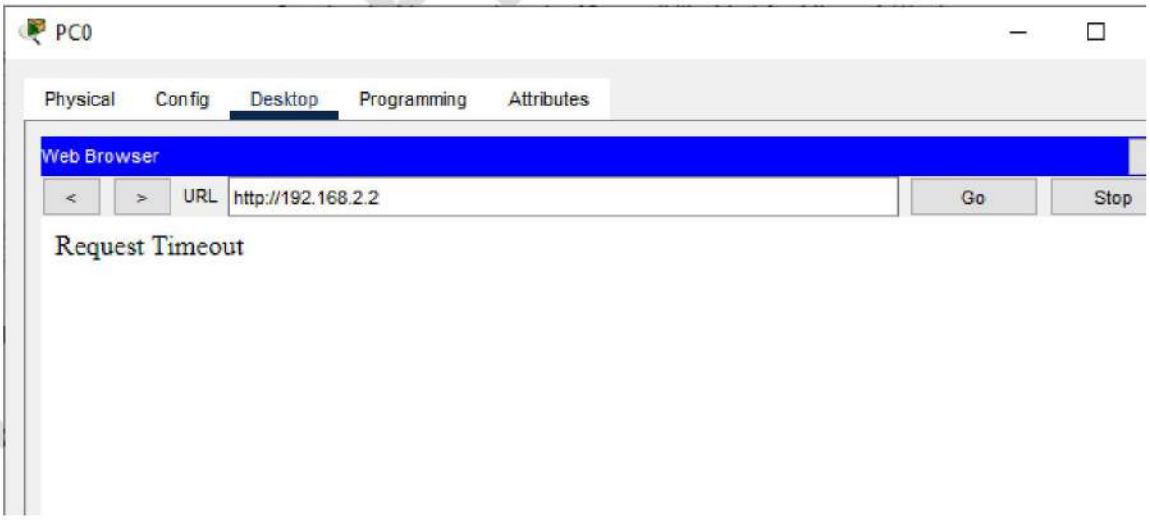
PC1:



It is failure

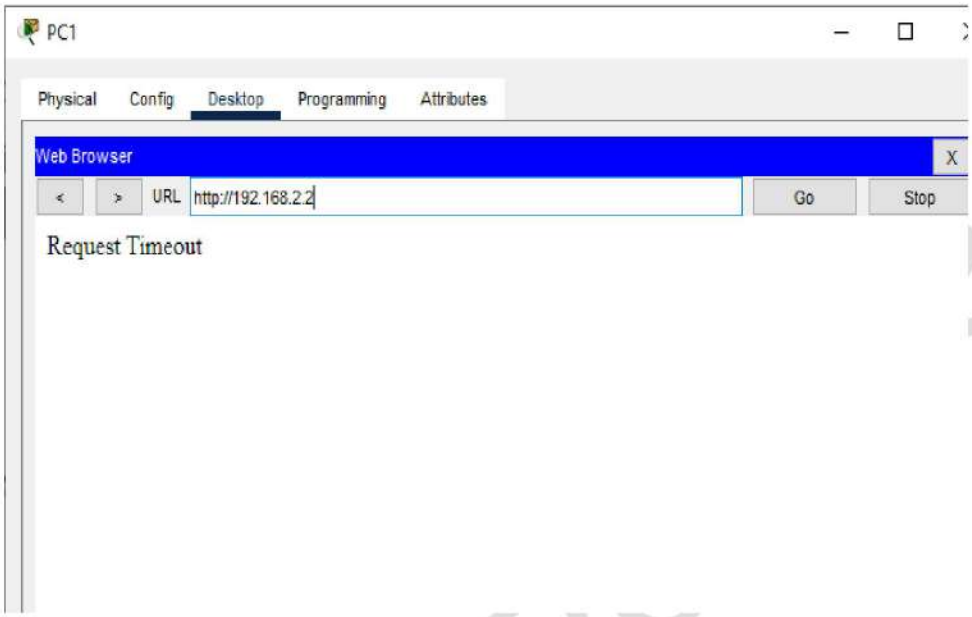
4) Checking for http:

PC0:



It does not connect and times out

PC1:



It does not connect and times out

The following things are configured while performing the Secure management plane and also verified

Test	PC1 (192.168.1.2)	PC2 (192.168.1.3)
Ping server	✅ Allowed	❌ Blocked
FTP to server	✅ Allowed	❌ Blocked
SSH to server	✅ Allowed	❌ Blocked
HTTP/HTTPS/DNS etc.	❌ Blocked	❌ Blocked

For Video demonstration of the given practical click on the link or scan the QR-code

https://youtu.be/OL8lwe0sRLs

Configure and Verify Path Control Using PBR

Aim: To configure and verify path control using Policy Based Routing (PBR)

Theory:
In traditional IP routing, routers forward packets based only on the destination IP address, using the longest prefix match from their routing table.

Policy-Based Routing (PBR) allows you to override this behavior by defining custom routing policies based on other criteria like:

1) Source IP address
2) Protocol
3) Packet size
4) Incoming interface

This gives network administrators greater flexibility in managing traffic flow based on business or technical policies.

PBR is commonly used when you need to:

1) Route specific users or devices through different ISPs or firewalls
2) Apply different Quality of Service (QoS) or bandwidth policies to selected traffic
3) Bypass certain links for sensitive data
4) Perform load balancing or failover routing between multiple paths

PBR uses three main configuration components on a router:

1) Access Control List (ACL): Used to match the traffic that should be treated differently (e.g., match source IPs).
2) Route Map: Defines the policy for matched traffic, such as setting a specific next-hop IP.
3) IP Policy Statement: Applied to an interface using the ip policy route-map command, telling the router to evaluate and apply the route map to incoming traffic on that interface.

Example Use Case
You have two exit routers (R2 and R3).
You want a specific PC (e.g., 192.168.1.2) to reach the internet only through R2, even though R3 is the default route.
PBR allows you to match traffic from that PC and force it through R2, regardless of the routing table.

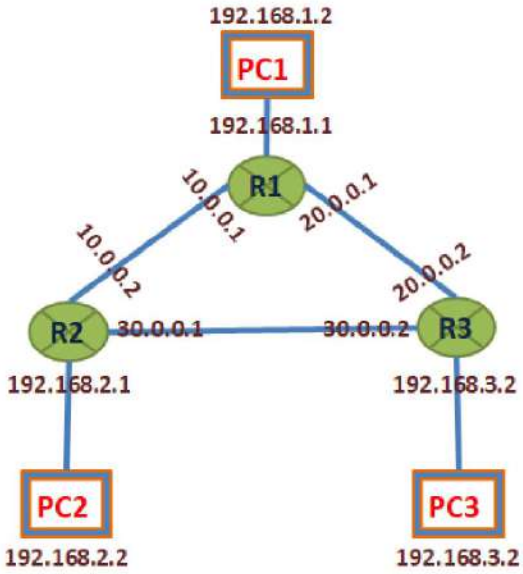
Benefits of PBR

1) Granular control over traffic flows
2) Can enforce security, compliance, and routing policies
3) Useful in multi-homed networks (with multiple ISPs)

Limitations of PBR

1) It only applies to incoming packets on the interface where the policy is configured.
2) Requires careful design to avoid routing loops or black holes
3) Adds some processing overhead on routers (not recommended on low-end devices for high-volume traffic)

We use the following topology



We do the configuration using the following steps

Step 1: Configure the IP addresses on all the Routers and PCs

Router 1

```
R1#
R1#configure terminal
R1(config)#
R1(config)#interface fastEthernet 0/0
R1(config-if)#
R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#

R1(config)#interface fastEthernet 1/0
R1(config-if)#
R1(config-if)#ip address 10.0.0.1 255.0.0.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#

R1(config)#interface fastEthernet 2/0
R1(config-if)#
R1(config-if)#ip address 20.0.0.1 255.0.0.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#
```

Router 2

```
R2#
R2#configure terminal
R2(config)#
R2(config)#interface fastEthernet 0/0
R2(config-if)#
R2(config-if)#ip address 192.168.2.1 255.255.255.0
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#

R2(config)#interface fastEthernet 1/0
R2(config-if)#
R2(config-if)#ip address 10.0.0.2 255.0.0.0
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#

R2(config)#interface fastEthernet 2/0
R2(config-if)#
R2(config-if)#ip address 30.0.0.1 255.0.0.0
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#
```

Router 3

```
R3#
R3#configure terminal
R3(config)#
R3(config)#interface fastEthernet 0/0
R3(config-if)#
R3(config-if)#ip address 192.168.3.1 255.255.255.0
R3(config-if)#no shutdown
R3(config-if)#exit
R3(config)#

R3(config)#interface fastEthernet 2/0
R3(config-if)#
R3(config-if)#ip address 30.0.0.2 255.0.0.0
R3(config-if)#no shutdown
R3(config-if)#exit
R3(config)#

R3(config)#interface fastEthernet 1/0
R3(config-if)#
R3(config-if)#ip address 20.0.0.2 255.0.0.0
R3(config-if)#no shutdown
R3(config-if)#exit
R3(config)#
```

Configuring PC1
PC1>ip 192.168.1.2 255.255.255.0 192.168.1.1

Configuring PC2
PC2>ip 192.168.2.2 255.255.255.0 192.168.2.1

Configuring PC3
PC3>ip 192.168.3.2 255.255.255.0 192.168.3.1

Step 2: Checking the connectivity between the networks using ping

```
R1#ping 192.168.2.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.2.2, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)


R1#ping 192.168.3.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.3.2, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)


R1#ping 192.168.1.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 48/63/88 ms
```

As seen from the above that ping is only successful within the network and fails for the hosts in other networks, hence we need to set the routing path using a suitable routing protocol.

Step 3: Setting up the Routing path using RIPv2 in all the Routers

Router 1
R1(config)#
R1(config)#router rip
R1(config-router)#
R1(config-router)#version 2
R1(config-router)#network 10.0.0.0
R1(config-router)#network 20.0.0.0
R1(config-router)#network 192.168.1.0
R1(config-router)#no auto-summary
R1(config-router)#exit
R1(config)#exit

Router 2

```
R2(config)#
R2(config)#router rip
R2(config-router)#
R2(config-router)#version 2
R2(config-router)#network 10.0.0.0
R2(config-router)#network 30.0.0.0
R2(config-router)#network 192.168.2.0
R2(config-router)#exit
R2(config)#
```

Router 3

```
R3(config)#
R3(config)#router rip
R3(config-router)#
R3(config-router)#version 2
R3(config-router)#network 20.0.0.0
R3(config-router)#network 30.0.0.0
R3(config-router)#network 192.168.3.0
R3(config-router)#exit
R3(config)#
```

Step 4: Checking the connectivity after setting the RIP

```
R1#ping 192.168.2.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 48/63/88 ms


R1#ping 192.168.3.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.3.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 48/63/88 ms


R1#ping 192.168.1.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 48/63/88 ms
```

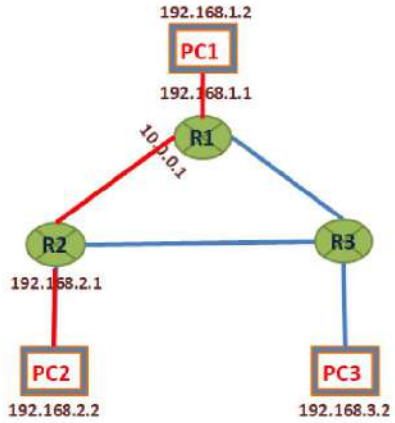
The ping is indeed successful and hence all the hosts are reachable from any host.

Step 5: Checking the path before setting Policy Based Routing (PBR)
Now we trace the route the packets traverse when passing from PC1 to PC2 and PCI to PC3

From PC1 to PC2

```
PC1> trace 192.168.2.2
trace to 192.168.2.2, 8 hops max, press Ctrl+C to stop
 1  192.168.1.1   35.841 ms  4.254 ms  23.860 ms
 2  10.0.0.2   130.033 ms  66.677 ms  72.610 ms
 3   *  *  *
 4  *192.168.2.2   75.028 ms (ICMP type:3, code:3, Destination port unreachable)
```

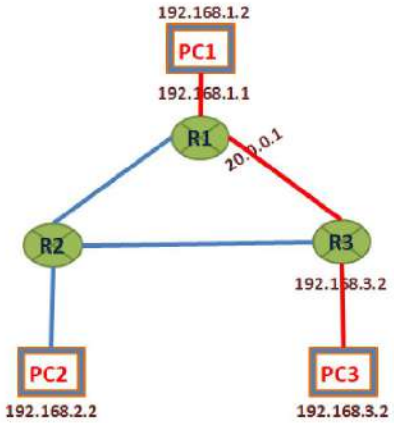
The above route can be visualized as (shown by red lines)



From PC1 to PC3

```
PC1> trace 192.168.3.2
trace to 192.168.3.2, 8 hops max, press Ctrl+C to stop
 1  192.168.1.1   37.195 ms  28.602 ms  36.704 ms
 2  20.0.0.2   80.154 ms  117.582 ms  110.090 ms
 3   *  *  *
 4  *192.168.3.2   77.879 ms (ICMP type:3, code:3, Destination port unreachable)
```

The above route can be visualized as (shown by red lines)



The given outputs are obvious as the best possible routes from PC1 to PC2 and also for PC1 to PC3

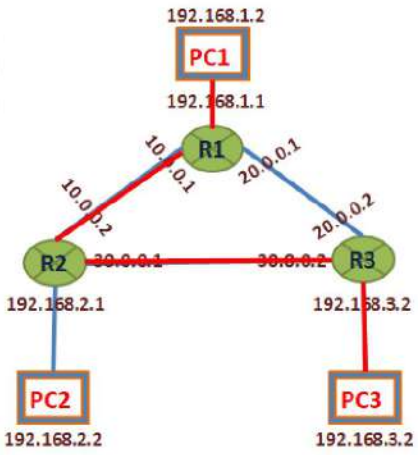
Step 6: Setting Policy Based Routing (PBR) in Router1

```
R1#
R1#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#
R1(config)#access-list 10 permit 192.168.1.2
R1(config)#route-map SMILE permit 10
R1(config-route-map)#
R1(config-route-map)#match ip address 10
R1(config-route-map)#set ip next-hop 10.0.0.2
R1(config-route-map)#exit
R1(config)#
R1(config)#interface fastEthernet 0/0
R1(config-if)#
R1(config-if)#ip policy route-map SMILE
R1(config-if)#exit
R1(config)#
```

Step 7: Verifying the route from PC1 to PC3

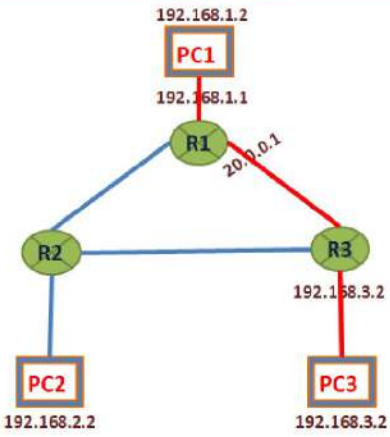
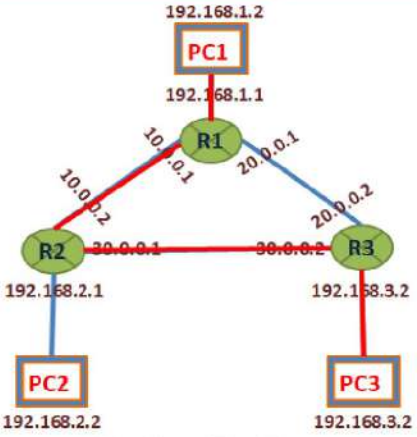
```
PC1> trace 192.168.3.2
trace to 192.168.3.2, 8 hops max, press Ctrl+C to stop
 1  192.168.1.1   9.682 ms  30.054 ms  20.347 ms
 2  10.0.0.2   79.511 ms  70.173 ms  89.750 ms
 3  30.0.0.2   100.528 ms  80.136 ms  109.753 ms
 4   *  *  *
 5  *192.168.3.2   160.514 ms (ICMP type:3, code:3, Destination port unreachable)
```

The above route can be visualized as (shown by red lines)



The packets from PC1 to PC3 are not forwarded directly from R1 to R3 but they take a long route due to the Policy Based Routing, they are forwarded from PC1 to R1 and from R1 to R2 and then from R2 to R3 and finally to PC3

The following gives the comparison between routes from PC1 to PC3 without and with PBR

Without PBR	With PBR
	
The routing path PC1-R1-R3-PC3	The routing path PC1-R1-R2-R3-PC3

For video demonstration of the given practical click on the link below or scan the QR-code

https://youtu.be/3M1sUhMg6YU

IP SLA and Remote SPAN in a Campus Environment

Aim: To demonstrate the configuration and use of IP SLA and Remote SPAN (RSPAN) in a campus environment for performance monitoring and traffic analysis.

Theory: 1. IP SLA (IP Service Level Agreement)
IP SLA is a Cisco feature used to measure network performance and monitor service levels by generating synthetic traffic between network devices. It helps in:

Measuring latency, jitter, packet loss, and availability

Verifying QoS (Quality of Service)

Troubleshooting network paths

Tracking reachability to trigger dynamic routing changes

How it works:
IP SLA sends test traffic (like ICMP, UDP, or HTTP) to a target IP address.

It records statistics like round-trip time and availability.

You can schedule these tests and track trends over time.

Remote SPAN (RSPAN)
Remote SPAN extends the Switched Port Analyzer (SPAN) feature to span traffic across multiple switches in a campus environment. It is used to:

Monitor traffic on remote switches

Capture packets from a source port/VLAN to a central analyzer

Troubleshoot performance or security issues centrally

How it works:
You configure a special RSPAN VLAN to carry mirrored traffic.

Source switches send mirrored traffic to this VLAN.

The destination switch receives mirrored traffic on an RSPAN port where an analyzer or packet capture tool is connected.

Code:
1. IP SLA Configuration Example (Cisco CLI)

```
conf t
ip sla 1
icmp-echo 192.168.1.1
frequency 10
exit
ip sla schedule 1 life forever start-time now
track 1 ip sla 1 reachability
```
This config pings 192.168.1.1 every 10 seconds and tracks its availability.

2. Remote SPAN Configuration Example
On the Source Switch:

```
conf t
vlan 999
 remote-span
exit
monitor session 1 source interface FastEthernet0/1
monitor session 1 destination remote vlan 999
```

On the Destination Switch:

```
conf t
vlan 999
remote-span
exit
monitor session 1 source remote vlan 999
monitor session 1 destination interface FastEthernet0/24
```

Connect a Wireshark PC or analyzer tool to Fa0/24 on the destination switch to capture traffic

Conclusion:
In this practical, we implemented IP SLA to simulate and monitor network performance metrics and configured Remote SPAN to mirror traffic across switches for remote analysis. These tools are essential in campus networks for proactive monitoring, troubleshooting, and ensuring service reliability, especially in large distributed environments.

Inter-VLAN Routing

Aim: To Study the Inter-VLAN routing on a layer-3 switch

Theory: VLANs are used to segment switched Layer 2 networks for a variety of reasons. Regardless of the reason, hosts in one VLAN cannot communicate with hosts in another VLAN unless there is a router or a Layer 3 switch to provide routing services.

Inter-VLAN routing is the process of forwarding network traffic from one VLAN to another VLAN.
There are three inter-VLAN routing options:

1) Legacy Inter-VLAN routing: This is a legacy solution. It does not scale well.
2) Router-on-a-Stick: This is an acceptable solution for a small- to medium-sized network.
3) Layer 3 switch using switched virtual interfaces (SVIs): This is the most scalable solution for medium to large organizations.

Inter-VLAN Routing on a Layer 3 Switch

The modern method of performing inter-VLAN routing is to use Layer 3 switches and switched virtual interfaces (SVI). An SVI is a virtual interface that is configured on a Layer 3 switch

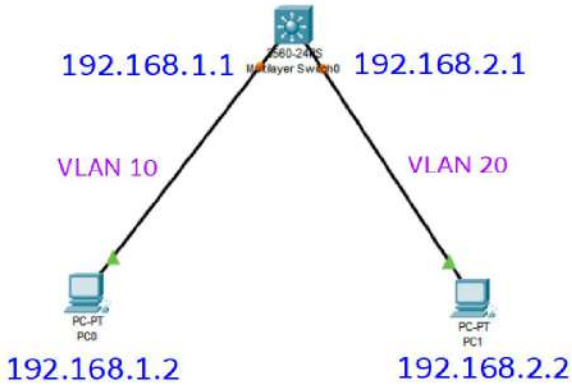
Inter-VLAN SVIs are created the same way that the management VLAN interface is configured. The SVI is created for a VLAN that exists on the switch. Although virtual, the SVI performs the same functions for the VLAN as a router interface would. Specifically, it provides Layer 3 processing for packets that are sent to or from all switch ports associated with that VLAN.

The following are advantages of using Layer 3 switches for inter-VLAN routing:

1) They are much faster than router-on-a-stick because everything is hardware switched and routed.
2) There is no need for external links from the switch to the router for routing.
3) They are not limited to one link because Layer 2 Ether Channels can be used as trunk links between the switches to increase bandwidth.
4) Latency is much lower because data does not need to leave the switch to be routed to a different network.
5) They are more commonly deployed in a campus LAN than routers.

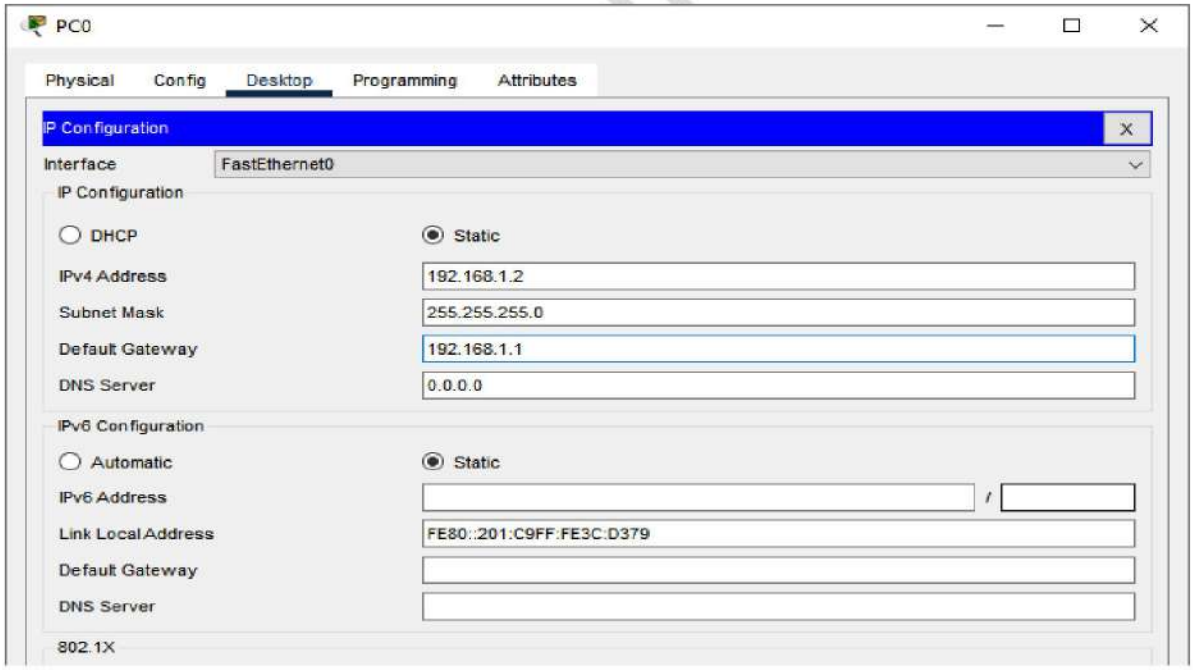
The only disadvantage is that Layer 3 switches are more expensive than Layer 2 switches, but they can be less expensive than a separate Layer 2 switch and router.

We use the following topology to study Inter-VLAN routing

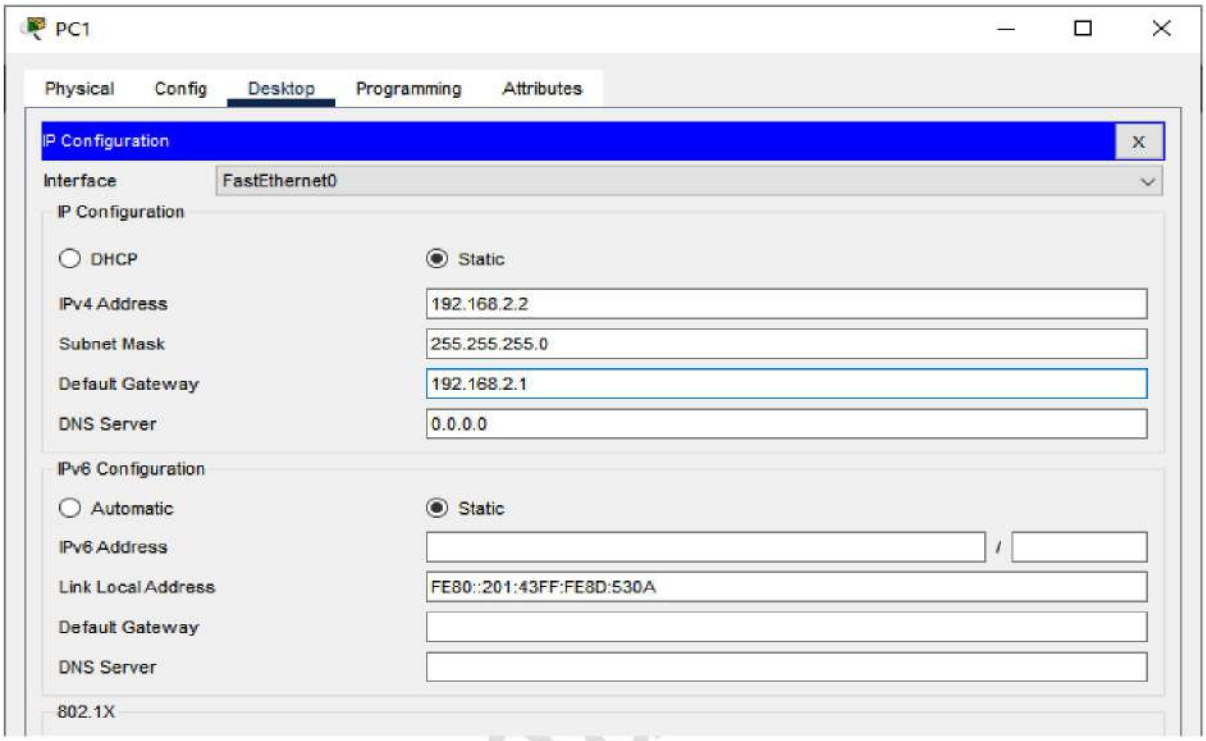


We Configure the IP addresses on the PC

PC0 :



PC1:



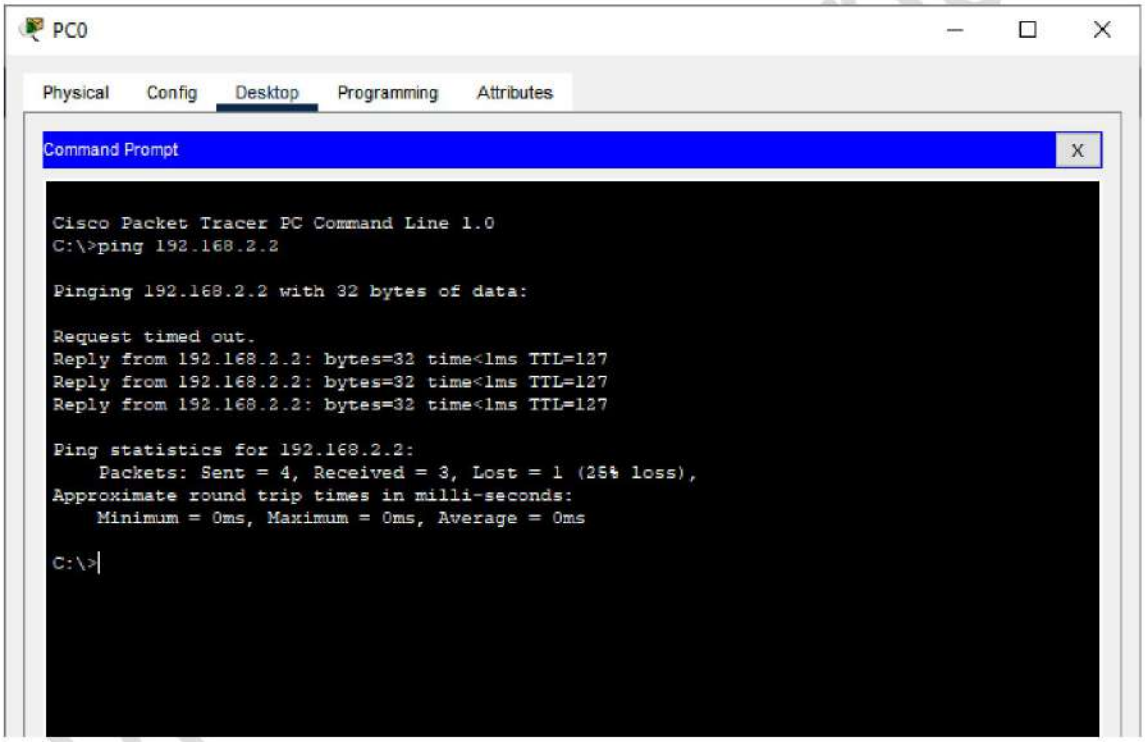
Now we configure the Multilayer switch using the following command in the CLI mode

```
Switch>enable
Switch#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)#vlan 10
Switch(config-vlan)#name smile
Switch(config-vlan)#exit
Switch(config)#
Switch(config)#vlan 20
Switch(config-vlan)#name cisco
Switch(config-vlan)#exit
Switch(config)#
Switch(config)#interface vlan 10
Switch(config-if)#
Switch(config-if)#ip address 192.168.1.1 255.255.255.0
Switch(config-if)#no shutdown
Switch(config-if)#exit
Switch(config)#
```

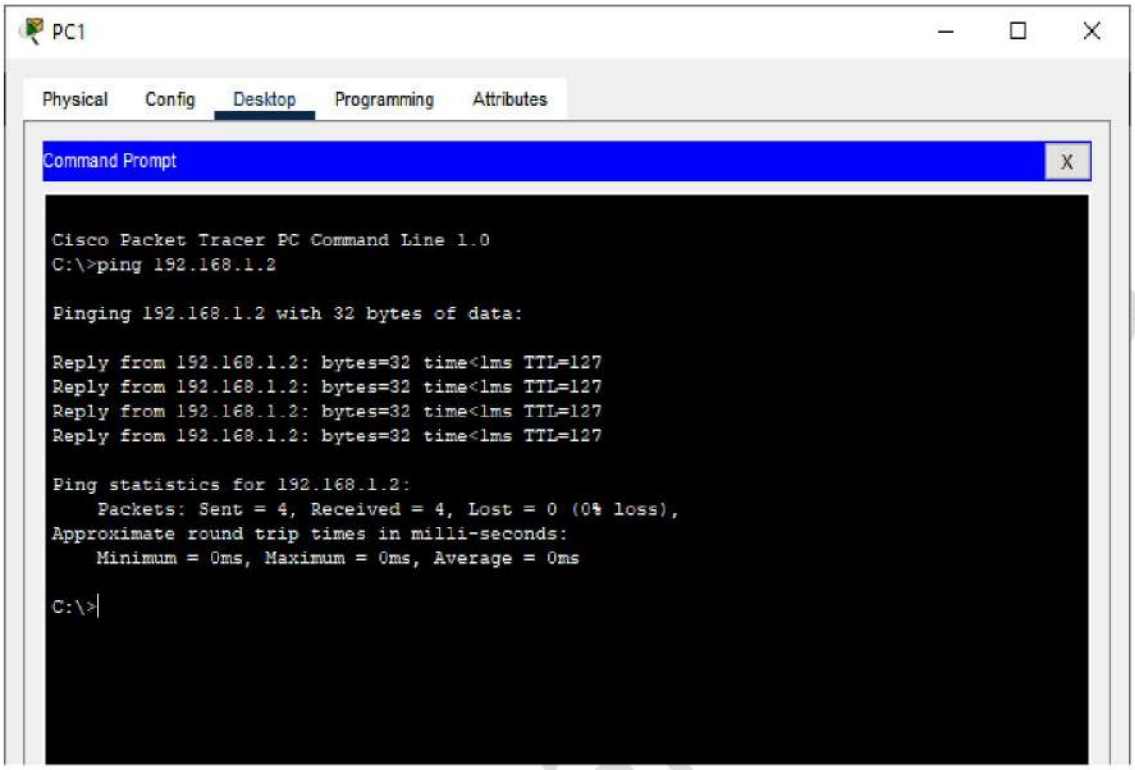
```
Switch(config)#interface vlan 20
Switch(config-if)#
Switch(config-if)#ip address 192.168.2.1 255.255.255.0
Switch(config-if)#no shutdown
Switch(config-if)#exit
Switch(config)#exit
Switch#
Switch#show ip interface brief
Switch(config)#ip routing
```

Output:

Now we ping PC1 from PC0 to check the connectivity



Similarly ping PC0 from PC1



Both the Pings are successful; hence the Inter-VLAN routing has been configured and verified

For Video demonstration of the given practical click on the link or scan the QR-code

https://youtu.be/Vaq9mgTM6-8?si=QcVxHVv5TBLO2t6b

Simulating OpenFlow Using Mininet

Aim: To simulate a Software Defined Networking (SDN) environment using Mininet and observe the working of the OpenFlow protocol between a switch and a controller.

Theory: Software Defined Networking (SDN) is a modern network architecture that separates the control plane (decision-making) from the data plane (packet forwarding). This is done using a central controller that communicates with network switches via a protocol like OpenFlow.

Mininet is a lightweight network emulator that creates a virtual network with hosts, switches, and controllers. It uses Open vSwitch (OVS) to simulate software-defined switches.

In an OpenFlow network:

Switches send unknown packet events to the controller.

The controller responds with flow rules that get installed into the switch's flow table.

Future packets matching those rules are forwarded directly by the switch without involving the controller again.

This practical demonstrates the basic operation of OpenFlow using a simple topology in Mininet.

Understanding Software Defined Networking (SDN)
Traditional networks are often rigid and hardware-driven, where each switch or router has its own control logic. In contrast, Software Defined Networking (SDN) is a modern approach where the control plane (decision-making) is separated from the data plane (packet forwarding). This architecture gives network administrators centralized control and greater flexibility.

In SDN, the controller is the brain of the network. It decides how traffic should flow and instructs the switches accordingly using a standardized protocol — typically OpenFlow.

What is OpenFlow?
OpenFlow is the first standard communications interface defined between the control and forwarding layers of an SDN architecture. It allows an external controller to interact with a switch's forwarding plane — inserting, modifying, and deleting flow rules dynamically.

Each OpenFlow-enabled switch contains:

A flow table with rules that match packets and specify actions (e.g., forward, drop, send to controller).

A secure channel to communicate with the SDN controller.

An OpenFlow protocol to manage flow table entries.

Mininet: A Lightweight SDN Testbed
Mininet is a network emulator that runs a collection of virtual hosts, switches, links, and controllers on a single Linux machine. It supports:

Rapid prototyping of SDN applications

Testing controller-switch interactions

Simulating various topologies using Python or CLI

Mininet uses Open vSwitch (OVS) as the virtual switch, which is OpenFlow-compatible and supports many SDN features.

How It All Works in Practice
When a host sends a packet:

The switch checks its flow table.

If no matching rule exists, it forwards the packet to the controller using OpenFlow.

The controller analyzes the packet and installs a flow rule in the switch.

The switch then forwards future packets directly, reducing latency and controller load.

This behavior mimics real-world programmable networks used in data centers, enterprise networks, and cloud infrastructure.

Educational Value
Simulating OpenFlow using Mininet gives students hands-on experience with SDN principles:

Dynamic network control

Traffic engineering

Protocol interaction between switches and controllers

Real-time flow table inspection

This prepares students to work with cutting-edge network technologies like SD-WAN, cloud networking, 5G, and network automation platforms.

Commands:

🔷 1. Start Mininet with 3 Hosts and 1 Switch
mn --topo single,3 --mac --switch ovsk --controller remote

🔷 2. Start the POX Controller (in a new terminal)
cd ~/pox
./pox.py forwarding.l2_learning
(Alternatively, use Mininet's default controller with --controller=default if POX is not installed)

🔷 3. Run Basic Connectivity Tests from Mininet CLI
mininet> pingall

🔷 4. Test Bandwidth Between Two Hosts
mininet> h1 iperf -s &
mininet> h2 iperf -c h1

🔷 5. View Flow Table Entries on the Switch
mininet> sh ovs-ofctl dump-flows s1

🔷 6. Exit Mininet
mininet> exit

Conclusion: In this practical, we successfully simulated an SDN network using Mininet. We observed how the OpenFlow protocol enables switches to communicate with a central controller to manage flow rules dynamically. This demonstrates the key SDN principle of separating the control and data planes, enabling more flexible and programmable network management.

For Video demonstration some part of the given practical click on the link or scan the QR-code

https://youtu.be/hSEN_YAbUZY?si=-yEWFWVT1m25wG28