A large enterprise company specialized in exporting Fruit, has plans to upgrade their network and to ensure they are ready for the future they want to implement IPv6. The backbone of the network however is still based on IPv4 and it is not allowed to make any changes Propose connectivity by tunnelling

IPv4/IPv6 Coexistence

The transition from IPv4 to IPv6 will not be something that is done overnight; it will take a number of years before IPv6 has anywhere near 100 percent implementation. In these intervening years (including now), a number of mechanisms have been (and will be) developed to make the transition as easy as possible.

The first of the available options is referred to as **dual stack**. When using this method, an organization essentially does not transition to IPv6 but simply builds a parallel IPv6 network next to their existing IPv4 network.

The second of the available options is **tunneling**. The basic idea behind tunneling methods is that IPv6 will be tunneled over an existing IPv4 network. A number of different tunneling methods are available and can be selected based on the requirements of the situation.

The third of the available options is **translation**. The idea behind translation is that at a boundary router between an IPv4 and an IPv6 network a translation process maps an IPv4 address to an IPv6 address (or vice versa).

Dual Stack

When a network is configured as dual stack, each device on the network is configured with both an IPv4 address and an IPv6 address, the idea being that once all the devices have implemented IPv6, the IPv4 part of the network will be depreciated. This method is common for businesses looking to slowly convert their existing devices from IPv4 to IPv6. These companies can configure their routing infrastructure to support both IPv4 and IPv6 but bring their other network devices over to IPv6 at a slower pace.

It is also possible for individual devices to be configured as **dual stack and use one of the tunneling technologies** discussed in the next section.

Tunneling

The concept behind tunneling is not new; many people use tunneling daily, but just use it for other reasons. For example, many companies use IPsec or Secure Sockets Layer (SSL) tunnels to secure information when it is being transmitted over an untrusted network.

Many different tunneling methods are available. Which one to use depends on the specific implementation details. Table 1 lists some commonly available tunneling methods and their suggested usage.

Tunneling Method	**Suggested Usage**
Manual	Used to provide a point-to-point IPv6 link over an existing IPv4 network; only supports IPv6 traffic.
GRE	Used to provide a point-to-point IPv6 link over an existing IPV4 network; supports multiple protocols, including IPv6.
6to4	Used to provide a point-to-multipoint IPv6 link over an existing IPv4 network; sites must use IPv6 addresses from the 2002::/16 range.
6rd (or 6RD)	Used to provide a point-to-multipoint IPv6 link over an existing IPv4 network; sites can use IPv6 addresses from any range.
ISATAP	Used to provide point-to-multipoint IPv6 links over an existing IPv4 network. Designed to be used between devices inside the same site.

Translation

The concept of address translation is also not a new concept to most network engineers; this is because Network Address Translation (NAT) is implemented between different IPv4 networks in almost every residential household. The concept behind this type of NAT and the newer technologies that support address translation between IPv4 and IPv6 networks is similar. IPv6 translation technologies differ from IPv6 tunneling technologies; this is because the translation technologies enable IPv4-only devices to speak to IPv6-only devices, which is not possible with any of the tunneling methods.

However, IPv4/IPv6 translation and IPv4-only translation entail a certain amount of complexity. What happens when an IPv6-only device is attempting to communicate with a device on the public IPv4 Internet and only an IPv4 DNS record (A) exists? In these situations, a secondary technology is required to step in and provide additional services for the connection to work.

The first method to be introduced to provide IPv6 translation services was Network Address Translation - Protocol Translation (NAT-PT). NAT-PT defined a mechanism to not only translate between IPv4 to IPv6 addresses but also a built-in ability to provide protocol translation services for Internet Control Message Protocol (ICMP), File Transfer Protocol (FTP), and Domain Name System (DNS). The component that was responsible for these translation services is called the application layer gateway (ALG).

The ALG piece of the NAT-PT method raised a number of issues. With additional testing and real-life experience, a new method was introduced that separated the address translation functionality and the application layer translation functionalities: NAT64 and DNS64.

DNS64 can synthesize IPv6 address resource records (AAAA) from IPv4 resource records (A); it does this by encoding the returned IPv4 address into a IPv6 address format.

Summary

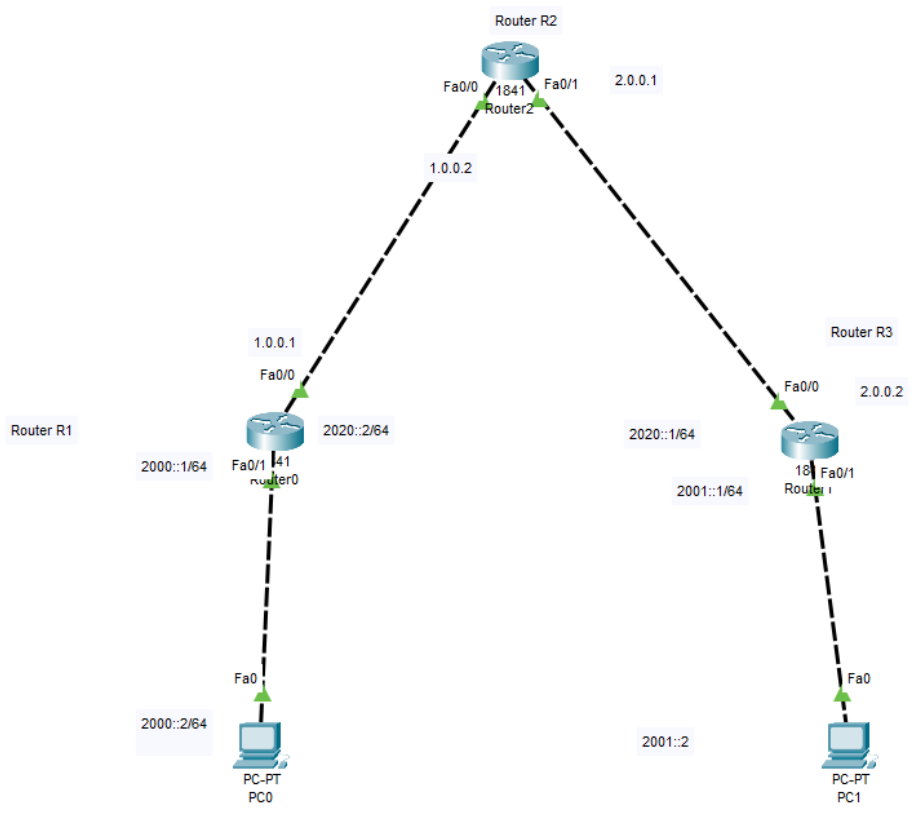
The selection of an IPv6 transition mechanism depends greatly on the current status of an organization's network and how fast they want to transition their devices from IPv4 to IPv6. Logic seems to say that those organizations with bleeding-edge technology tastes and small staffs will probably be (or are already) the first people in line to transition over to IPv6. Those larger companies that have tens of thousands of network devices will most likely transition a piece at a time following the experience level of each department.

The transition to IPv6 is coming, and all those network engineers reading this article should become experts in IPv6 as quickly as possible. The process of converting networks from IPv4 to IPv6 will shortly become a large-scale request, and those with the correct skills will be in demand, a fact even more important in the current economy.

Dual Stack

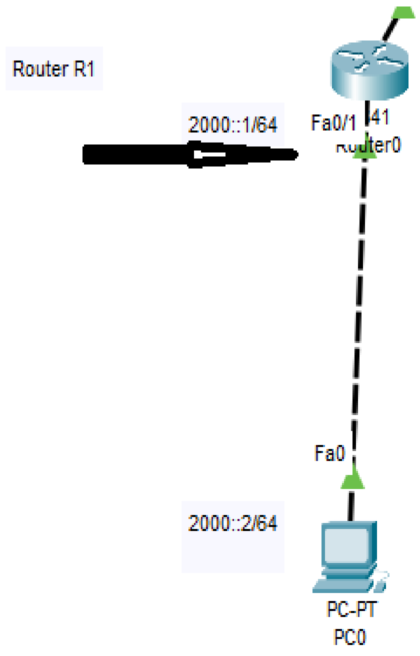
A dual stack network is a network in which both IPv4 and IPv6 would exist at all the nodes and routers. Dual stack networks are one of the many IPv4 to IPv6 migration strategies that have been presented in recent years. Below, we shall see how to setup a dual stack simulation on packet tracer.

Tunneling


Router R2
Fa0/0
1841
Router2
Fa0/1
2.0.0.1
1.0.0.2
1.0.0.1
Fa0/0
Router R1
2020::2/64
2000::1/64
Fa0/1
Router0
Router R3
Fa0/0
2.0.0.2
2020::1/64
Fa0/1
2001::1/64
Fa0
2000::2/64
PC-PT
PC0
2001::2
Fa0
PC-PT
PC1

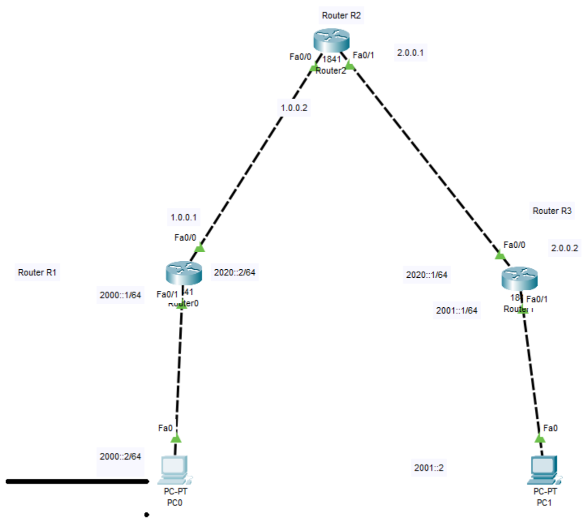
ROUTER'S CONFIGURATION

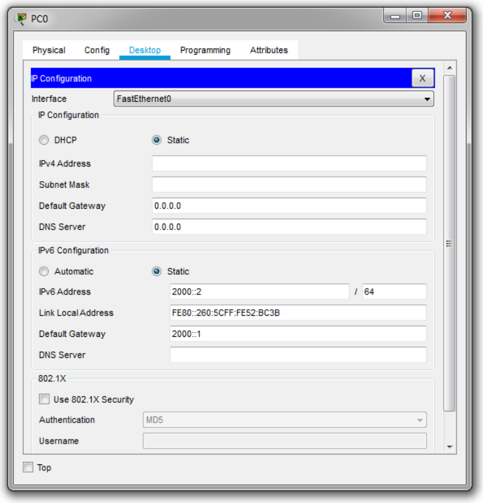
Configure Router R1   F0/1 with IPV6 Address



```
 Router>en
Router#config t
Enter configuration commands, one per line.
End with CNTL/Z.
Router(config)#ipv6 unicast-routing
Router(config)#int fa0/1
Router(config-if)#ipv6 add 2000::1/64
Router(config-if)#no shut

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
```

Configure PC0 with IPV6 Address





Interface	FastEthernet0

IP Configuration

○ DHCP [✓ Static]

IPv4 Address	
Subnet Mask	
Default Gateway	0.0.0.0
DNS Server	0.0.0.0

IPv6 Configuration

○ Automatic [✓ Static]

IPv6 Address	2000::2 / 64
Link Local Address	FE80::260:5CFF:FE52:BC3B
Default Gateway	2000::1
DNS Server	

802.1X

☐ Use 802.1X Security

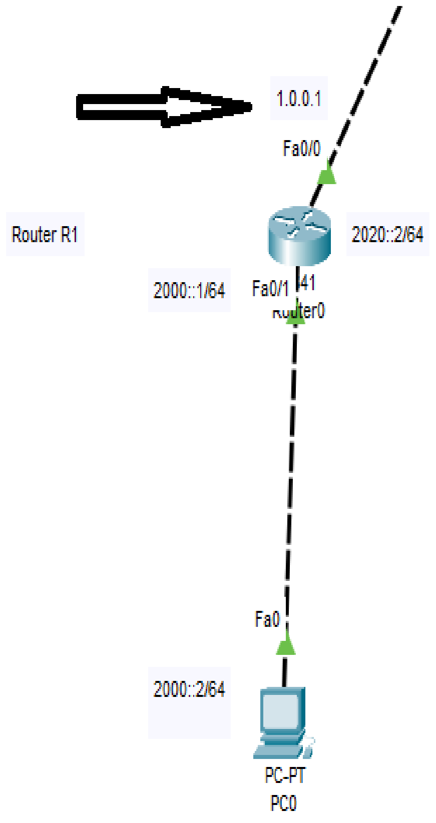
Authentication	MD5
Username	

☐ Top

Configure Router R1 with Routing eigrp protocol

```
Router(config-if)#exit
Router(config)#router eigrp 1
Router(config-router)#network 1.0.0.0 255.0.0.0
Router(config-router)#exit
```

Configure f0/0 with IPv4 address



```
Router(config)#int fa0/0
Router(config-if)#ip add 1.0.0.1
255.0.0.0
Router(config-if)#no shut

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

Router(config-if)#exit
Router(config)#
%LINEPROTO-5-UPDOWN: Line
protocol on Interface FastEthernet0/0,
changed state to up
```

```
%DUAL-5-NBRCHANGE: IP-EIGRP 1: Neighbor 1.0.0.2 (FastEthernet0/0)
is up: new adjacency
```

Configure tunnel 0 at router R1

```
Router(config)#int tunnel 0

Router(config-if)#
%LINK-5-CHANGED: Interface Tunnel0, changed state to up

Router(config-if)#tunnel source fa0/0   ( Router's POrt from where this
tunnel will start )

Router(config-if)#tunnel destination 2.0.0.2  ( Ip address of router's port
where tunnel will end )
Router(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed
state to up
```

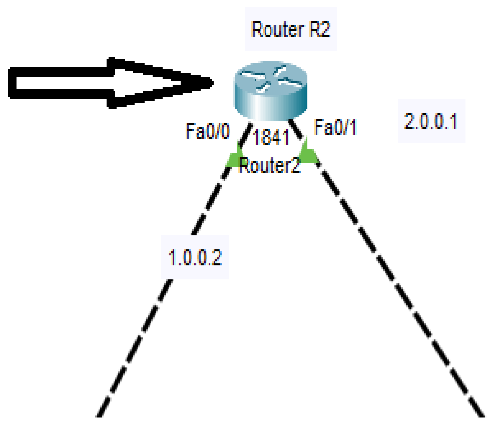
Assign IP address to the tunnel 0

```
Router(config-if)#tunnel mode ipv6ip
Router(config-if)#ipv6 add 2020::2/64  (  new ipv6 network to this tunnel
port )
Router(config-if)#exit
```

STATIC ROUTING

```
Router(config)#ipv6 route 2001::/64 2020::1
Router(config)#
```

R2 ROUTER's CONFIGURATION

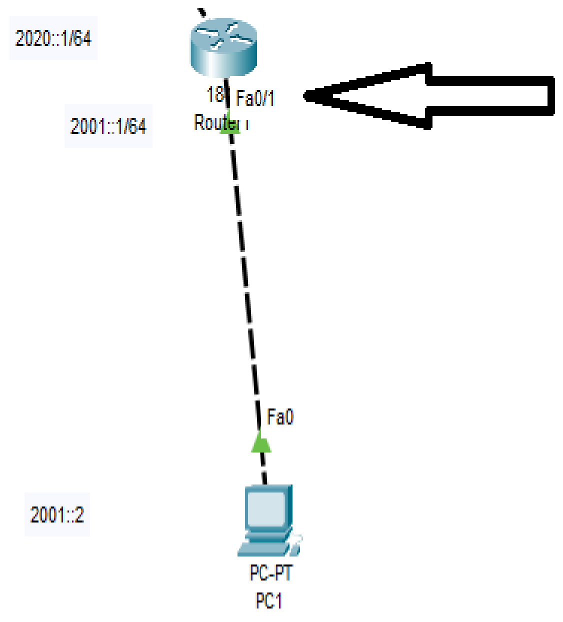


```
Router>en
Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int fa0/1
Router(config-if)#ip add 2.0.0.1 255.0.0.0
Router(config-if)#no shut

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/1, changed state to up

Router(config-if)#exit
Router(config)#int fa0/0
Router(config-if)#ip add 1.0.0.2 255.0.0.0
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
```

Configure Router R2 with Routing eigrp protocol

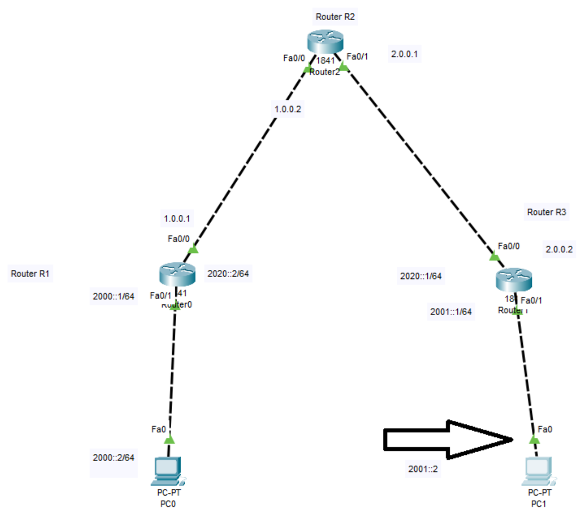
```
Router(config)#router eigrp 1
Router(config-router)#network 1.0.0.0 255.0.0.0
Router(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 1: Neighbor 1.0.0.1 (FastEthernet0/0)
is up: new adjacency
```

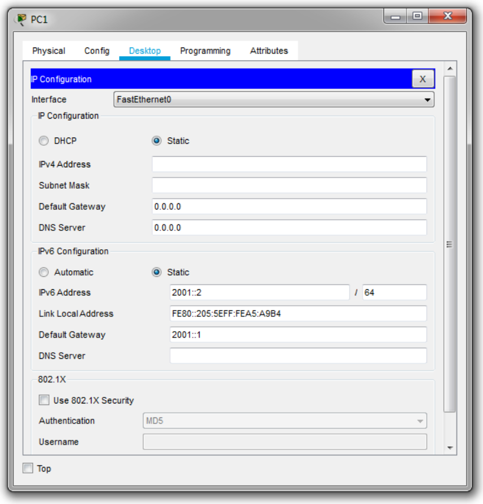
```
Router(config-router)#network 2.0.0.0 255.0.0.0
Router(config-router)#exit
Router(config)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1,
changed state to up

%DUAL-5-NBRCHANGE: IP-EIGRP 1: Neighbor 2.0.0.2 (FastEthernet0/1)
is up: new adjacency
```



R3 ROUTER's CONFIGURATION

```
Router>en
Router#config t
Enter configuration commands,
one per line. End with CNTL/Z.
Router(config)#ipv6
unicast-routing
Router(config)#int fa0/1
Router(config-if)#ipv6 add
2001::1/64
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
```





PC1

Physical | Config | Desktop | Programming | Attributes

IP Configuration

Interface: FastEthernet0

IP Configuration

DHCP [✓ Static]

IPv4 Address:

Subnet Mask:

Default Gateway: 0.0.0.0

DNS Server: 0.0.0.0

IPv6 Configuration

Automatic [✓ Static]

IPv6 Address: 2001::2 / 64

Link Local Address: FE80::205:5EFF:FEA5:A9B4

Default Gateway: 2001::1

DNS Server:

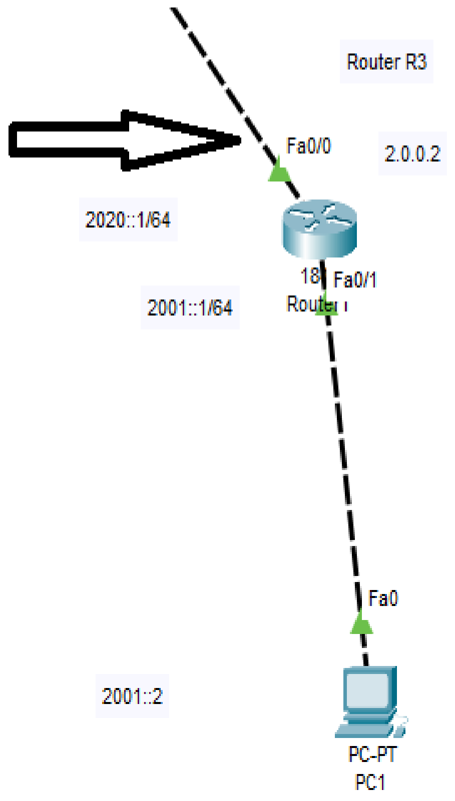
802.1X

Use 802.1X Security

Authentication: MD5

Username:

Top



```
Router(config-if)#exit

Router(config)#int fa0/0

Router(config-if)#ip add 2.0.0.2
255.0.0.0

Router(config-if)#no shut

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
```

Configure Router R3 with Routing eigrp protocol

```
Router(config-if)#exit
Router(config)#router eigrp 1
Router(config-router)#network 2.0.0.0 255.0.0.0
Router(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 1: Neighbor 2.0.0.1 (FastEthernet0/1)
is up: new adjacency

Router(config-router)#exit
```

TUNNEL CONFIGURATION IN ROUTER R3

```
Router(config)#int tunnel 0

Router(config-if)#
%LINK-5-CHANGED: Interface Tunnel0, changed state to up

Router(config-if)#tunnel source fa0/0
Router(config-if)#tunnel destination 1.0.0.1
Router(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed
state to up
```

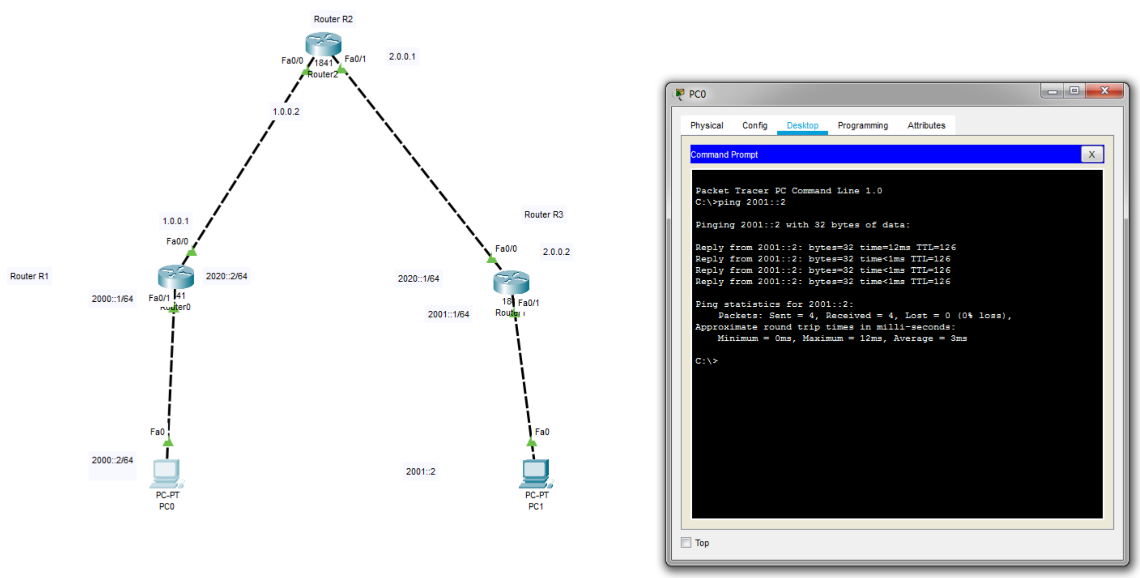
Assign IP address to the tunnel 0

```
Router(config-if)#tunnel mode ipv6ip
Router(config-if)#ipv6 add 2020::1/64    (  new IPV6 ADDRESS TO THIS
TUNNEL PORT )
Router(config-if)#exit
```

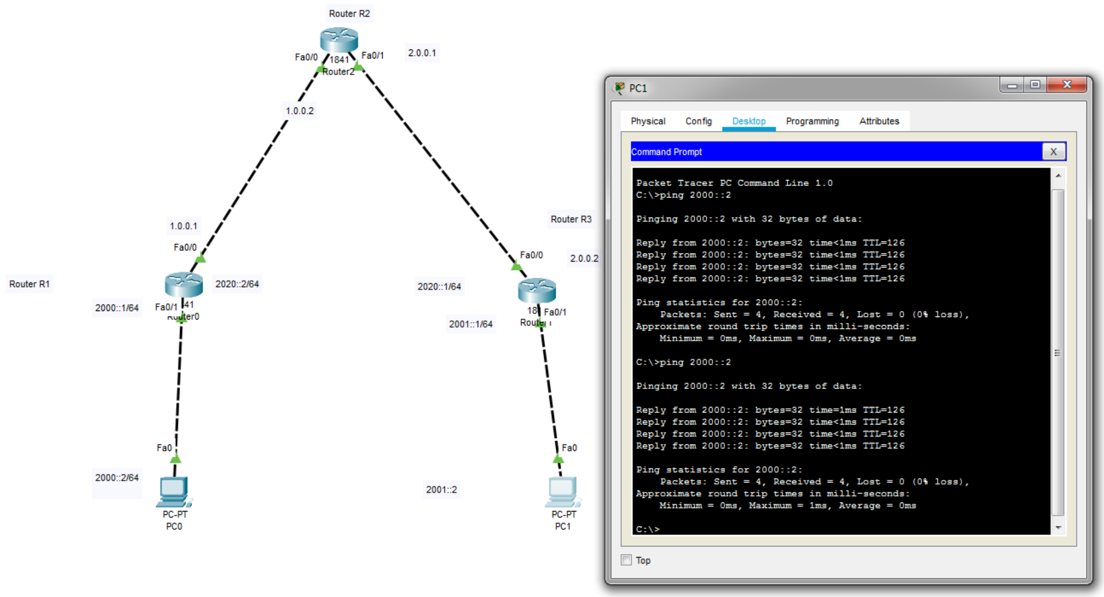
Static Routing

```
Router(config)#ipv6 route 2000::/64 2020::2
Router(config)#
```

Check the output



```
Packet Tracer PC Command Line 1.0
C:\>ping 2001::2

Pinging 2001::2 with 32 bytes of data:

Reply from 2001::2: bytes=32 time=12ms TTL=126
Reply from 2001::2: bytes=32 time<1ms TTL=126
Reply from 2001::2: bytes=32 time<1ms TTL=126
Reply from 2001::2: bytes=32 time<1ms TTL=126

Ping statistics for 2001::2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 12ms, Average = 3ms

C:\>
```



```
Packet Tracer PC Command Line 1.0
C:\>ping 2000::2

Pinging 2000::2 with 32 bytes of data:

Reply from 2000::2: bytes=32 time<1ms TTL=126
Reply from 2000::2: bytes=32 time<1ms TTL=126
Reply from 2000::2: bytes=32 time<1ms TTL=126
Reply from 2000::2: bytes=32 time<1ms TTL=126

Ping statistics for 2000::2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 2000::2

Pinging 2000::2 with 32 bytes of data:

Reply from 2000::2: bytes=32 time=1ms TTL=126
Reply from 2000::2: bytes=32 time<1ms TTL=126
Reply from 2000::2: bytes=32 time<1ms TTL=126
Reply from 2000::2: bytes=32 time<1ms TTL=126

Ping statistics for 2000::2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\>
```