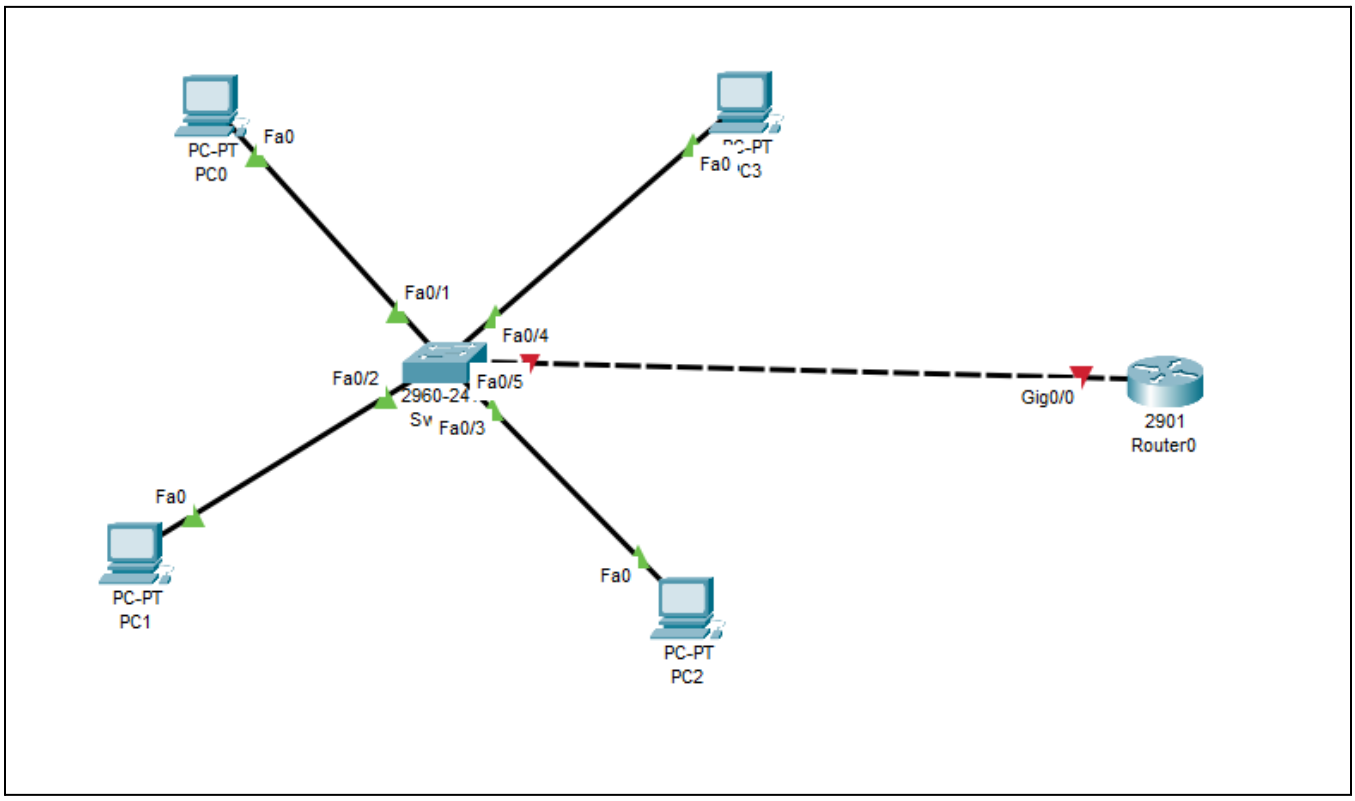
INDEX

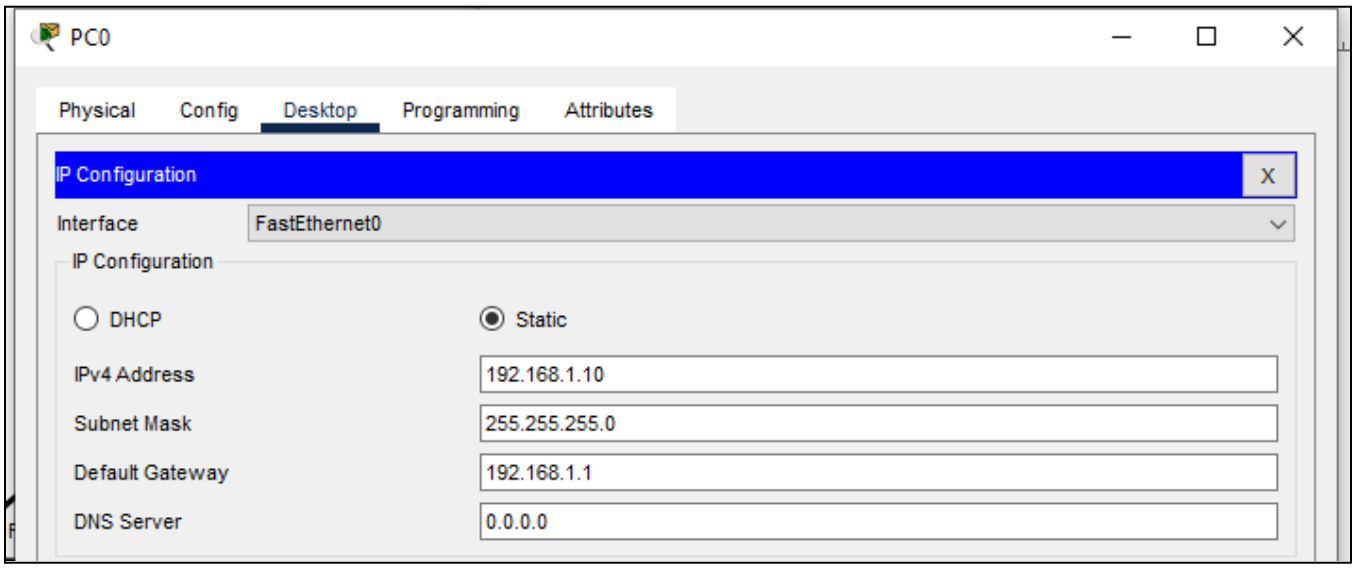
Sr. No.	Practical	Date	Sign
1	Implement Inter VLAN Routing		
2	OSPF		
3	Dynamic NAT controller		
4	Implementing SDN using mininet		
5	Emulating different network topologies (simple, linear, tree) using virtual box and Mininet		
6	Access Mininet VM remotely		
7	Configuring switch in Mininet to implement firewall using POX controller		
8	Simulating network & report generation using NET SIM		

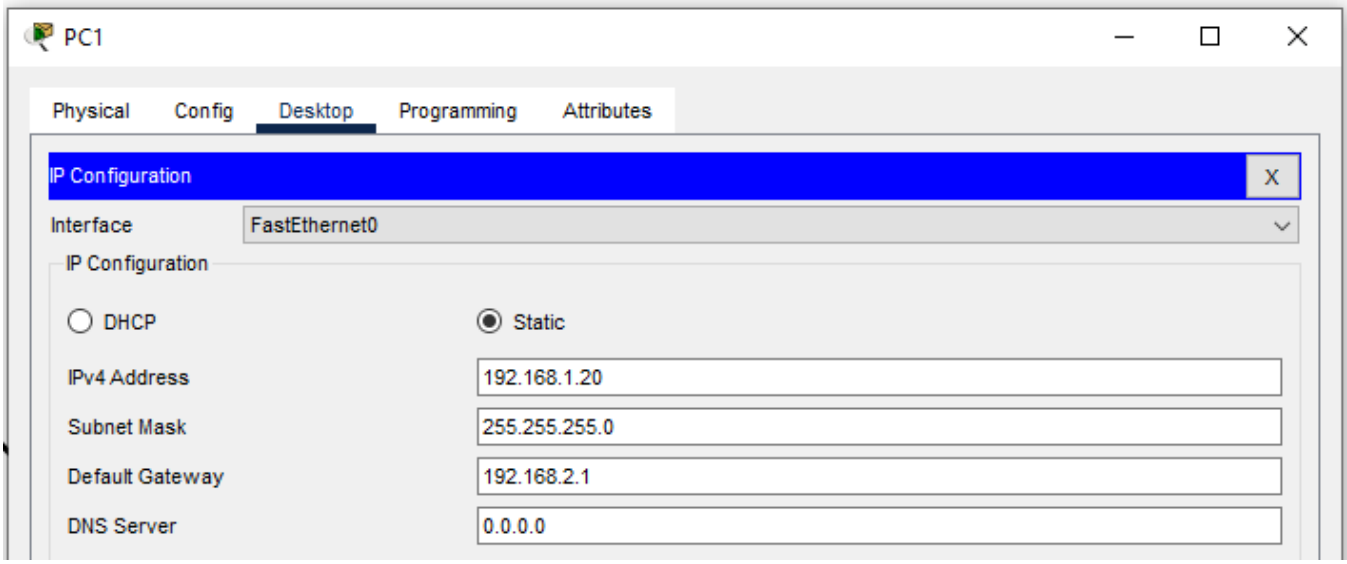
SDN Practical 1

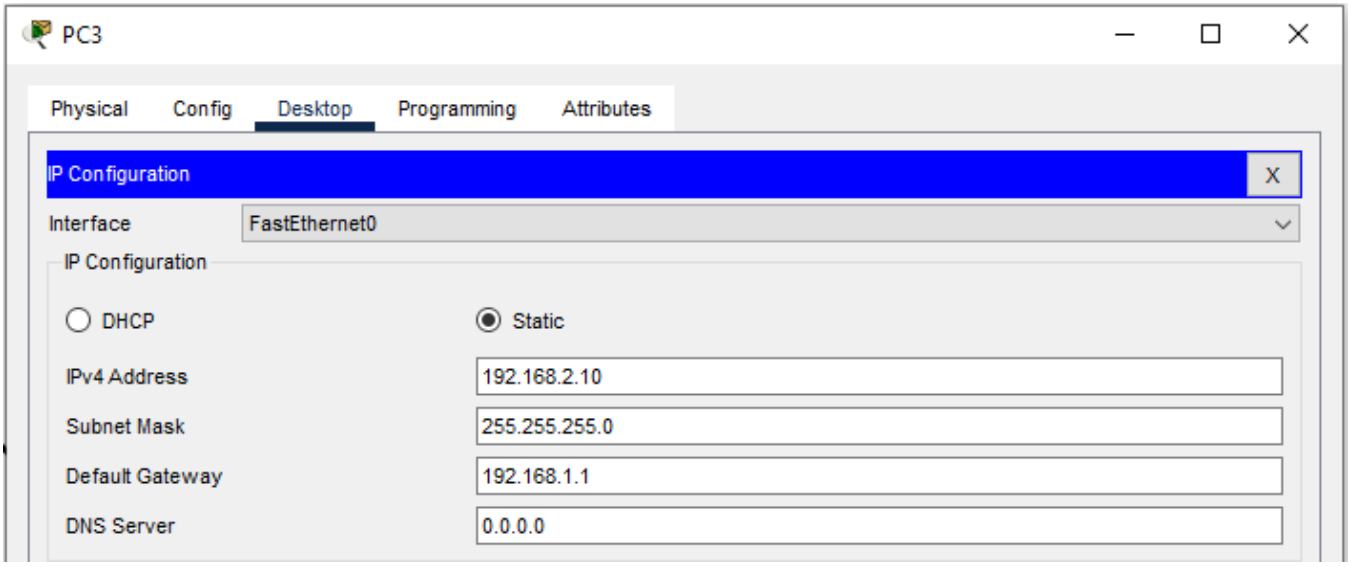
Step1: Create The topology

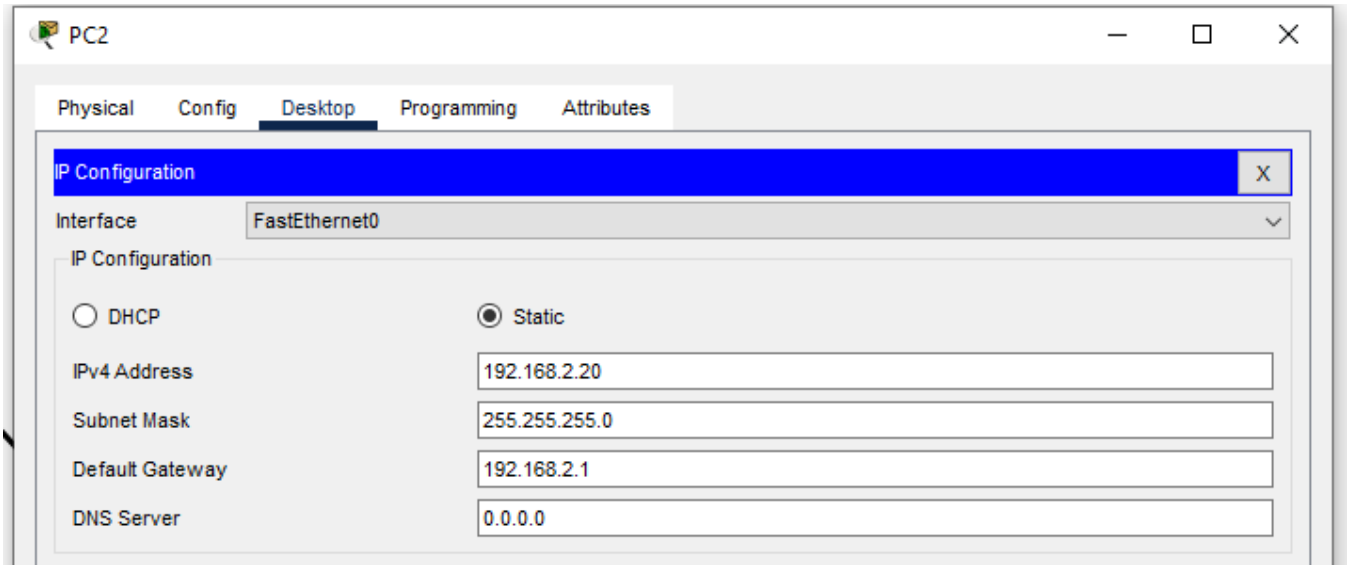


Now assign the ip address, desktop -> ip address




PC1
Physical
Config
Desktop
Programming
Attributes
IP Configuration
X
Interface
FastEthernet0
IP Configuration
DHCP
Static
IPv4 Address
192.168.1.20
Subnet Mask
255.255.255.0
Default Gateway
192.168.2.1
DNS Server
0.0.0.0


PC3
Physical
Config
Desktop
Programming
Attributes
IP Configuration
X
Interface
FastEthernet0
IP Configuration
DHCP
Static
IPv4 Address
192.168.2.10
Subnet Mask
255.255.255.0
Default Gateway
192.168.1.1
DNS Server
0.0.0.0


PC2
Physical
Config
Desktop
Programming
Attributes
IP Configuration
X
Interface
FastEthernet0
IP Configuration
DHCP
Static
IPv4 Address
192.168.2.20
Subnet Mask
255.255.255.0
Default Gateway
192.168.2.1
DNS Server
0.0.0.0

Step 2 - create the vlans

Go to switch, click on switch, CLI mode, we have to enable the switch and once it is enabled w will get hash so we need to get enable and conf

```
Switch>enable
Switch#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#vlan 10
Switch(config-vlan)#name student
Switch(config-vlan)#exit
Switch(config)#
```

Exit and go to vlan 20

```
Switch(config-vlan)#name staff
Switch(config-vlan)#exit
```

Step 3 - Assign the ports to VLAN

We are having 4 ports
Giving to first pc, fa0/1

Now give access to that port with the command switchport mode access
Now we have the access to the port

```
Switch(config)#int fa0/1
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 10
Switch(config-if)#exit
```

Now go to the next port fa0/2
They still can't communicate, because we have created the security – VLAN
Similarly put the things another ports, fa0/3 ko 10 me put karna h and fa0/4 ko 20 me

```
Switch(config)#int fa0/1
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 10
Switch(config-if)#exit
Switch(config)#
Switch(config)#int fa0/2
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 20
Switch(config-if)#exit
Switch(config)#int fa0/3
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 10
Switch(config-if)#exit
Switch(config)#int fa0/4
Switch(config-if)#switchport mode access
Switch(config-if)#switchport access vlan 20
Switch(config-if)#exit
```

If its is LAN devices will talk to each other but when VLAN devices will not talk , so we need to trunk…
And then save it, by running command “do write”

```
Switch(config)#int fa0/5
Switch(config-if)#switchport mode trunk
Switch(config-if)#do write
Building configuration...
[OK]
Switch(config-if)#
```

Step 4 - Configure the Router

Click on router, go to CLI, use enable and go to config mode

```
Router#en
Router#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int g0/0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0, changed state to up

Router(config-if)#
```

```
Router(config-if)#int g0/0.10
Router(config-subif)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0.10, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.10, changed state
up

Router(config-subif)#
```

This would allow the traffic to go

```
Router(config-subif)#encapsulation dot1q 10
Router(config-subif)#ip add 192.168.1.1
% Incomplete command.
```

Because we need to do unmasking

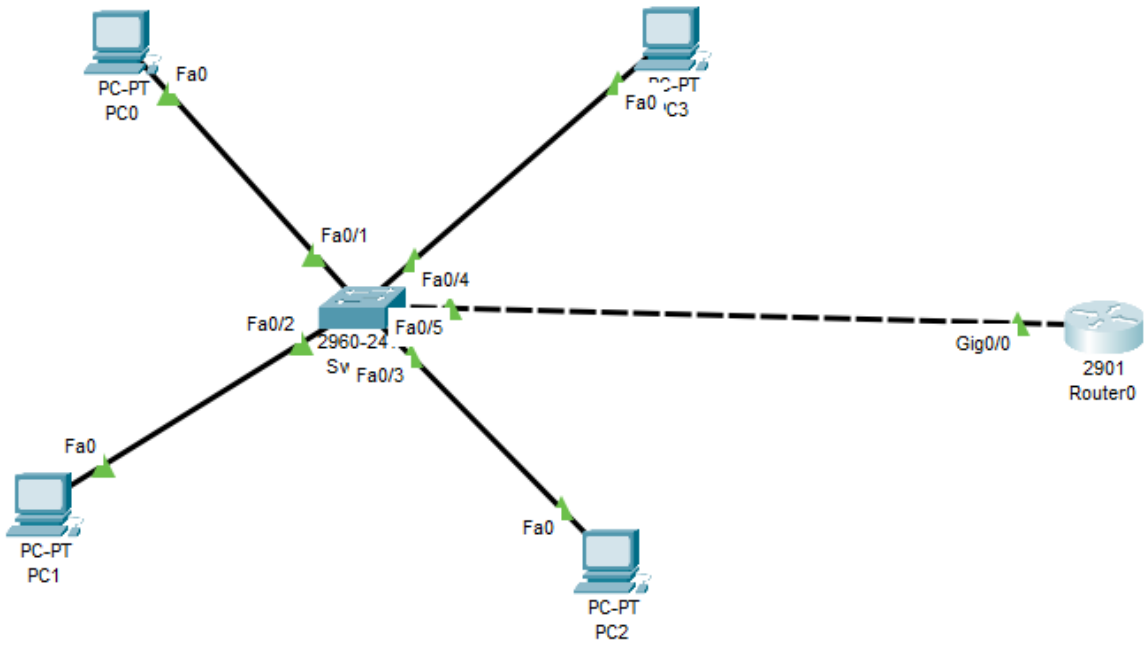
```
Router(config-subif)#encapsulation dotlq 10
Router(config-subif)#ip add 192.168.1.1 255.255.255.0
```

Now do the same

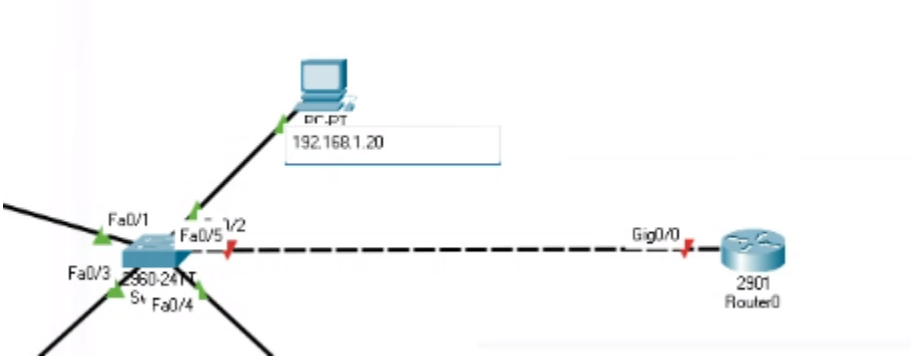
And use 0.20

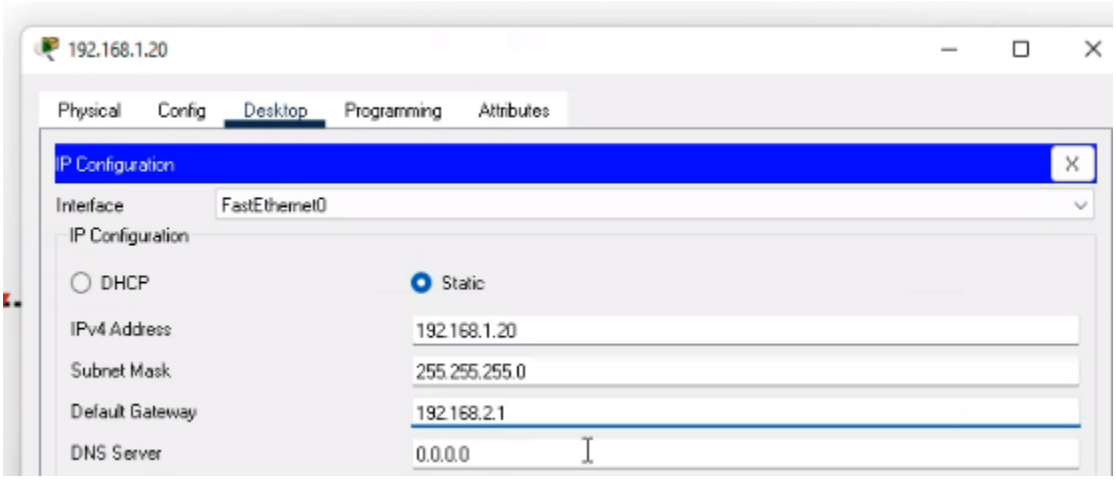
```
Router(config-subif)#
Router(config-subif)#encapsulation dotlq 10
Router(config-subif)#ip add 192.168.1.1 255.255.255.0
Router(config-subif)#
Router(config-subif)#encapsulation dotlq 20
Router(config-subif)#ip add 192.168.2.1 255.255.255.0
Router(config-subif)#exit
Router(config)#do write
Building configuration...
[OK]
```

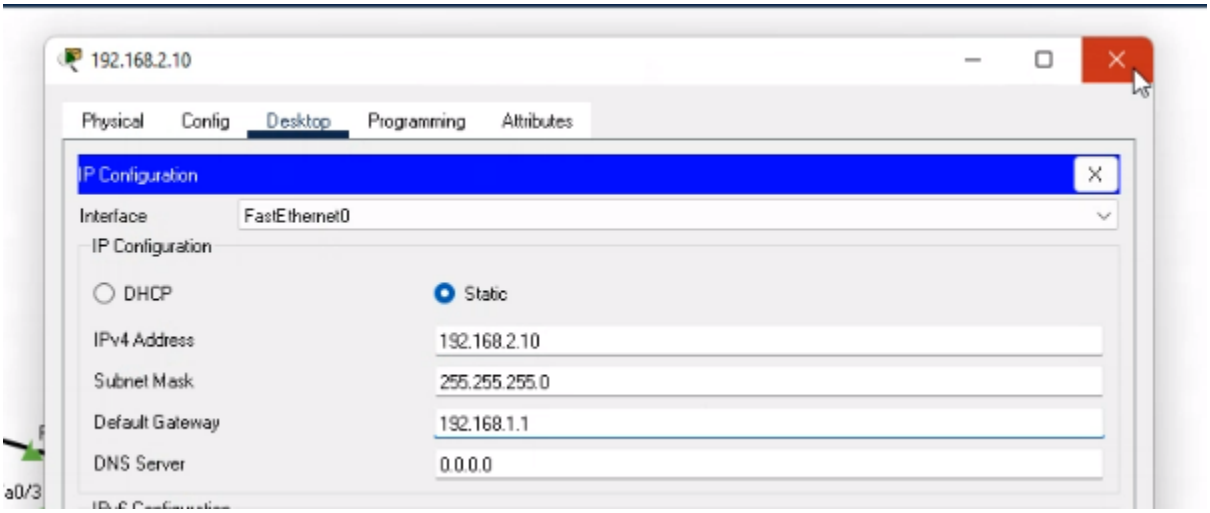
Now check connectivity using ping command



Repeat for everyone:







Step 2 - create the vlans

Go to switch, click on switch, CLI mode, we have to enable the switch and once it is enabled w will get hash so we need to get enable and conf

```
Switch>enable
Switch#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
```

```
Switch>enable
Switch#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#vlan 10
Switch(config-vlan)#name student
Switch(config-vlan)#exit
Switch(config)#
```

Exit and go to vlan 20

```
Switch>enable
Switch#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#vlan 10
Switch(config-vlan)#name student
Switch(config-vlan)#exit
Switch(config)#vlan 20
Switch(config-vlan)#name staff
Switch(config-vlan)#exit
Switch(config)#
```

Step 3 - Assign the ports to VLAN

We are having 4 ports
Giving to first pc, fa0/1

Now give access to that port with the command switchport mode access
Now we have the access to the port

```
Switch(config)#int fa0/1
Switch(config-if)#switchport mode access
Switch(config-if)#switchport  access vlan 10
Switch(config-if)#exit
Switch(config)#
```

Now go to the next port fa0/2

```
Switch(config)#int fa0/2
Switch(config-if)#switchport mode access
Switch(config-if)#switchport  access vlan 20
Switch(config-if)#
```

They still can't communicate, because we have created the security – VLAN
Similarly put the things another ports, fa0/3 ko 10 me put karna h and fa0/4 ko 20 me

```
Switch(config)#int fa0/1
Switch(config-if)#switchport mode access
Switch(config-if)#switchport  access vlan 10
Switch(config-if)#exit
Switch(config)#int fa0/2
Switch(config-if)#switchport mode access
Switch(config-if)#switchport  access vlan 20
Switch(config-if)#exit
Switch(config)#int fa0/3
Switch(config-if)#switchport mode access
Switch(config-if)#switchport  access vlan 10
Switch(config-if)#exit
Switch(config)#int fa0/4
Switch(config-if)#switchport mode access
Switch(config-if)#switchport  access vlan 20
Switch(config-if)#exit
```

If its is LAN devices will talk to each other but when VLAN devices will not talk , so we need to trunk…
And then save it, by running command “do write”

```
Switch(config)#int fa0/4
Switch(config-if)#switchport mode access
Switch(config-if)#switchport  access vlan 20
Switch(config-if)#exit
Switch(config)#int fa0/5
Switch(config-if)#switchport mode trunk
Switch(config-if)#do write
Building configuration...
[OK]
Switch(config-if)#
```

Step 4 - Configure the Router
Click on router, go to CLI, use enable and go to config mode

```
Router>
Router>en
Router#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int g0/0
```

And start it by using command”no shutdown”

```
Router>
Router>en
Router#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int g0/0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0, changed state to up

Router(config-if)#in
```

Int - interface

```
Router(config-if)#int g0/0.10
Router(config-subif)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0.10, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.10, changed state to
up
```

This would allow traffic to go

To encapsulate the traffic – encapsulation dot1q 10

```
Router(config-subif)#encapsulation dot1q 10
Router(config-subif)#ip add 192.168.1.1
```

Using masking

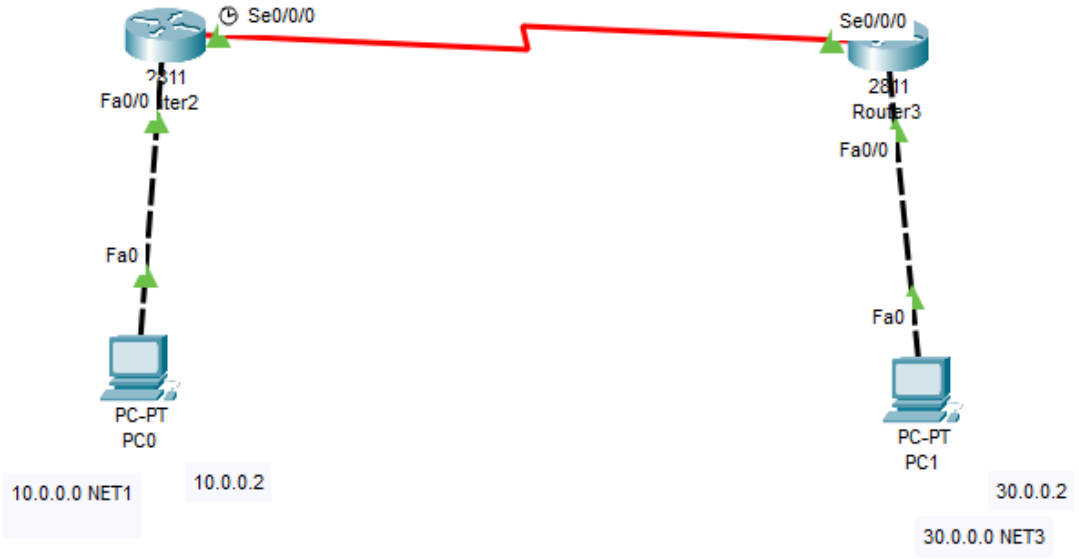
```
Router(config-subif)#encapsulation dot1q 10
Router(config-subif)#ip add 192.168.1.1 255.255.255.0
```

Now do the same
And use 0.20

```
Router(config-if)#int g0/0.20
Router(config-subif)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0.20, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0.20, changed state to
up

Router(config-subif)#encapsulation dot1q 20
Router(config-subif)#ip add 192.168.2.1 255.255.255.0
Router(config-subif)#exit
Router(config)#do write
Building configuration...
[OK]
Router(config)#e
```

Check connectivity using ping command.

PRACTICAL 2



ROUTER 1

Router>en
Router#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int fa0/0
Router(config-if)#ip add 10.0.0.1 255.0.0.0
Router(config-if)#no shut
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)#int serial 0/0/0/
^
% Invalid input detected at '^' marker.

Router(config-if)#exit
Router(config)#int serial 0/0/0
Router(config-if)#ip add 20.0.0.1 255.0.0.0
Router(config-if)#no shut
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to down
Router(config-if)#exit
Router(config)#
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to up
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
Router#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router ospf 1
Router(config-router)#10.0.0.0 0.255.255.255 area 0
^
% Invalid input detected at '^' marker.

Router(config-router)#network 10.0.0.0 0.255.255.255 area 0
Router(config-router)#network 20.0.0.0 0.255.255.255 area 0
Router(config-router)#
00:13:06: %OSPF-5-ADJCHG: Process 1, Nbr 30.0.0.1 on Serial0/0/0 from LOADING to
FULL, Loading Done
```

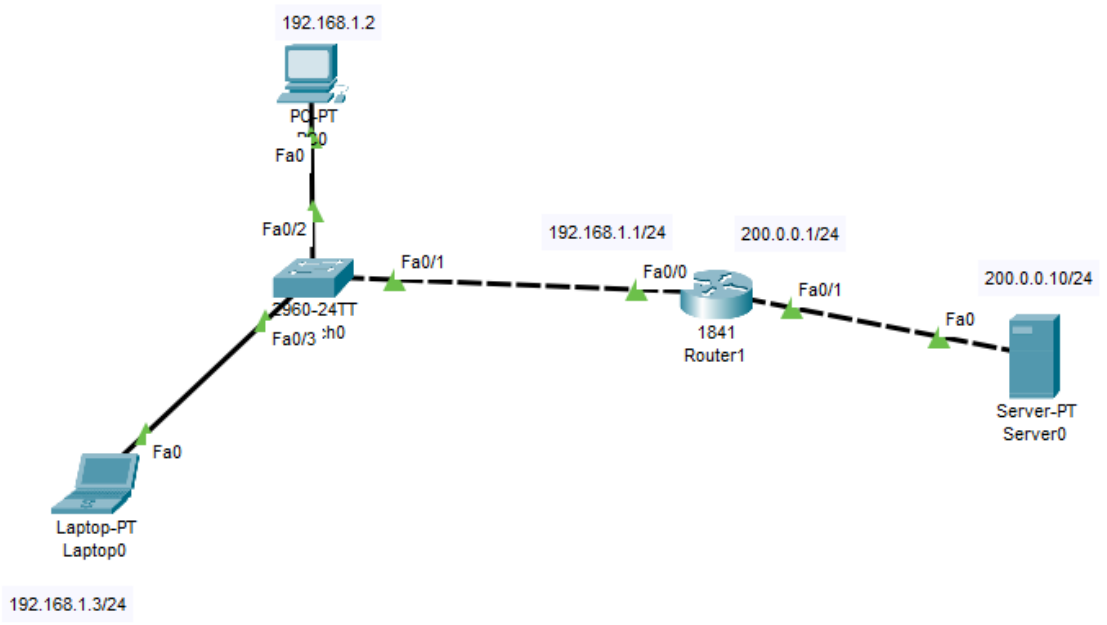
ROUTER 2

```
Router>en
Router#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int fa0/0
Router(config-if)#ip add 30.0.0.1 255.0.0.0
Router(config-if)#no shut
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)#int serial0/0/0
Router(config-if)#ip add 20.0.0.2 255.0.0.0
```

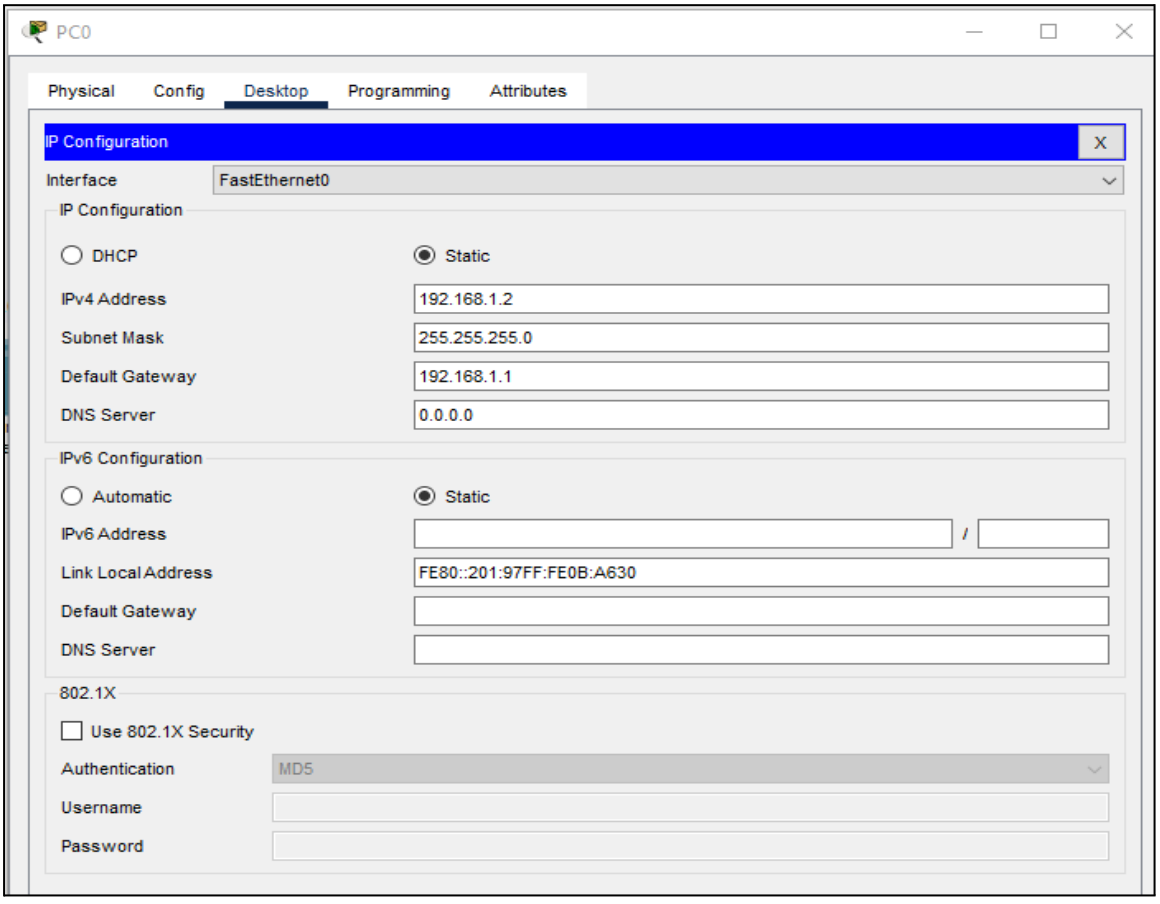
```
Router(config-if)#no shut
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to up
Router(config-if)#exit
Router(config)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to up
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
Router#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router ospf 2
Router(config-router)#network 20.0.0.0 0.255.255.255 area 0
Router(config-router)#network 30.0.0.0 0.255.255.255 area 0
00:13:10: %OSPF-5-ADJCHG: Process 2, Nbr 20.0.0.1 on Serial0/0/0 from LOADING to
FULL, Loading Done
Router(config-router)#
```

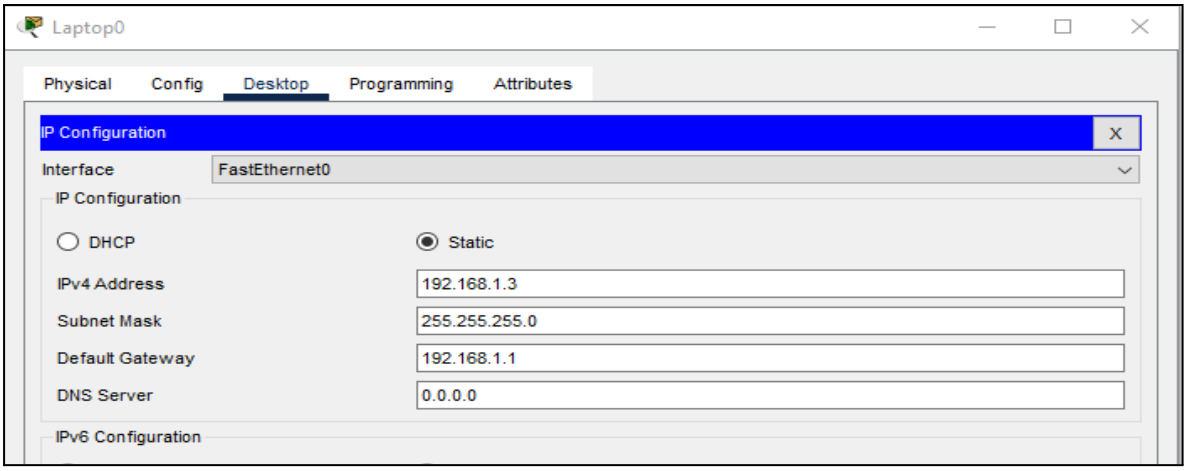
PRACTICAL 3

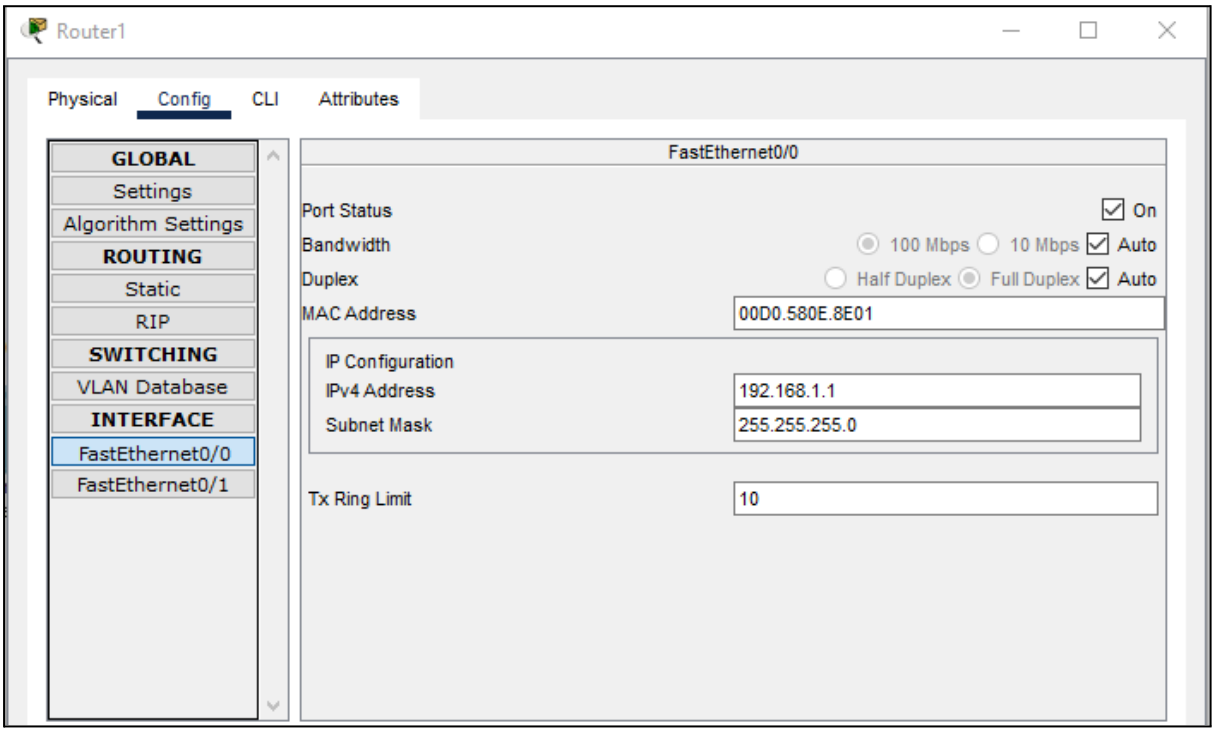
AIM: NAT

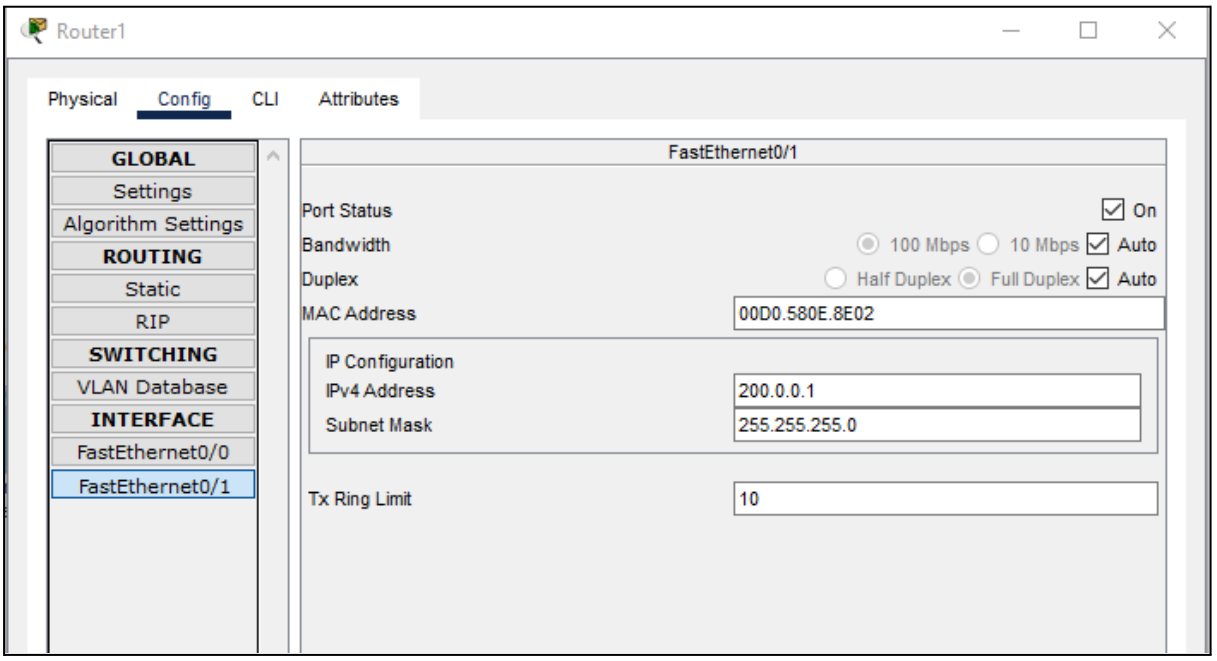


PC Ip add 192.168.1.2/24 Default gateway 192.168.1.1(int fa0/0)
Laptop Ip add 192.168.1.3/24 Default gateway 192.168.1.1(int fa0/0)
Server IP add 200.0.0.10/24 Default gateway 200.0.0.1 (int fa0/1)




Laptop0
Physical
Config
Desktop
Programming
Attributes
IP Configuration
Interface
FastEthernet0
IP Configuration
DHCP
Static
IPv4 Address
192.168.1.3
Subnet Mask
255.255.255.0
Default Gateway
192.168.1.1
DNS Server
0.0.0.0
IPv6 Configuration


Router1
Physical
Config
CLI
Attributes
GLOBAL
Settings
Algorithm Settings
ROUTING
Static
RIP
SWITCHING
VLAN Database
INTERFACE
FastEthernet0/0
FastEthernet0/1
FastEthernet0/0
Port Status
On
Bandwidth
100 Mbps
10 Mbps
Auto
Duplex
Half Duplex
Full Duplex
Auto
MAC Address
00D0.580E.8E01
IP Configuration
IPv4 Address
192.168.1.1
Subnet Mask
255.255.255.0
Tx Ring Limit
10


Router1
Physical
Config
CLI
Attributes
GLOBAL
Settings
Algorithm Settings
ROUTING
Static
RIP
SWITCHING
VLAN Database
INTERFACE
FastEthernet0/0
FastEthernet0/1
FastEthernet0/1
Port Status
On
Bandwidth
100 Mbps
10 Mbps
Auto
Duplex
Half Duplex
Full Duplex
Auto
MAC Address
00D0.580E.8E02
IP Configuration
IPv4 Address
200.0.0.1
Subnet Mask
255.255.255.0
Tx Ring Limit
10

Router1

Physical Config CLI Attributes

IOS Command Line Interface

```
Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface FastEthernet0/0
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
ip address 192.168.1.1 255.255.255.0
Router(config-if)#
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/1
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to up
ip address 200.0.0.1 255.255.255.0
Router(config-if)#ip address 200.0.0.1 255.255.255.0
```

```
Router(config-if)#int fa0/1
Router(config-if)#ip add 200.0.0.1 255.255.255.0
Router(config-if)#no shut
Router(config-if)#int fa0/0
Router(config-if)#ip add 200.0.0.1 255.255.255.0
% 200.0.0.0 overlaps with FastEthernet0/1
Router(config-if)#int fa0/0
Router(config-if)#ip nat inside
Router(config-if)#exit
Router(config)#int fa0/1
Router(config-if)#ip nat outside
Router(config-if)#exit
Router(config)#interface fa0/0
Router(config-if)#access-list 1 permit 192.168.1.0 0.0.0.0.5
```

Server0

Physical Config Services Desktop Programming Attributes

IP Configuration X

IP Configuration

DHCP Static

IPv4 Address 200.0.0.10

Subnet Mask 255.255.255.0

Default Gateway 200.0.0.1

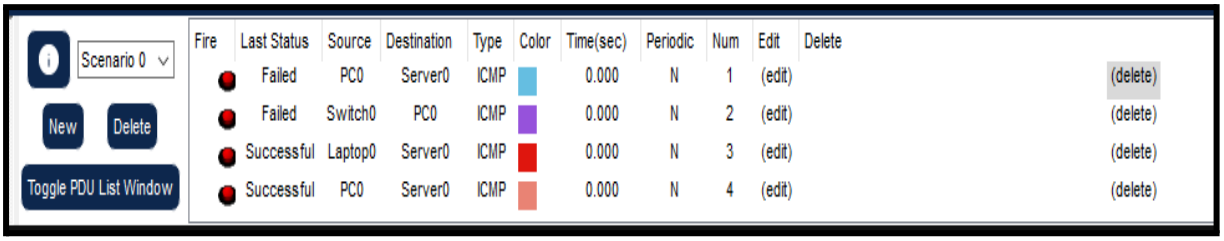
DNS Server 0.0.0.0

IPv6 Configuration

Automatic Static

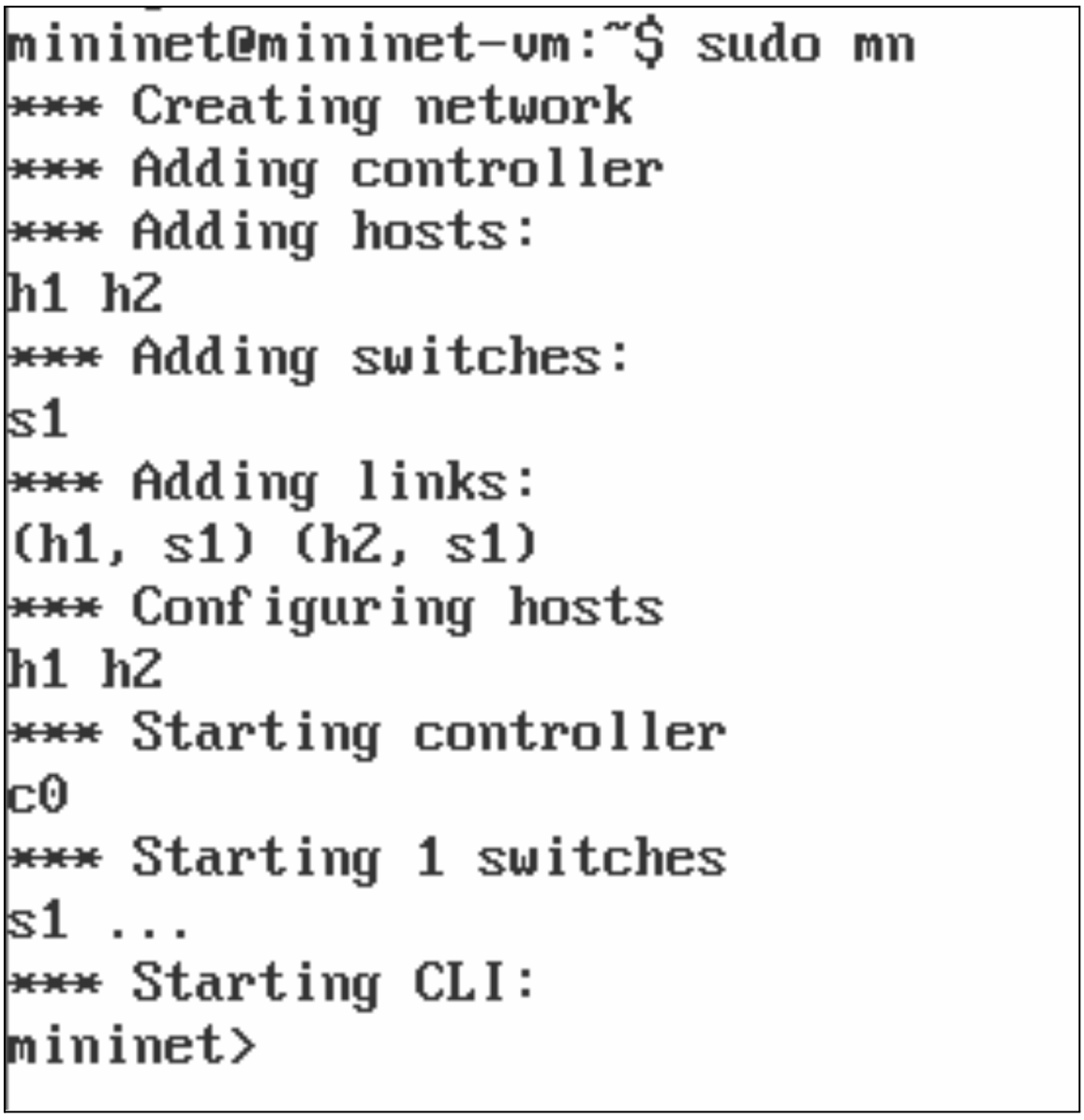
```
Router#show ip nat translations
Pro  Inside global     Inside local       Outside local
Outside global
icmp 155.21.21.10:10   192.168.1.2:10     200.0.0.10:10
200.0.0.10:10
icmp 155.21.21.10:11   192.168.1.2:11     200.0.0.10:11
200.0.0.10:11
icmp 155.21.21.10:12   192.168.1.2:12     200.0.0.10:12
200.0.0.10:12
icmp 155.21.21.10:9    192.168.1.2:9      200.0.0.10:9
200.0.0.10:9
```

OUTPUT:



Practical Mininet

Install and create the environment to execute the following



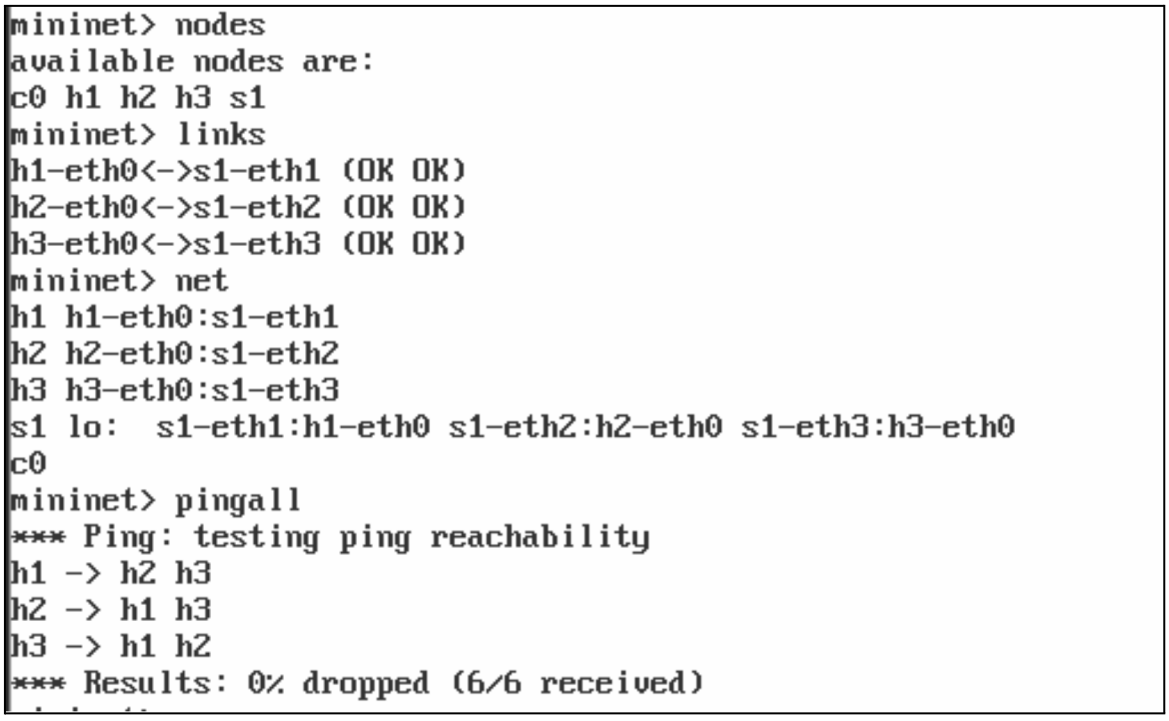
Display nodes

```
mininet> nodes
available nodes are:
c0 h1 h2 s1
```

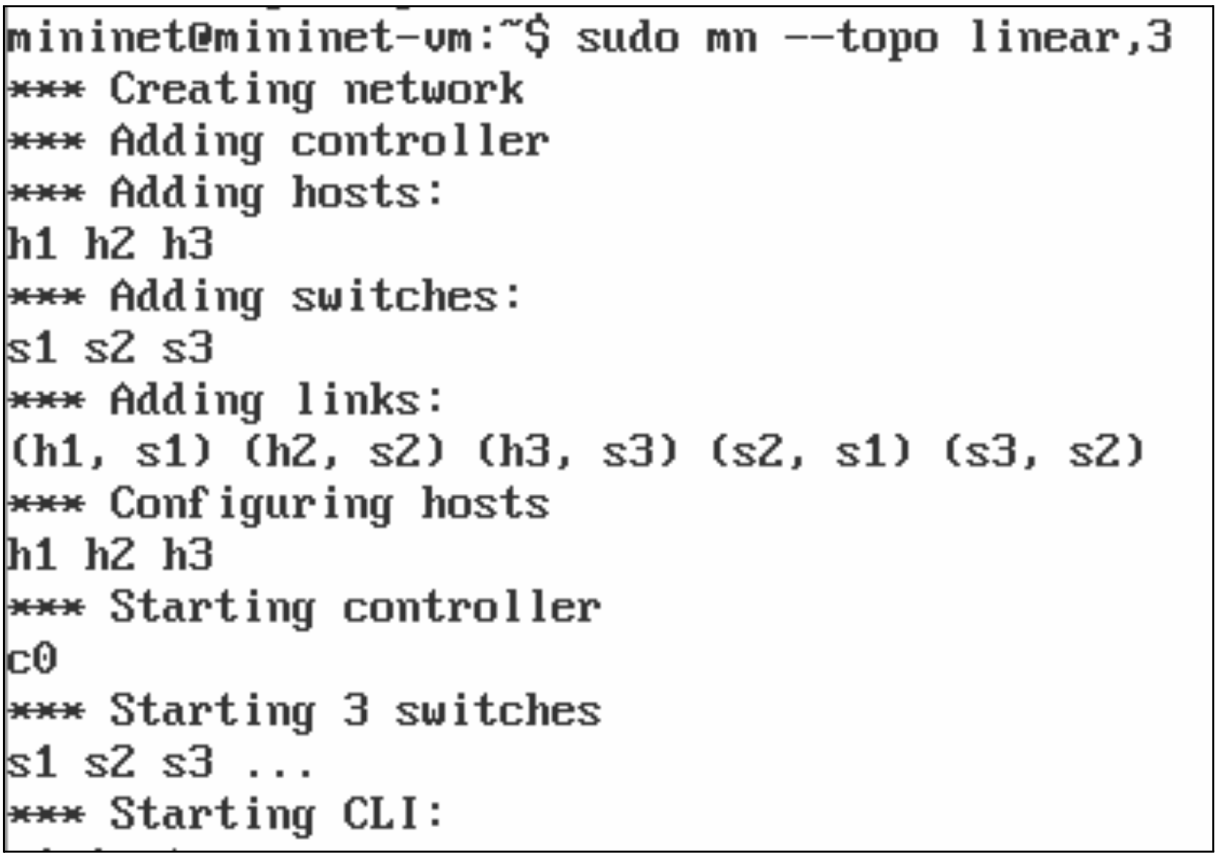
Display topology (all 3 types-simple, linear and tree)

Simple

```
mininet@mininet-vm:~$ sudo mn --topo single,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
```


mininet> nodes
available nodes are:
c0 h1 h2 h3 s1
mininet> links
h1-eth0<->s1-eth1 (OK OK)
h2-eth0<->s1-eth2 (OK OK)
h3-eth0<->s1-eth3 (OK OK)
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
h3 h3-eth0:s1-eth3
s1 lo: s1-eth1:h1-eth0 s1-eth2:h2-eth0 s1-eth3:h3-eth0
c0
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3
h2 -> h1 h3
h3 -> h1 h2
*** Results: 0% dropped (6/6 received)

Linear


mininet@mininet-vm:~$ sudo mn --topo linear,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1 s2 s3
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (s2, s1) (s3, s2)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:

```
mininet> nodes
available nodes are:
c0 h1 h2 h3 s1 s2 s3
mininet> links
h1-eth0<->s1-eth1 (OK OK)
h2-eth0<->s2-eth1 (OK OK)
h3-eth0<->s3-eth1 (OK OK)
s2-eth2<->s1-eth2 (OK OK)
s3-eth2<->s2-eth3 (OK OK)
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s2-eth1
h3 h3-eth0:s3-eth1
s1 lo:  s1-eth1:h1-eth0 s1-eth2:s2-eth2
s2 lo:  s2-eth1:h2-eth0 s2-eth2:s1-eth2 s2-eth3:s3-eth2
s3 lo:  s3-eth1:h3-eth0 s3-eth2:s2-eth3
c0
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3
h2 -> h1 h3
h3 -> h1 h2
*** Results: 0% dropped (6/6 received)
mininet>
```

Tree

```
mininet@mininet-vm:~$ sudo mn --topo tree,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7
*** Adding links:
(s1, s2) (s1, s5) (s2, s3) (s2, s4) (s3, h1) (s3, h2) (s4, h3) (s4, h4) (s5, s6) (s5, s7) (s6, h5) (
s6, h6) (s7, h7) (s7, h8)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8
*** Starting controller
c0
*** Starting 7 switches
s1 s2 s3 s4 s5 s6 s7 ...
*** Starting CLI:
```

```
mininet> nodes
available nodes are:
c0 h1 h2 h3 h4 h5 h6 h7 h8 s1 s2 s3 s4 s5 s6 s7
mininet>
```

```
mininet> links
s1-eth1<->s2-eth3 (OK OK)
s1-eth2<->s5-eth3 (OK OK)
s2-eth1<->s3-eth3 (OK OK)
s2-eth2<->s4-eth3 (OK OK)
s3-eth1<->h1-eth0 (OK OK)
s3-eth2<->h2-eth0 (OK OK)
s4-eth1<->h3-eth0 (OK OK)
s4-eth2<->h4-eth0 (OK OK)
s5-eth1<->s6-eth3 (OK OK)
s5-eth2<->s7-eth3 (OK OK)
s6-eth1<->h5-eth0 (OK OK)
s6-eth2<->h6-eth0 (OK OK)
s7-eth1<->h7-eth0 (OK OK)
s7-eth2<->h8-eth0 (OK OK)
```

```
mininet> net
h1 h1-eth0:s3-eth1
h2 h2-eth0:s3-eth2
h3 h3-eth0:s4-eth1
h4 h4-eth0:s4-eth2
h5 h5-eth0:s6-eth1
h6 h6-eth0:s6-eth2
h7 h7-eth0:s7-eth1
h8 h8-eth0:s7-eth2
s1 lo:  s1-eth1:s2-eth3 s1-eth2:s5-eth3
s2 lo:  s2-eth1:s3-eth3 s2-eth2:s4-eth3 s2-eth3:s1-eth1
s3 lo:  s3-eth1:h1-eth0 s3-eth2:h2-eth0 s3-eth3:s2-eth1
s4 lo:  s4-eth1:h3-eth0 s4-eth2:h4-eth0 s4-eth3:s2-eth2
s5 lo:  s5-eth1:s6-eth3 s5-eth2:s7-eth3 s5-eth3:s1-eth2
s6 lo:  s6-eth1:h5-eth0 s6-eth2:h6-eth0 s6-eth3:s5-eth1
s7 lo:  s7-eth1:h7-eth0 s7-eth2:h8-eth0 s7-eth3:s5-eth2
c0
```

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8
h2 -> h1 h3 h4 h5 h6 h7 h8
h3 -> h1 h2 h4 h5 h6 h7 h8
h4 -> h1 h2 h3 h5 h6 h7 h8
h5 -> h1 h2 h3 h4 h6 h7 h8
h6 -> h1 h2 h3 h4 h5 h7 h8
h7 -> h1 h2 h3 h4 h5 h6 h8
h8 -> h1 h2 h3 h4 h5 h6 h7
*** Results: 0% dropped (56/56 received)
```

Practical 5

1. Implement all 3 types-simple, linear and tree
    a. Show the network interfaces
    b. Test the connectivity

Simple

```
from mininet.net import Mininet
from mininet.topo import Topo
from mininet.node import RemoteController

class SimpleTopology(Topo):
        def build(self):
                s1 = self.addSwitch('s1')
                h1 = self.addHost('h1')
                h2 = self.addHost('h2')
                self.addLink(h1,s1)
                self.addLink(h2, s1)

topo = SimpleTopology()
net = Mininet(topo=topo, controller=lambda name: RemoteController(name, ip='127.0.0.1'))
net.start()
net.pingAll()
net.stop()
```

```
mininet@mininet-vm:~$ sudo python simple.py
Unable to contact the remote controller at 127.0.0.1:6653
Unable to contact the remote controller at 127.0.0.1:6633
Setting remote controller to 127.0.0.1:6653
*** Ping: testing ping reachability
h1 -> X
h2 -> X
*** Results: 100% dropped (0/2 received)
```

Linear

```
from mininet.net import Mininet
from mininet.topo import Topo
from mininet.node import RemoteController

class LinearTopology(Topo):
        def build(self):
                s1 = self.addSwitch('s1')
                s2 = self.addSwitch('s2')
                s3 = self.addSwitch('s3')
                h1 = self.addHost('h1')
                h2 = self.addHost('h2')
                h3 = self.addHost('h3')
                self.addLink(h1, s1)
                self.addLink(h2, s2)
                self.addLink(h3, s3)
                self.addLink(s1, s2)
                self.addLink(s2, s3)

topo = LinearTopology()
net = Mininet(topo=topo, controller=lambda name: RemoteController(name, ip='127.0.0.1'))
net.start()
net.pingAll()
net.stop()
```

```
mininet@mininet-vm:~$ sudo python simple.py
Unable to contact the remote controller at 127.0.0.1:6653
Unable to contact the remote controller at 127.0.0.1:6633
Setting remote controller to 127.0.0.1:6653
*** Ping: testing ping reachability
h1 -> X X
h2 -> X X
h3 -> X X
*** Results: 100% dropped (0/6 received)
```

Tree

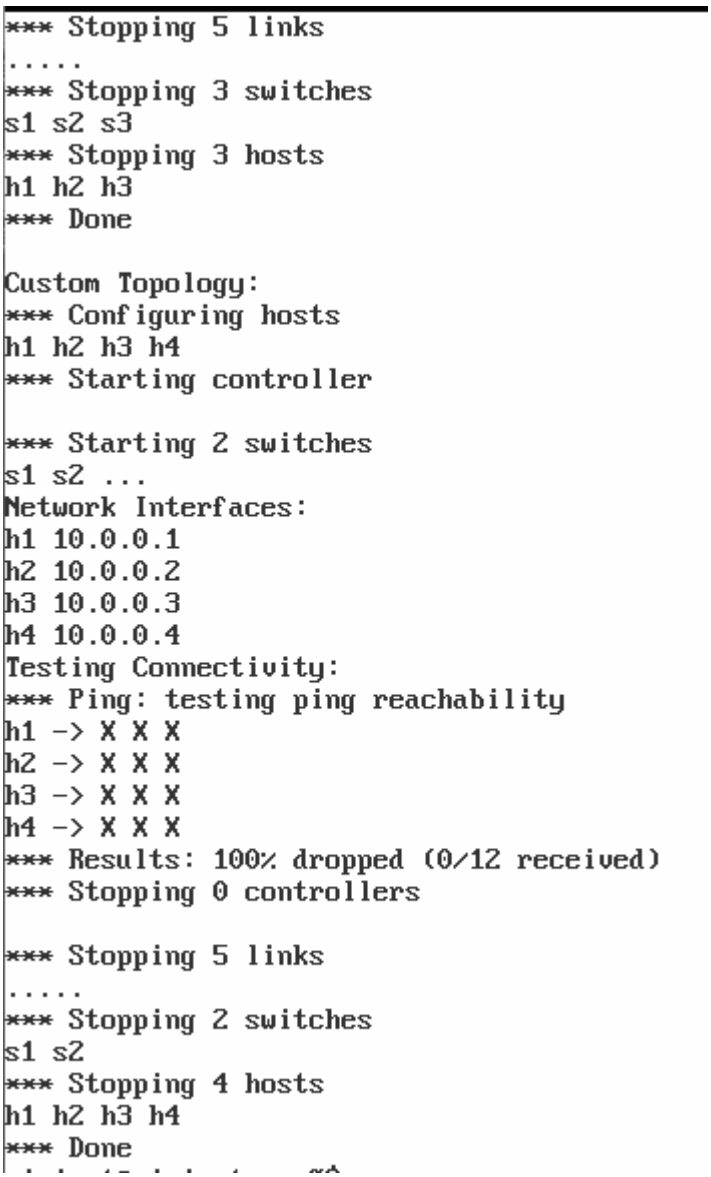
```
from mininet.net import Mininet
from mininet.topo import Topo
from mininet.node import RemoteController

class TreeTopology(Topo):
        def build(self):
                s1 = self.addSwitch('s1')
                s2 = self.addSwitch('s2')
                s3 = self.addSwitch('s3')
                h1 = self.addHost('h1')
                h2 = self.addHost('h2')
                h3 = self.addHost('h3')
                h4 = self.addHost('h4')
                self.addLink(h1, s2)
                self.addLink(h2, s2)
                self.addLink(h3, s3)
                self.addLink(h4, s3)
                self.addLink(s2, s1)
                self.addLink(s3, s1)

topo = TreeTopology()
net = Mininet(topo=topo, controller=lambda name: RemoteController(name, ip='127.0.0.1'))
net.start()
net.pingAll()
net.stop()
```

```
mininet@mininet-vm:~$ sudo python simple.py
Unable to contact the remote controller at 127.0.0.1:6653
Unable to contact the remote controller at 127.0.0.1:6633
Setting remote controller to 127.0.0.1:6653
*** Ping: testing ping reachability
h1 -> X X X
h2 -> X X X
h3 -> X X X
h4 -> X X X
*** Results: 100% dropped (0/12 received)
```

2. Implement the following custom topologies using mininet.
   a. Simple network with 2 hosts and 1 switch.
   b. Design simple linear network with 3 switches and 3 hosts.
   c. Design the following

```
from mininet.net import Mininet
from mininet.node import Controller
from mininet.link import Link, TCLink
from mininet.log import setLogLevel

def simple_topology():
    net = Mininet(controller=Controller)
    h1 = net.addHost('h1')
    h2 = net.addHost('h2')
    s1 = net.addSwitch('s1')

    net.addLink(h1, s1)
    net.addLink(h2, s1)

    net.start()

    print("Network Interfaces:")
    for host in [h1, h2]:
        print(host.name, host.IP())

    print("Testing Connectivity:")
    net.pingAll()

    net.stop()

def linear_topology():
    net = Mininet(controller=Controller)
    hosts = []
    switches = []

    for i in range(3):
        host = net.addHost('h{}'.format(i+1))
```

```
    for i in range(3):
        host = net.addHost('h{}'.format(i+1))
        hosts.append(host)
        switch = net.addSwitch('s{}'.format(i+1))
        switches.append(switch)

        net.addLink(host, switch)
        if i > 0:
            net.addLink(switches[i-1], switch)

    net.start()

    print("Network Interfaces:")
    for host in hosts:
        print(host.name, host.IP())

    print("Testing Connectivity:")
    net.pingAll()

    net.stop()

def custom_topology():
    net = Mininet(controller=Controller)
    h1 = net.addHost('h1')
    h2 = net.addHost('h2')
    h3 = net.addHost('h3')
    h4 = net.addHost('h4')
    s1 = net.addSwitch('s1')
    s2 = net.addSwitch('s2')

    net.addLink(h1, s1)
    net.addLink(h2, s1)
    net.addLink(h3, s2)
```




```
class LinearTopology(Topo):
    def build(self):
        s1 = self.addSwitch('s1')
        s2 = self.addSwitch('s2')
        s3 = self.addSwitch('s3')
        h1 = self.addHost('h1')
        h2 = self.addHost('h2')
        h3 = self.addHost('h3')
        self.addLink(h1, s1)
        self.addLink(h2, s2)
        self.addLink(h3, s3)
```

```
        self.addLink(s1, s2)
        self.addLink(s2, s3)

topo = LinearTopology()
net = Mininet(topo=topo, controller=lambda name:
RemoteController(name, ip='127.0.0.1'))
net.start()
net.pingAll()
net.stop()

class TreeTopology(Topo):
    def build(self):
        s1 = self.addSwitch('s1')
        s2 = self.addSwitch('s2')
        s3 = self.addSwitch('s3')
        h1 = self.addHost('h1')
        h2 = self.addHost('h2')
        h3 = self.addHost('h3')
        h4 = self.addHost('h4')
        self.addLink(h1, s2)
        self.addLink(h2, s2)
        self.addLink(h3, s3)
        self.addLink(h4, s3)
        self.addLink(s2, s1)
        self.addLink(s3, s1)

topo = TreeTopology()
net = Mininet(topo=topo, controller=lambda name:
RemoteController(name, ip='127.0.0.1'))
net.start()
net.pingAll()
net.stop()
```

PRACTICAL 6 : Accessing Mininet VM Remotely

```
ubuntu@ubuntu:~/Desktop$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.103  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::2846:5d86:634a:3866  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:83:f1:3e  txqueuelen 1000  (Ethernet)
        RX packets 57  bytes 11024 (11.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 278  bytes 28409 (28.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 142  bytes 13232 (13.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 142  bytes 13232 (13.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
mininet@mininet-vm:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.102  netmask 255.255.255.0  broadcast 192.168.56.255
        ether 08:00:27:8a:6c:fa  txqueuelen 1000  (Ethernet)
        RX packets 4381  bytes 528045 (528.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 37  bytes 12444 (12.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 3240  bytes 194984 (194.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3240  bytes 194984 (194.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

mininet@mininet-vm:~$ ssh -Y mininet@192.168.56.103
ssh: connect to host 192.168.56.103 port 22: Connection refused
```

```
ubuntu@ubuntu:~/Desktop$ ssh -Y mininet@192.168.56.102
The authenticity of host '192.168.56.102 (192.168.56.102)' can't be established.
ED25519 key fingerprint is SHA256:jmDfxfKW+fe/DR6uz70MhP6alWKjIT70OizYr5goVeg.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.102' (ED25519) to the list of known hosts.
mininet@192.168.56.102's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-42-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your In

Last login: Thu Oct  5 02:11:36 2023
/usr/bin/xauth:  file /home/mininet/.Xauthority does not exist
```

```
mininet@mininet-vm:~$ sudo mn --mac --topo single,4 --controller none
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller

*** Starting 1 switches
s1 ...
*** Starting CLI:
```

```
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
From 10.0.0.1 icmp_seq=4 Destination Host Unreachable
From 10.0.0.1 icmp_seq=5 Destination Host Unreachable
From 10.0.0.1 icmp_seq=6 Destination Host Unreachable
From 10.0.0.1 icmp_seq=7 Destination Host Unreachable
From 10.0.0.1 icmp_seq=8 Destination Host Unreachable
From 10.0.0.1 icmp_seq=9 Destination Host Unreachable
From 10.0.0.1 icmp_seq=10 Destination Host Unreachable
From 10.0.0.1 icmp_seq=11 Destination Host Unreachable
From 10.0.0.1 icmp_seq=12 Destination Host Unreachable
From 10.0.0.1 icmp_seq=13 Destination Host Unreachable
From 10.0.0.1 icmp_seq=14 Destination Host Unreachable
From 10.0.0.1 icmp_seq=15 Destination Host Unreachable
```

```
root@ubuntu:/home/ubuntu/Desktop# cat firewall.csv
00:00:00:00:00:01 00:00:00:00:00:03
00:00:00:00:00:02 00:00:00:00:00:04
root@ubuntu:/home/ubuntu/Desktop#
```

```
root@ubuntu:/home/ubuntu/Desktop/pox# ./pox.py ved
POX 0.1.0 (betta) / Copyright 2011-2013 James McCauley, et al.
INFO:core:POX 0.1.0 (betta) is up.
INFO:openflow.of_01:[None 1] closed
INFO:openflow.of_01:[00-00-00-00-00-01 2] connected
```

```
*** Cleanup complete.
root@ubuntu:/home/ubuntu/Desktop/mininet# mn --mac --topo single,4 --controller remote
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6653
Connecting to remote controller at 127.0.0.1:6633
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> h1 ping -c2 h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.

--- 10.0.0.3 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1002ms

mininet> h2 ping -c2 h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.

--- 10.0.0.4 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1018ms
```

```
--- 10.0.0.4 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1018ms

mininet> h1 ping -c2 h
ping: h: Temporary failure in name resolution
mininet> h1 ping -c2 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=53.8 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.036 ms

--- 10.0.0.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.036/26.920/53.805/26.884 ms
mininet>
```

```
 cookie=0x0, duration=51.997s, table=0, n_packets=2, n_bytes=196, dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:04 actions=drop
root@ubuntu:/home/ubuntu/Desktop/mininet# ovs-ofctl dump-flows s1
 cookie=0x0, duration=190.931s, table=0, n_packets=2, n_bytes=196, dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:03 actions=drop
 cookie=0x0, duration=190.891s, table=0, n_packets=2, n_bytes=196, dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:04 actions=drop
root@ubuntu:/home/ubuntu/Desktop/mininet#
```

```python
from pox.core import core
from pox.lib.addresses import IPAddr, EthAddr
import pox.openflow.libopenflow_01 as of
import os

class Switch:
  def __init__ (self, connection):
    self.connection = connection
    self.macToPort = {}

    connection.addListeners(self)

  def _handle_PacketIn (self, event):
    in_port=event.port
    dpid=event.dpid
    packet = event.parsed
    eth = packet.find("ethernet")
    self.macToPort[eth.src]=in_port
    if eth.dst in self.macToPort:
        out_port=self.macToPort[eth.dst]
    else:
        out_port=of.OFPP_FLOOD

    if out_port!=of.OFPP_FLOOD:
            msg = of.ofp_flow_mod()
            msg.match = of.ofp_match()
            msg.match.dl_dst=eth.dst
            msg.match.in_port=event.port
            msg.idle_timeout = 10
            msg.hard_timeout = 30
            msg.actions.append(of.ofp_action_output(port = out_port))
            msg.data = event.ofp
            self.connection.send(msg)
    else:
        msg = of.ofp_packet_out()
        msg.actions.append(of.ofp_action_output(port = out_port))
        msg.data = event.ofp
        self.connection.send(msg)
```

```
38          self.connection.send(msg)
39
40  def _handle_ConnectionUp (event):
41
42      policyFile = "Sfirewall.csv"
43      rules_file = open(policyFile,"r")
44      rules=[rule.strip()for rule in rules_file]
45      for i in range(len(rules)):
46          rule_list=rules[i].split(' ')
47          fw_add_rule=of.ofp_flow_mod()
48          fw_add_rule.match=of.ofp_match()
49          fw_add_rule.match.dl_src=EthAddr(rule_list[0])
50          fw_add_rule.match.dl_dst=EthAddr(rule_list[1])
51          event.connection.send(fw_add_rule)
52
53      Switch(event.connection)
54
55  def launch ():
56    core.openflow.addListenerByName("ConnectionUp", _handle_ConnectionUp)
```

Practical 8 : NET SIM