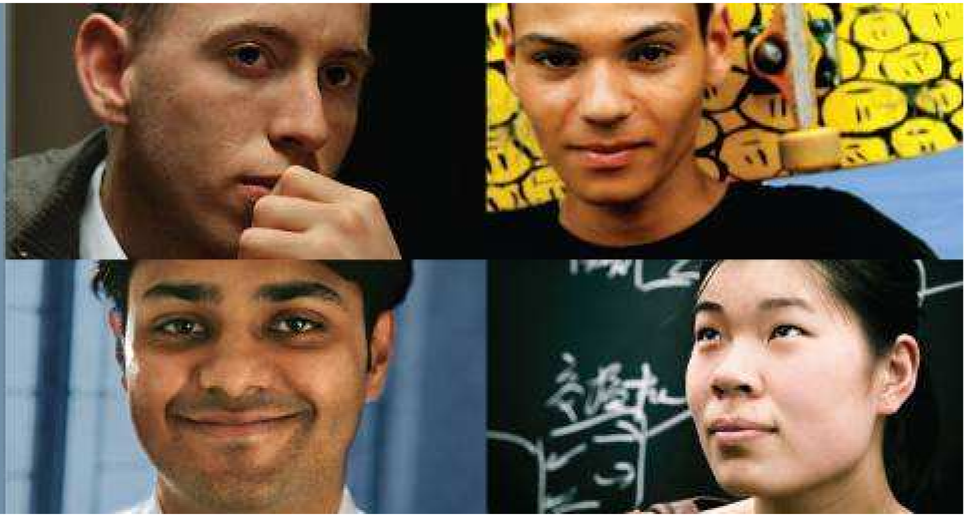
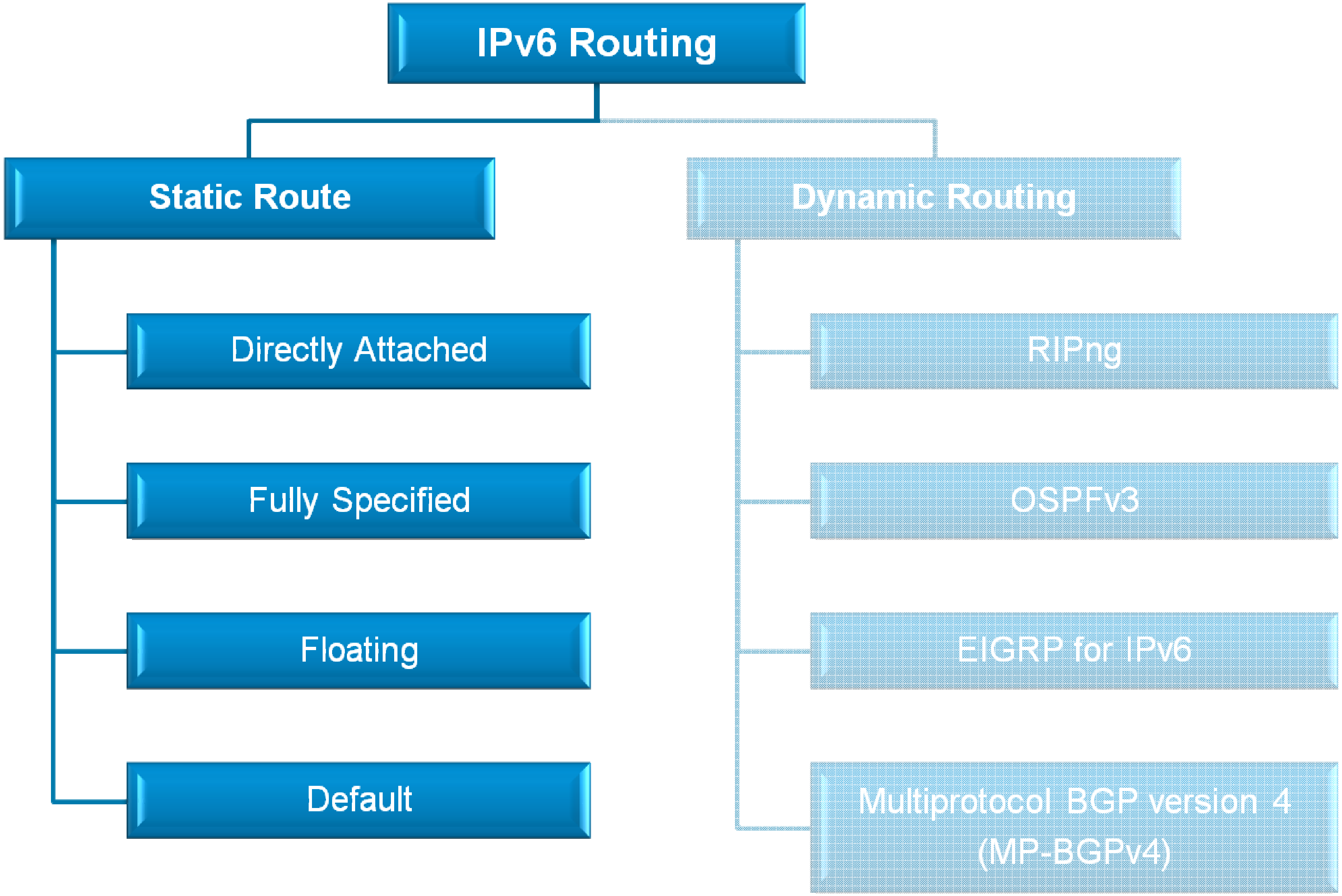
IPv6 Static Routes



Overview

Cisco | Networking Academy®
Mind Wide Open™


IPv6 Routing
Static Route
Directly Attached
Fully Specified
Floating
Default
Dynamic Routing
RIPng
OSPFv3
EIGRP for IPv6
Multiprotocol BGP version 4
(MP-BGPv4)

IPv6 Static Routes

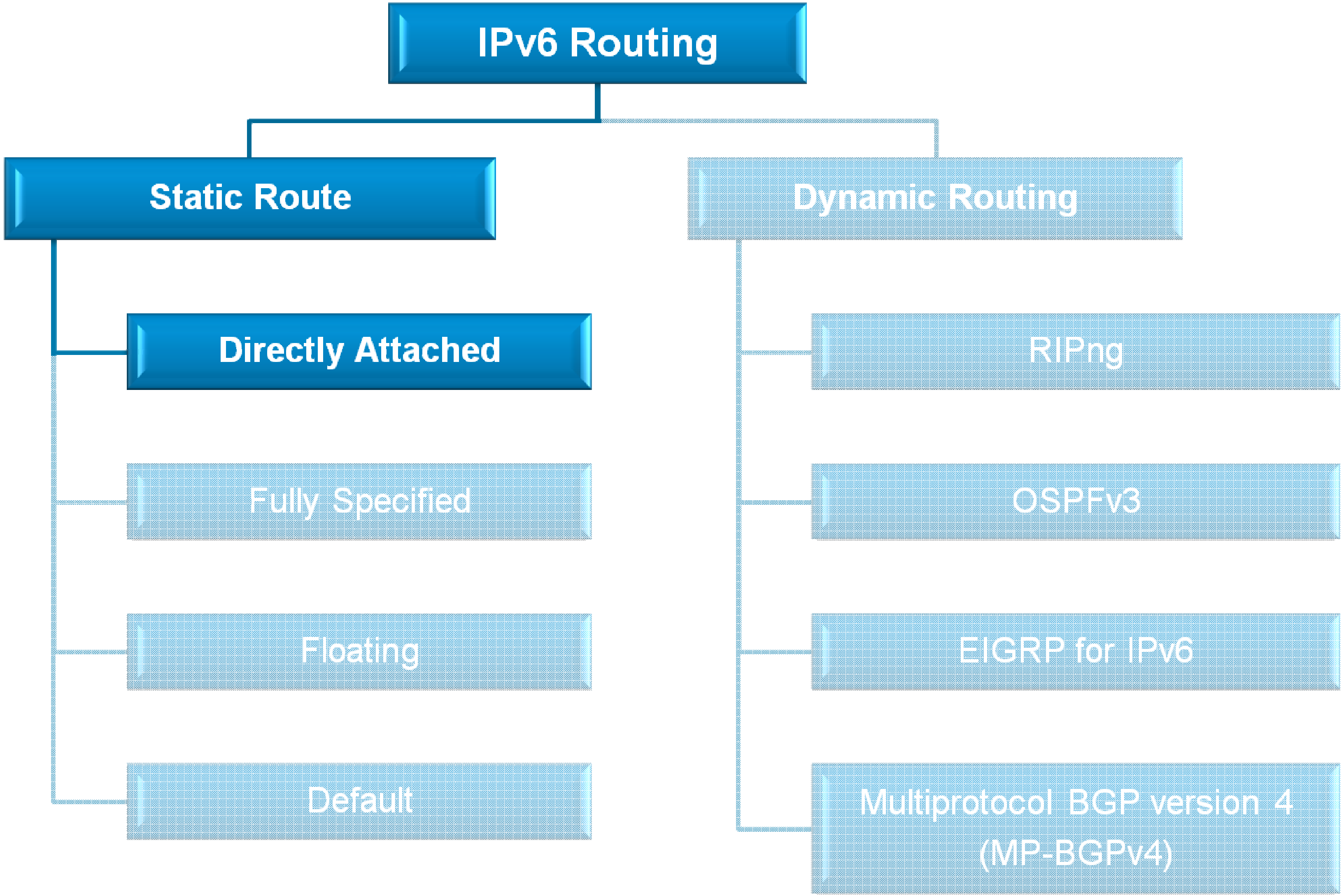
- Static routes are manually configured and define an explicit path between two networking devices.
- Configuring an IPv6 static route is very similar to IPv4 except that the command is now **`ipv6 route`**.
- The following must be configured before entering a static IPv6 route:
  - **`ipv6 unicast-routing`**
  - IPv6 enabled on at least one interface
  - An IPv6 address on that interface.

Complete IPv6 Static Route Syntax

```
Router(config)#
```

```
ipv6 route ipv6-prefix/prefix-length
  {ipv6-address | interface-type interface-number [ipv6-address]}

  [administrative-distance]
```

- The syntax of the IPv6 command contains more parameters than the IPv4 version.
- The following command parameters are not required to configure directly attached, fully specified, floating and default static routes.
  - Refer to cisco.com for more information on these parameters.


IPv6 Routing
Static Route
Dynamic Routing
Directly Attached
Fully Specified
Floating
Default
RIPng
OSPFv3
EIGRP for IPv6
Multiprotocol BGP version 4 (MP-BGPv4)

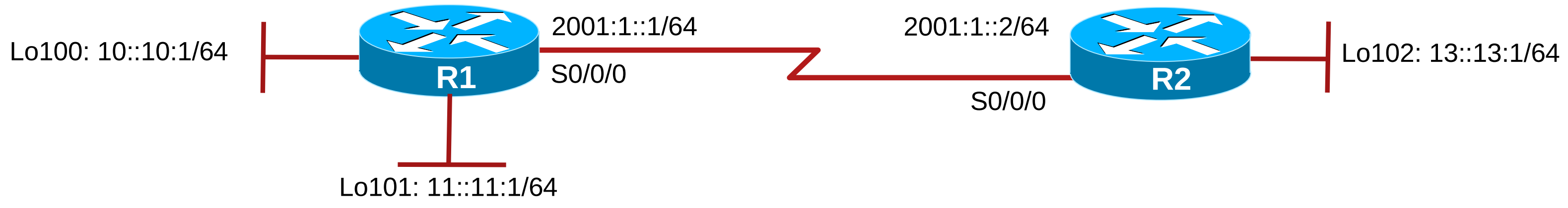
Directly Attached IPv6 Static Route

```
Router(config)#
```

```
ipv6 route ipv6-prefix/prefix-length
  {ipv6-address | interface-type interface-number [ipv6-address]}
  [administrative-distance]
```

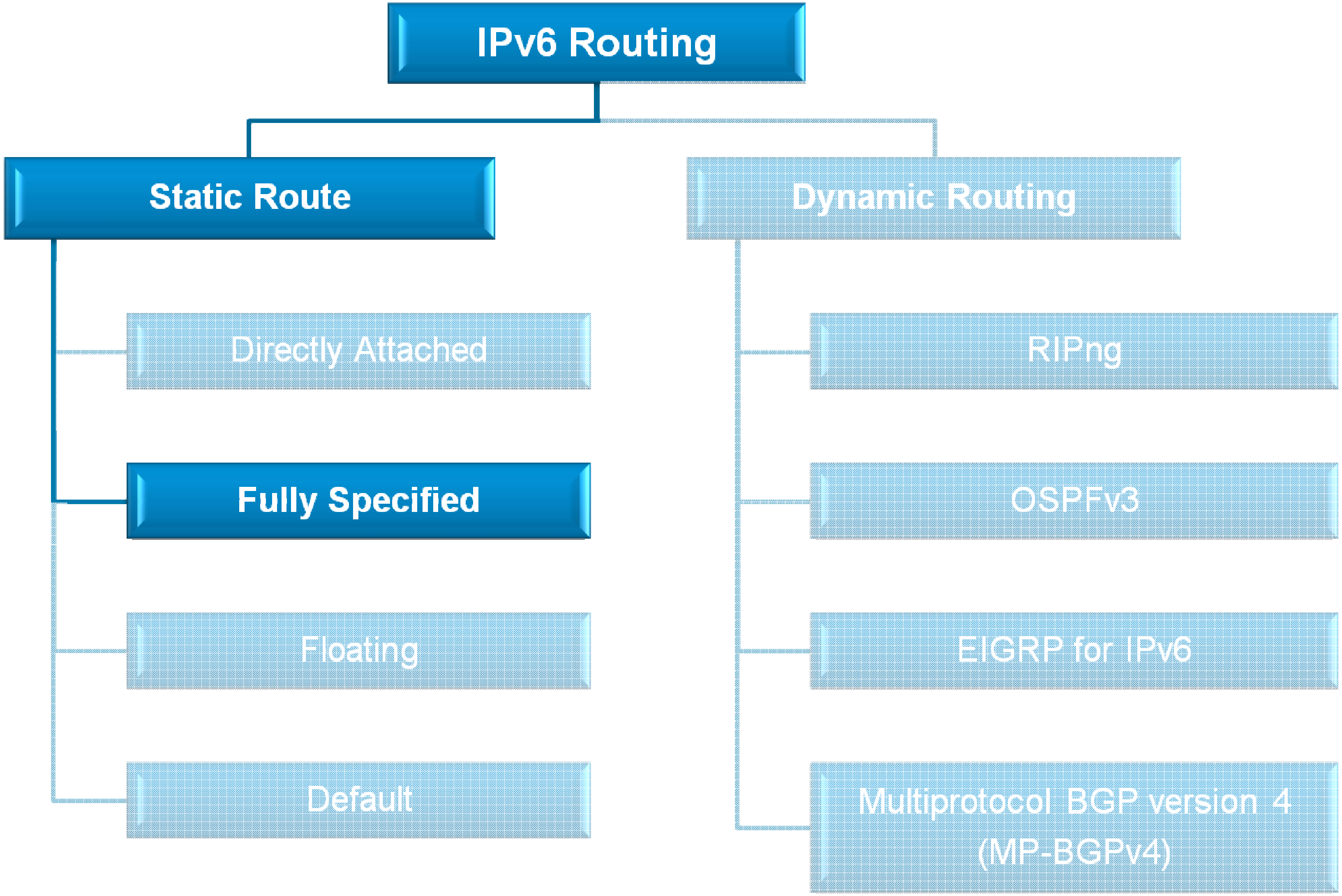
- A directly attached IPv6 static route is created when specifying only outgoing interface.
- The *ipv6-prefix/prefix-length* parameter identifies the destination IPv6 network and its prefix length.
- The *interface-type interface-number* parameter specifies the interface through which the destination network can be reached.

Directly Attached IPv6 Static Route Example




```
R1# config t
R1(config)# ipv6 route 13::/64 s0/0/0
R1(config)# exit
R1# show ipv6 route static
IPv6 Routing Table – 9 entries
Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP
       U – Per-user Static route
       I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary
       O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2
       ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2
S    13::/64 [1/0]
      via ::, Serial0/0/0
R1#
```

- A directly attached static route to the 13::13:1/64 network is configured on router R1.


IPv6 Routing
Static Route
Dynamic Routing
Directly Attached
Fully Specified
Floating
Default
RIPng
OSPFv3
EIGRP for IPv6
Multiprotocol BGP version 4 (MP-BGPv4)

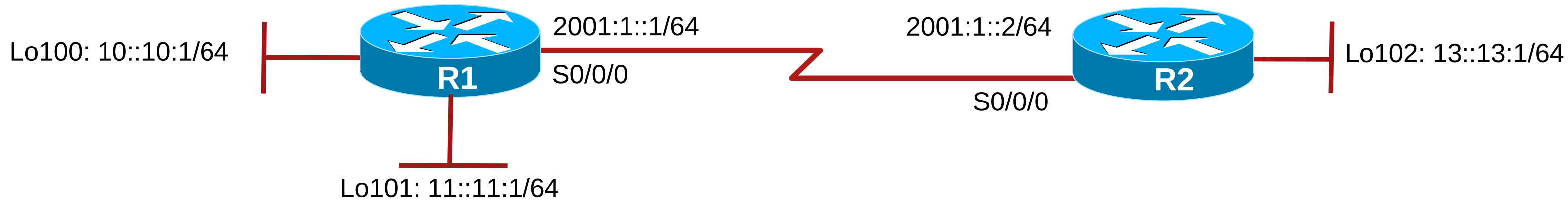
Fully Specified IPv6 Static Route

```
Router(config)#
```

```
ipv6 route ipv6-prefix/prefix-length
  {ipv6-address | interface-type interface-number [ipv6-address]}
  [administrative-distance]
```

- A fully specified static route is created when specifying:
  - The outgoing interface
  - And the next hop IP address.
- This method avoids a recursive lookup.

Fully Specified IPv6 Static Route Example




```
R1# config t
R1(config)# ipv6 route 13::/64 s0/0/0 2001:1::2
R1(config)# exit
R1# show ipv6 route static
IPv6 Routing Table - Default - 8 entries
Codes: C - Connected, L - Local, S - Static, U - Per-user Static route
       B - BGP, M - MIPv6, R - RIP, I1 - ISIS L1
       I2 - ISIS L2, IA - ISIS interarea, IS - ISIS summary, D - EIGRP
       EX - EIGRP external
       O - OSPF Intra, OI - OSPF Inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
       ON1 - OSPF NSSA ext 1, ON2 - OSPF NSSA ext 2
S   13::/64 [1/0]
     via 2001:1::2, Serial0/0/0
R1#
```

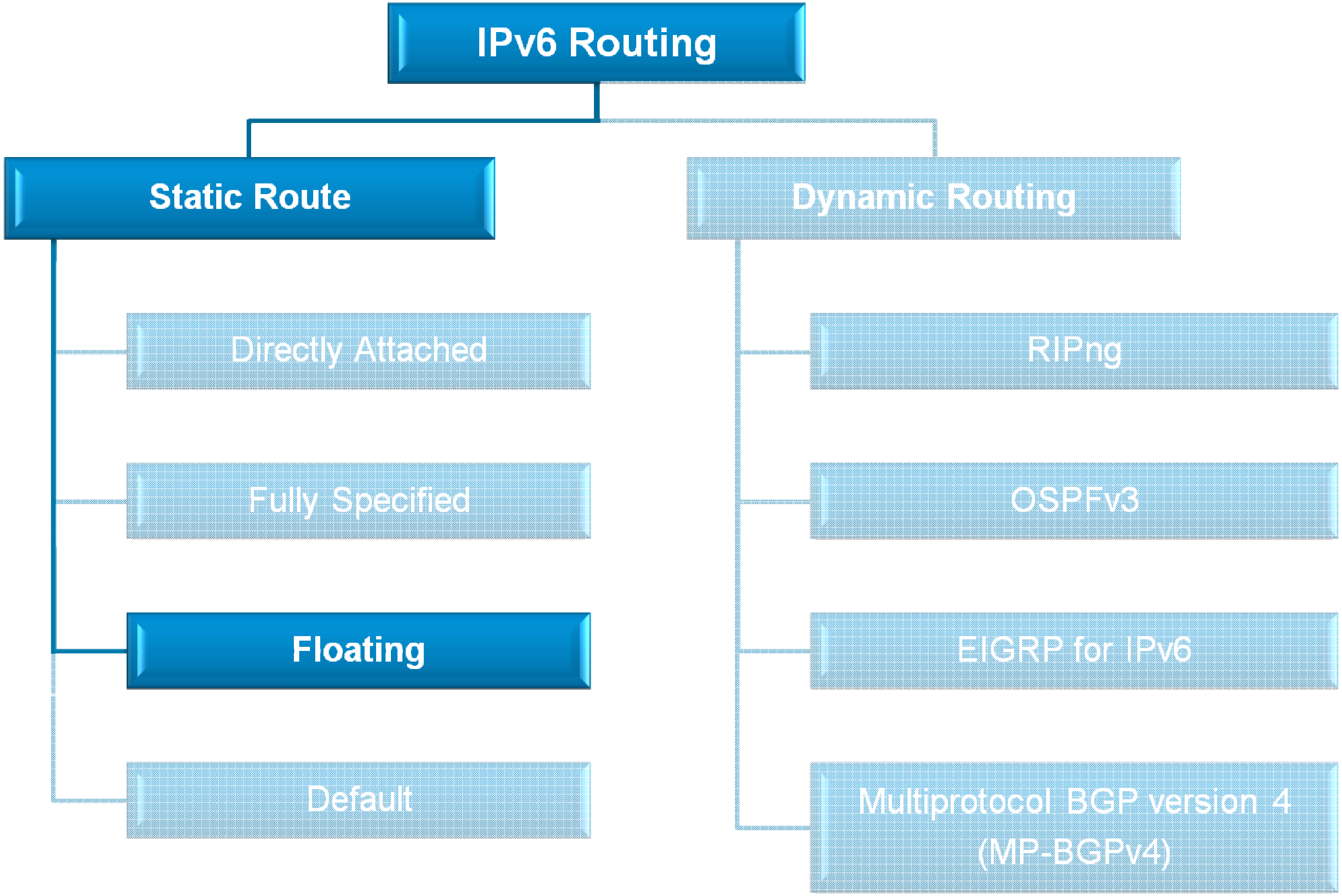
- A fully specified static route to the 13::13:1/64 network is configured on router R1.

Note: Recursive IPv6 Static Route

```
Router(config)#
```

```
ipv6 route ipv6-prefix/prefix-length
  {ipv6-address | interface-type interface-number [ipv6-address]}
  [administrative-distance]
```

- A recursive static route is configured when specifying the next hop IP address of the neighbor.
  - This makes the router perform a second route lookup to resolve the outgoing interface to the specified next hop address.
- Typically, recursive static routes should be avoided.


IPv6 Routing
Static Route
Dynamic Routing
Directly Attached
Fully Specified
Floating
Default
RIPng
OSPFv3
EIGRP for IPv6
Multiprotocol BGP version 4 (MP-BGPv4)

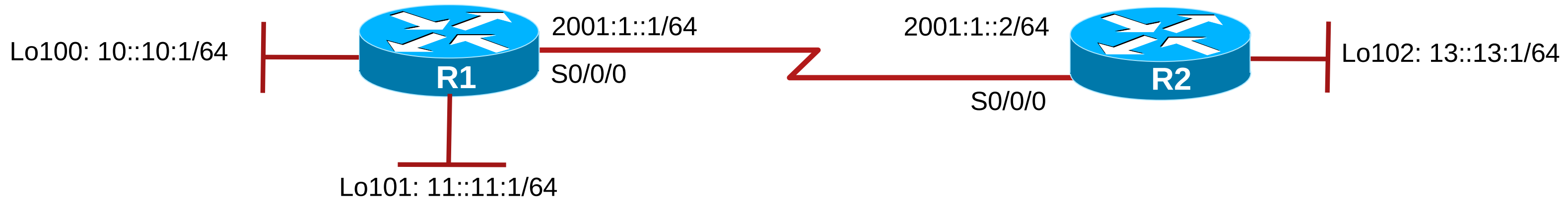
Floating IPv6 Static Route

```
Router(config)#
```

```
ipv6 route ipv6-prefix/prefix-length
  {ipv6-address | interface-type interface-number [ipv6-address]}
  [administrative-distance]
```

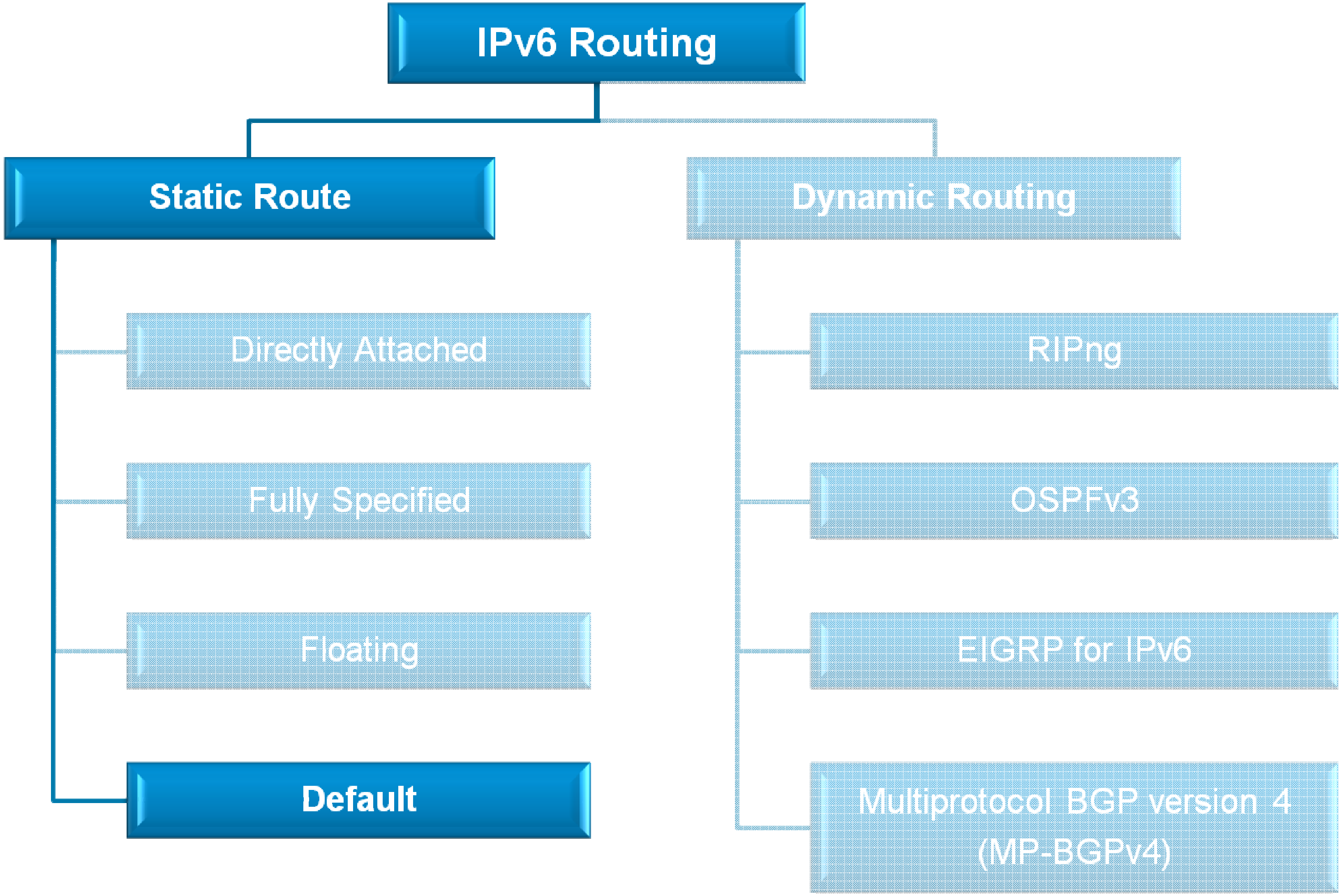
- A floating static route is usually configured when there are multiple paths to a destination network and a standby backup route is required to support IGP discovered routes.
  - It will only be added to the routing table if the IGP entry is deleted.
- The *administrative-distance* parameter specifies the value of the route, which should be higher than the IGP in the routing table.
  - The default value is 1, which is why static routes have precedence over any other type of route except connected routes.

Floating IPv6 Static Route Example




```
R1# config t
R1(config)# ipv6 route 13::/64 130
R1(config)# exit
R1#
```

- For example, R1 is configured with a floating static route specifying an administrative distance of 130 to the R2 LAN.
  - If an IGP already has an entry in the IPv6 routing table to this LAN, then the static route would only appear in the routing table if the IGP entry was removed.


IPv6 Routing
Static Route
Dynamic Routing
Directly Attached
Fully Specified
Floating
Default
RIPng
OSPFv3
EIGRP for IPv6
Multiprotocol BGP version 4 (MP-BGPv4)

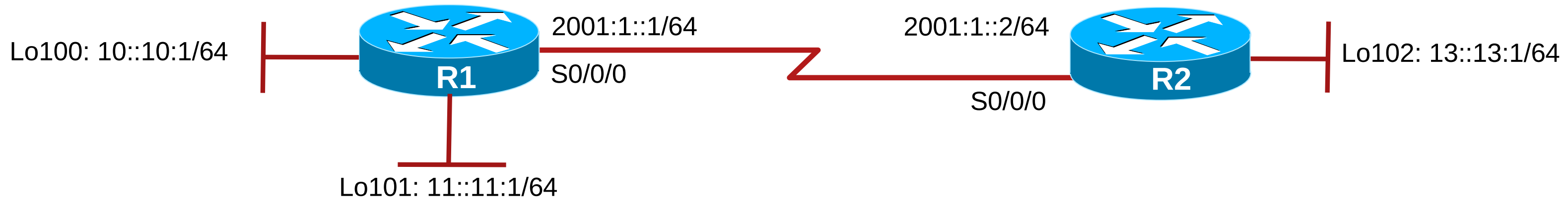
Default IPv6 Static Route

```
Router(config)#
```

ipv6 route ::/0
{*ipv6-address* | *interface-type interface-number* **[***ipv6-address***]}**
[*administrative-distance***]**

- IPv6 also has a default static route similar to the IPv4 quad zero (0.0.0.0) static default route.
- Instead, the IPv6 command uses the **::/0** notation to specify all networks.

Default IPv6 Static Route Example

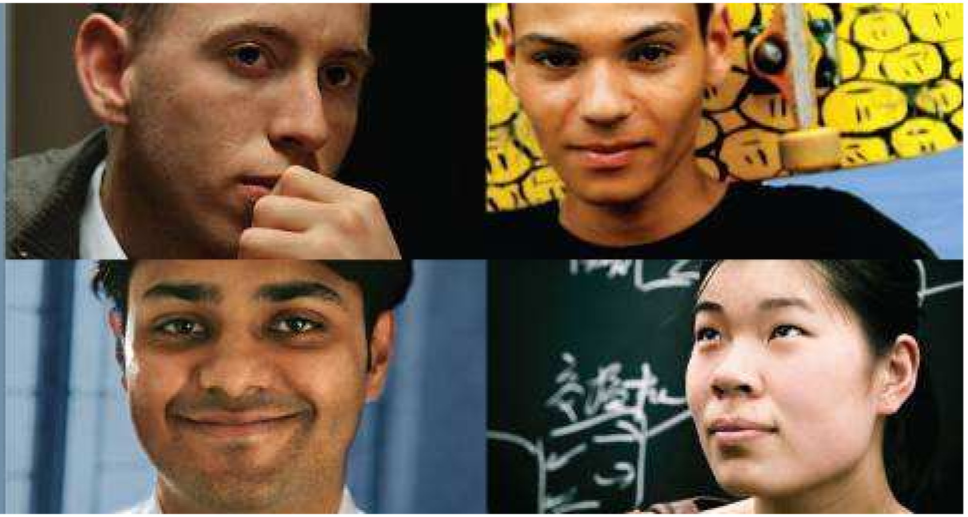



```
R2# config t
R2(config)# ipv6 route ::/0 s0/0/0
R2(config)# exit
R2# show ipv6 route static
IPv6 Routing Table – 9 entries
Codes: C – Connected, L – Local, S – Static, R – RIP, B – BGP
       U – Per-user Static route
       I1 – ISIS L1, I2 – ISIS L2, IA – ISIS interarea, IS – ISIS summary
       O – OSPF intra, OI – OSPF inter, OE1 – OSPF ext 1, OE2 – OSPF ext 2
       ON1 – OSPF NSSA ext 1, ON2 – OSPF NSSA ext 2
S     ::/0 [1/0]
       via ::, Serial0/0/0
R2#
```

- For example, a default static route as specified by the “**::/0**” entry is configured on router R2 to reach all other networks connected to R1.

Cisco | Networking Academy®
Mind Wide Open™

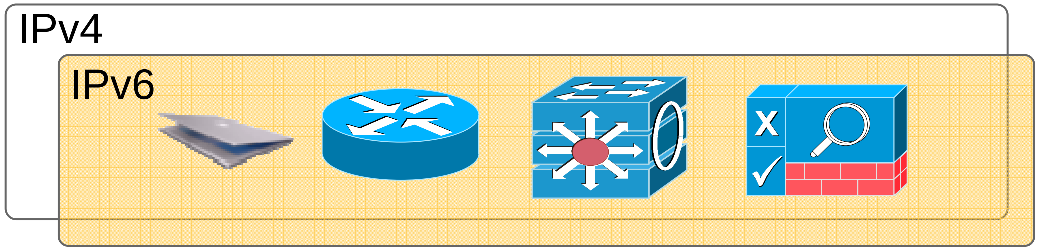
Transitioning IPv4 to IPv6



Cisco | Networking Academy®
Mind Wide Open™

IPv6 Co-existence Solutions

Dual-Stack

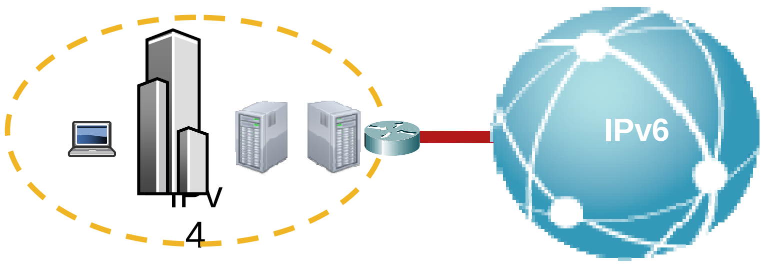


Enterprise Co-existence strategy

Tunneling Services

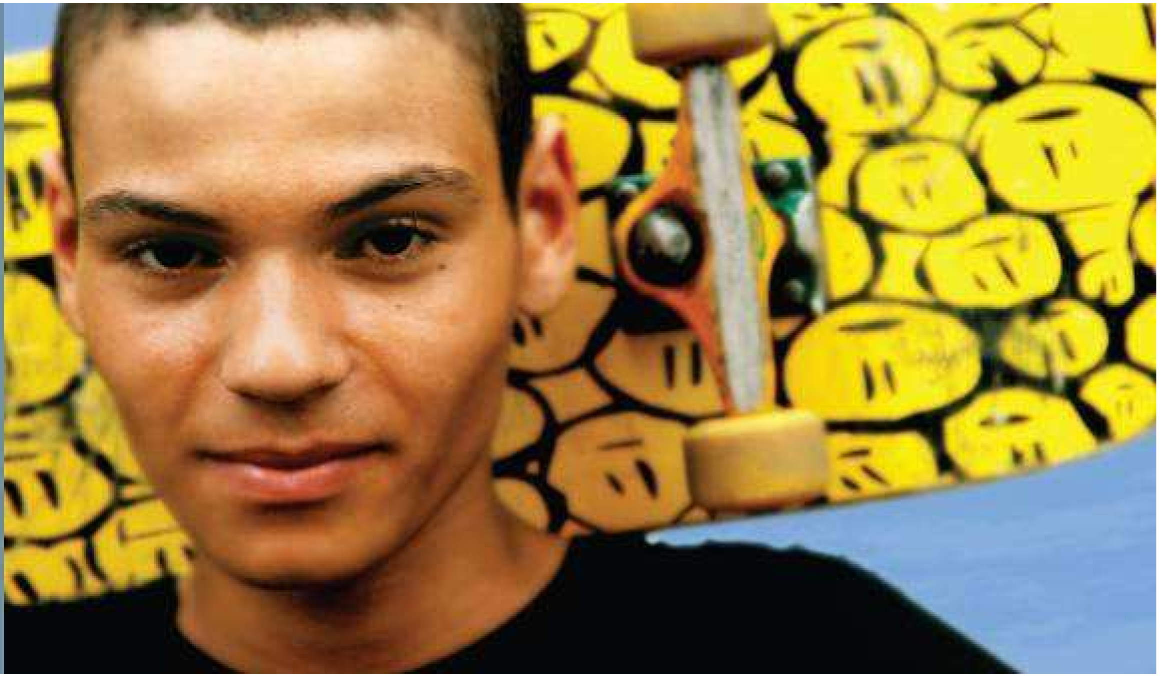
Connect Islands of IPv6 or IPv4

Translation Services



Connect to the IPv6 community

Dual Stack



Dual-Stack Techniques

- Hosts and network devices run both IPv4 and IPv6 at the same time.
  - This technique is useful as a temporary transition, but it adds overhead and uses many resources.
- Cisco IOS Software is IPv6 ready.
  - As soon as IPv4 and IPv6 configurations are complete, the interface is dual stacked and it forwards both IPv4 and IPv6 traffic.
- Drawback of dual stacking includes:
  - The additional resources required to keep and process dual routing tables, routing protocol topology tables, etc.
  - The higher administrative overhead, troubleshooting, and monitoring, is more complex.

Dual-Stack Example


```
R1(config)# interface fa0/0
R1(config-if)# ip address 10.10.10.1 255.255.255.0
R1(config-if)# ipv6 address 2001:12::1/64
R1(config-if)# ^Z
R1#
```

- The FastEthernet 0/0 interface of R1 is dual stacked.
  - It is configured with an IPv4 and an IPv6 address.
  - Also notice that for each protocol, the addresses on R1 and R2 are on the same network.

Dual-Stack Example


```
R1# show ip interface fa0/0
FastEthernet0/0 is up, line protocol is up
  Internet address is 10.10.10.1/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Outgoing access list is not set
  Inbound access list is not set
  Proxy ARP is enabled
  Local Proxy ARP is disabled
  Security level is default
  Split horizon is enabled
  ICMP redirects are always sent
  ICMP unreachables are always present

<output omitted>
```

- The output confirms that the Fa0/0 interface is operational and uses the IPv4 address.

Dual-Stack Example


```
R1# show ipv6 interface fa0/0
FastEthernet0/0 is up, line protocol is up
    IPv6 is enabled, link-local address is FE80::219:56FF:FE2C:9F60
    Global unicast address(es):
      2001:12::1, subnet is 2001:12::/64
    Joined group address(es):
      FF02::1
      FF02::2
      FF02::1:FF00:1
      FF02::1:FF2C:9F60
    MTU is 1500 bytes
    ICMP error messages limited to one every 100 milliseconds
    ICMP redirects are enabled
    ND DAD is enabled, number of DAD attempts: 1
    ND reachable time is 30000 milliseconds

<output omitted>
```

- The output confirms that the Fa0/0 interface is operational and also uses the IPv6 address.

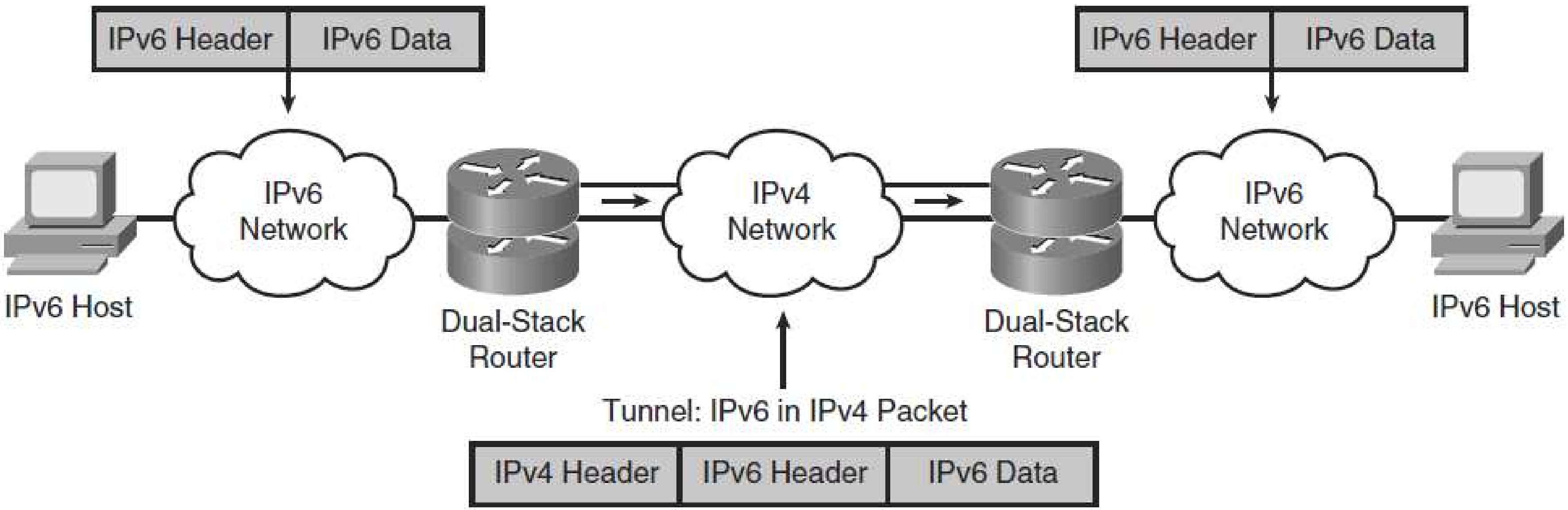
Tunneling

Tunneling Techniques

- Isolated IPv6 networks are connected over an IPv4 infrastructure using tunnels.
- The edge devices are the only ones that need to be dual-stacked.
- Scalability may be an issue if many tunnels need to be created.
  - Tunnels can be either manually or automatically configured, depending on the scale required and administrative overhead tolerated.

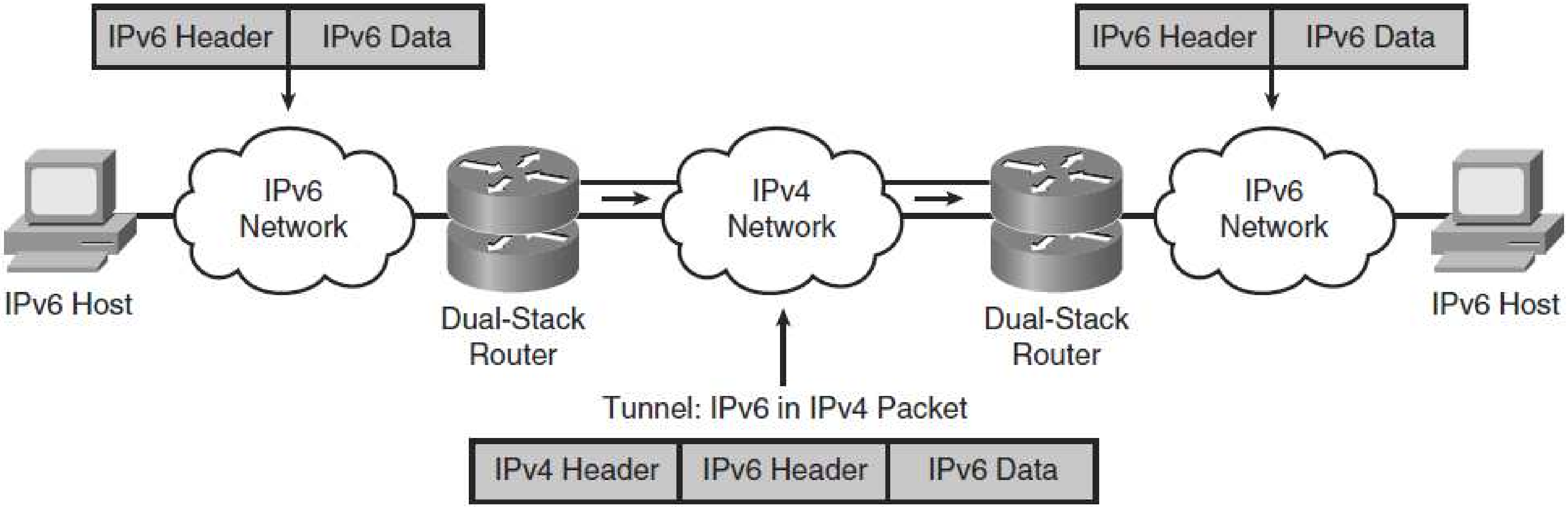
Tunneling Techniques

- For IPv6, tunneling is an integration method in which an IPv6 packet is encapsulated within IPv4.
- This enables the connection of IPv6 islands without the need to convert the intermediary network to IPv6.



Tunneling Techniques

- In this example, the tunnel between sites is using:
  - IPv4 as the transport protocol (the protocol over which the tunnel is created).
  - IPv6 is the passenger protocol (the protocol encapsulated in the tunnel and carried through the tunnel).
  - GRE is used to create the tunnel, and is known as the tunneling protocol.



Types of Tunnels

- Tunnels can be created manually using:
  - Manual IPv6 tunnels
  - GRE IPv6 tunnels (not covered in this presentation)
- Tunnels can also be created automatically using:
  - IPv4-Compatible IPv6 Tunnels (now deprecated)
  - 6to4 tunnels
  - ISATAP Tunnels

Manual Tunnels

Manual Tunnel Configuration

- Create a tunnel interface.

```
Router(config)#
interface tunnel number
```

- Creates a tunnel interface which is virtual.
- Once in interface configuration mode, configure the tunnel parameters including:
  - IP address
  - Tunnel source
  - Tunnel destination
  - Tunnel mode (type of tunnel)

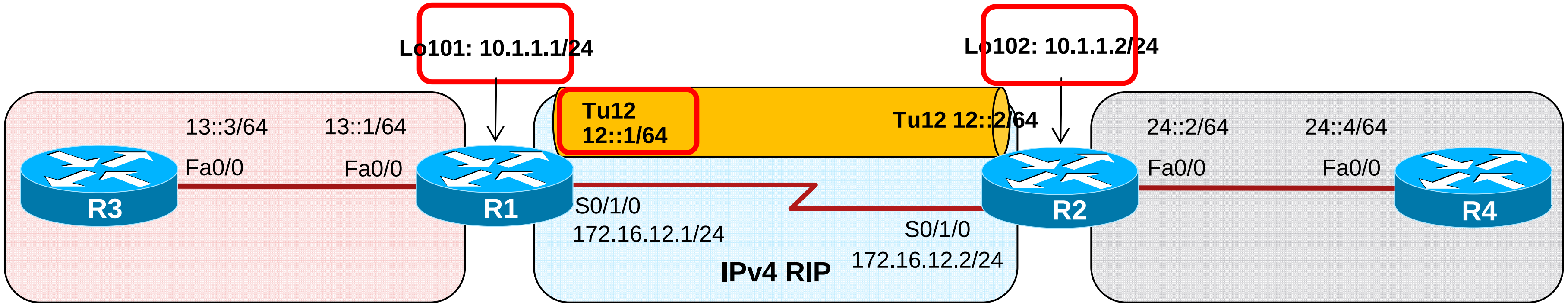
Tunnel Configuration Commands

Command	Description
tunnel source *interface-type interface-number*	An interface configuration command that sets the source address for a tunnel interface as the address of the specified interface
tunnel destination *ip-address*	An interface configuration command that specifies the destination address for a tunnel interface. In this case the *ip-address* parameter is an IPv4 address
tunnel mode ipv6ip	An interface configuration command that sets the encapsulation mode for the tunnel interface to use IPv6 as the passenger protocol, and IPv4 as both the encapsulation and transport protocol.

Tunnel Troubleshooting Commands

Command	Description
`debug tunnel`	EXEC command that enables the display of the tunnel encapsulation and decapsulation process.
`debug ip packet detail`	EXEC command that enables the display of details about IP packets traversing the router.

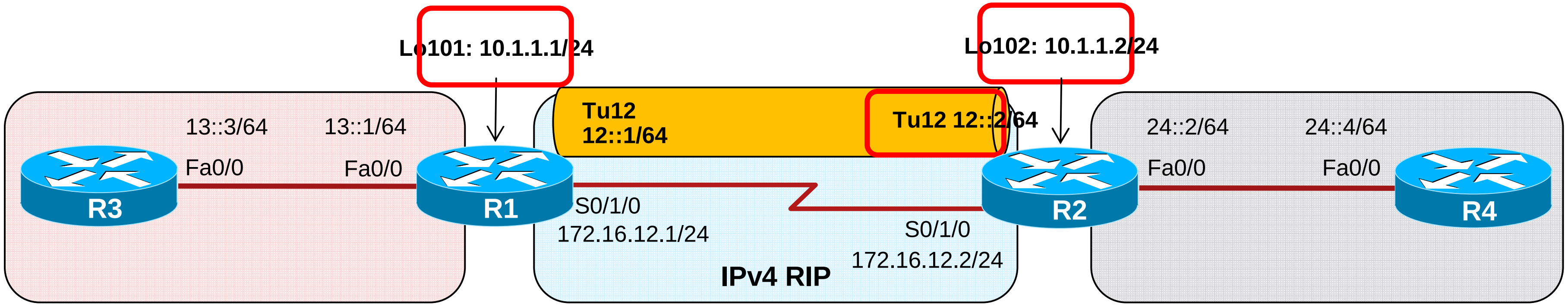
Manual IPv6 Tunnel Example




```
R1(config)# interface tunnel 12
R1(config-if)#
*Aug 16 09:34:46.643: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12,
changed state to down
R1(config-if)# no ip address
R1(config-if)# ipv6 address 12::1/64
R1(config-if)# tunnel source loopback 101
R1(config-if)# tunnel destination 10.1.1.2
R1(config-if)#
*Aug 16 09:36:52.051: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12,
changed state to up
R1(config-if)# tunnel mode ipv6ip
R1(config-if)#
```

- R1 is configured with the manual tunnel configuration.

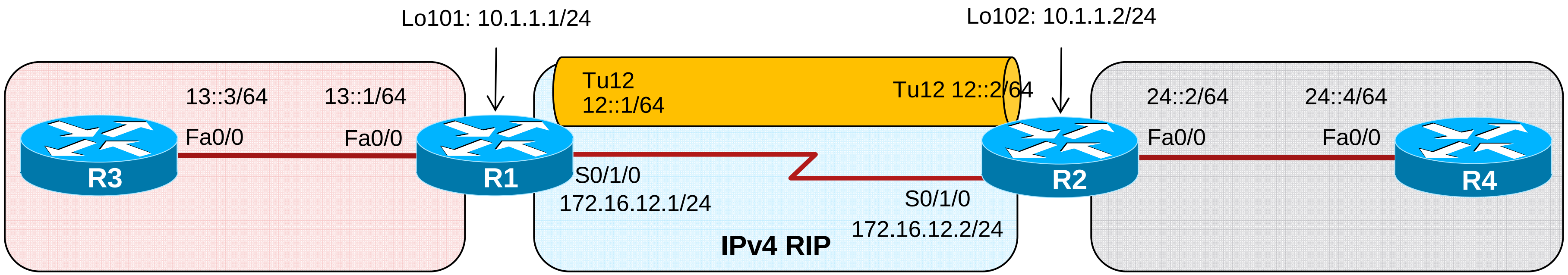
Manual IPv6 Tunnel Example




```
R2(config)# interface tunnel 12
R2(config-if)#
*Aug 16 09:38:47.532: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12,
changed state to down
R2(config-if)# no ip address
R2(config-if)# ipv6 address 12::2/64
R2(config-if)# tunnel source loopback 101
R2(config-if)# tunnel destination 10.1.1.1
R2(config-if)#
*Aug 16 09:39:24.056: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12,
changed state to up
R2(config-if)# tunnel mode ipv6ip
R2(config-if)#
```

- R2 is configured with the manual tunnel configuration.

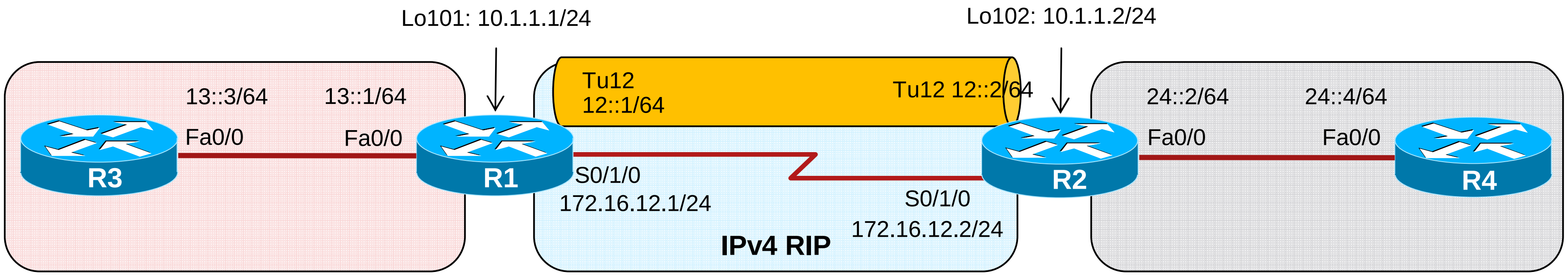
Manual IPv6 Tunnel Example




```
R1# show interface tunnel 12
Tunnel12 is up, line protocol is up
  Hardware is Tunnel
  MTU 1514 bytes,BW 9 Kbit/sec, DLY 500000 usec,
    reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive not set
  Tunnel source 10.1.1.1 (Loopback101), destination 10.1.1.2
  Tunnel protocol/transport IPv6/IP
  Tunnel TTL 255
  Fast tunneling enabled

<output omitted>
```

- The tunnel interface is examined.
- Next, RIPng will be configured to cross the tunnel.

Manual IPv6 Tunnel Example

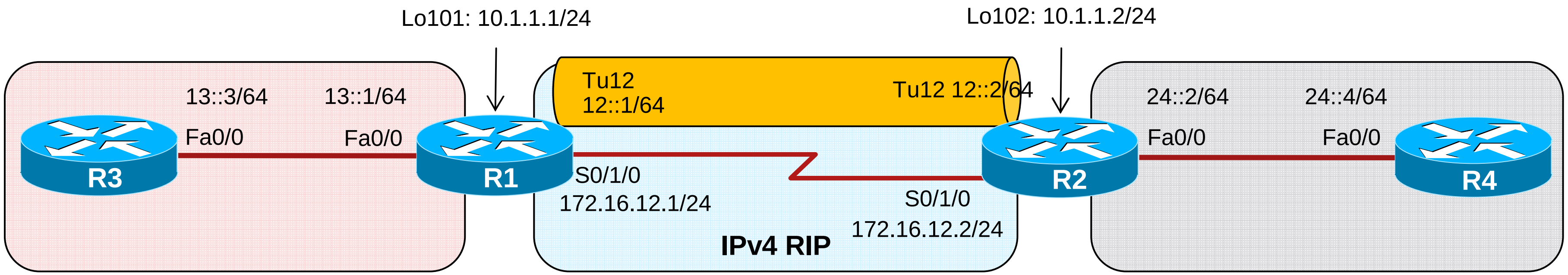



```
R1(config)# ipv6 unicast-routing
R1(config)# interface tunnel 12
R1(config-if)# ipv6 rip RIPoTU enable
R1(config-if)# interface fa0/0
R1(config-if)# ipv6 rip RIPoTU enable
R1(config-if)#
```

```
R2(config)# ipv6 unicast-routing
R2(config)# interface tunnel 12
R2(config-if)# ipv6 rip RIPoTU enable
R2(config-if)# interface fa0/0
R2(config-if)# ipv6 rip RIPoTU enable
R2(config-if)#
```

- RIPng is enabled on the tunnel interfaces and on the FastEthernet interfaces of R1 and R2.

Manual IPv6 Tunnel Example

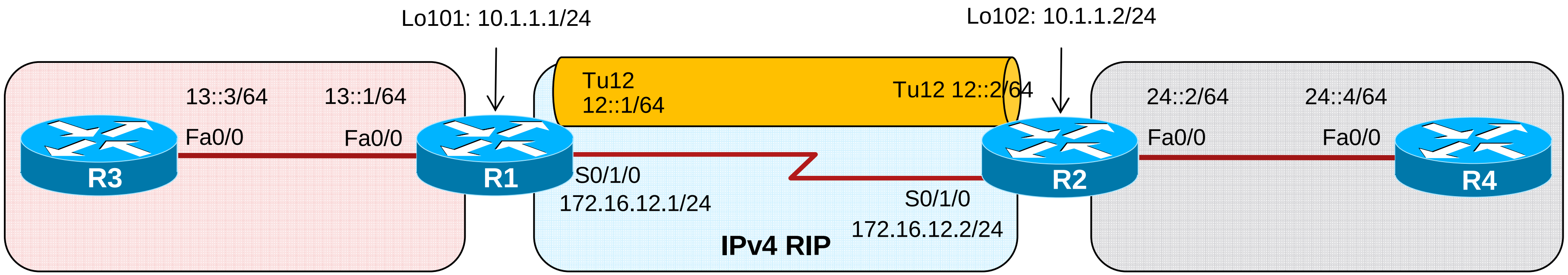



```
R3(config)# ipv6 unicast-routing
R3(config)# interface fa0/0
R3(config-if)# ipv6 rip RIPoTU enable
R3(config-if)#
```

```
R4(config)# ipv6 unicast-routing
R4(config)# interface fa0/0
R4(config-if)# ipv6 rip RIPoTU enable
R4(config-if)#
```

- RIPng is enabled on the FastEthernet interfaces of R3 and R4.
- Now end-to-end connectivity should be achieved.

Manual IPv6 Tunnel Example



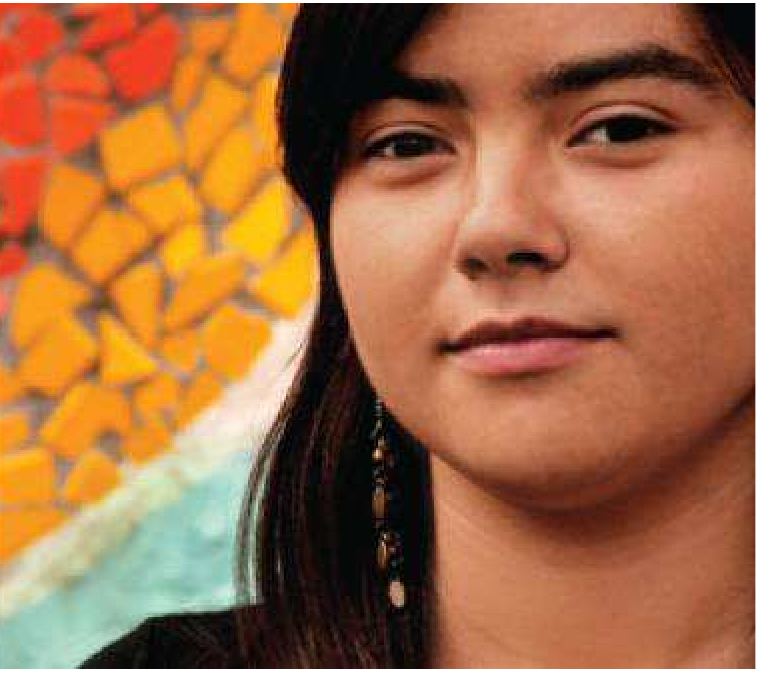

```
R4# show ipv6 route rip

<output omitted>

R   12::/64 [120/2]
     via FE80::2, FastEthernet0/0
R   13::/64 [120/3]
     via FE80::2, FastEthernet0/0
R4#
```

```
R3# ping 24::4
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 24::4, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/18/20 ms
R3#
```

Manual IPv6 Tunnel Summary

- Manual tunnels are simple to configure, and are therefore useful for a small number of sites.
- However, for large networks manual tunnels are not scalable, from both a configuration and management perspective.
- The edge routers on which the tunnels terminate need to be dual stacked, and therefore must be capable of running both protocols and have the capacity to do so.

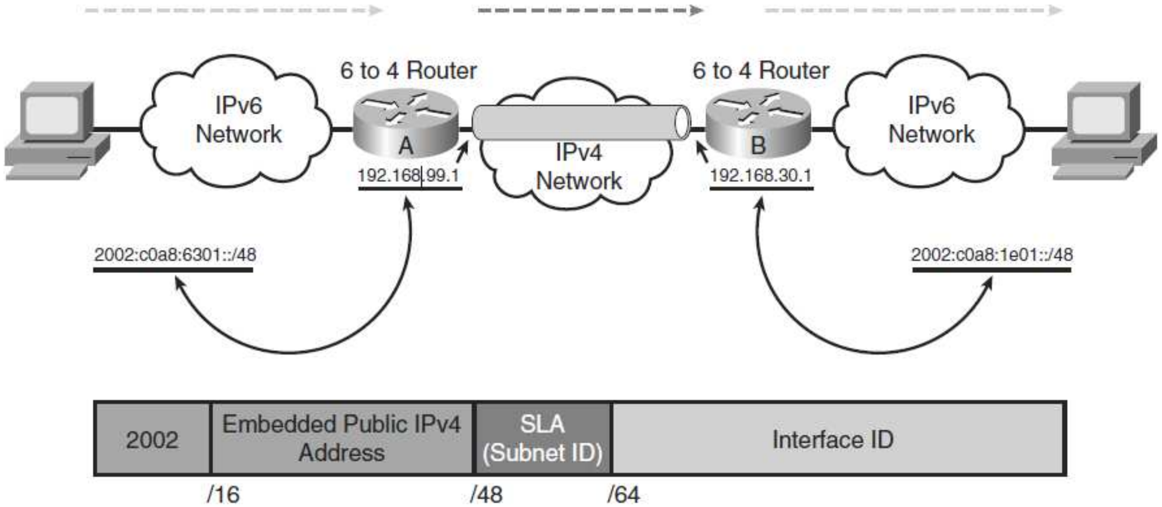
6to4 Tunnels



6to4 Tunnels

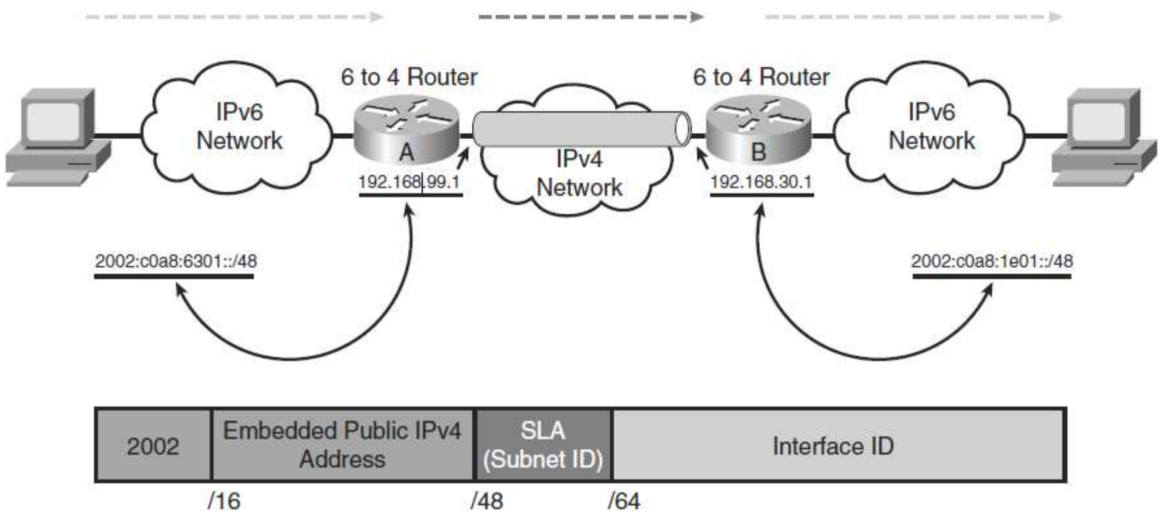
- 6to4 tunnels, also known as a 6-to-4 tunnel, is an automatic tunneling method.
- 6to4 tunnels are point-to-multipoint, rather than the point-to-point tunnels.
- The 6to4 tunnels are built automatically by the edge routers, based on embedded IPv4 address within the IPv6 addresses of the tunnel interfaces on the edge routers.
- 6to4 tunnels enable the fast deployment of IPv6 in a corporate network without the need for public IPv6 addresses from ISPs or registries.

6to4 Tunnel Example



- When Router A receives an IPv6 packet with a destination address in the range of 2002::/16 (the address 2002:c0a8:1e01::/48 in the example), it determines that the packet must traverse the tunnel.
  - The router extracts the IPv4 address embedded in the third to sixth octets, inclusively, in the IPv6 next-hop address.
  - In this example, these octets are c0a8:1e01 which is therefore 192.168.30.1.
- This IPv4 address is the IPv4 address of the 6to4 router at the destination site, Router B.

6to4 Tunnel Example

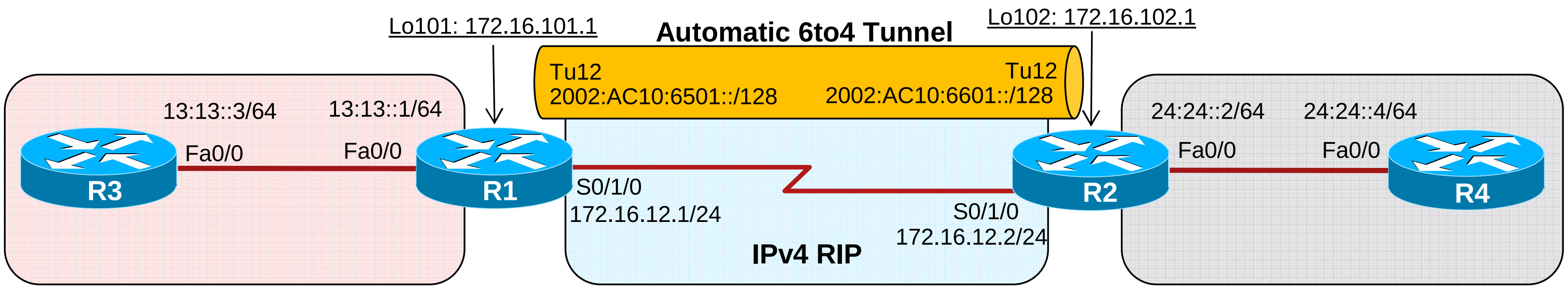


- Router A encapsulates the IPv6 packet in an IPv4 packet with Router B's extracted IPv4 address as the destination address.
  - The packet passes through the IPv4 network.
- Router B, decapsulates the IPv6 packet from the received IPv4 packet and forwards the IPv6 packet to its final destination.

6to4 Limitations

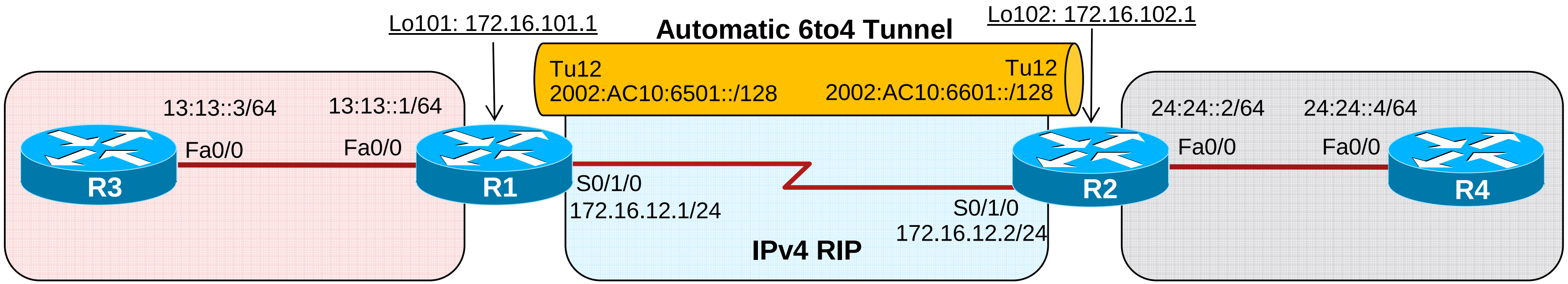
- Only static routes or BGP are supported.
  - This is because the other routing protocols use link-local addresses to form adjacencies and exchange updates and these do not conform to the address requirements for 6to4 tunnels.
  - The example presented here will use static routes.
- NAT cannot be used along the IPv4 path of the tunnel, again because of the 6to4 address requirements.

6to4 Tunnel Example



- In this example, there are two IPv6 networks separated by an IPv4 network.
- The objective of this example is to again provide full connectivity between the IPv6 islands over the IPv4-only infrastructure.
- The first step is to configure routers R1 and R2 so that they can establish the 6to4 tunnel between them.

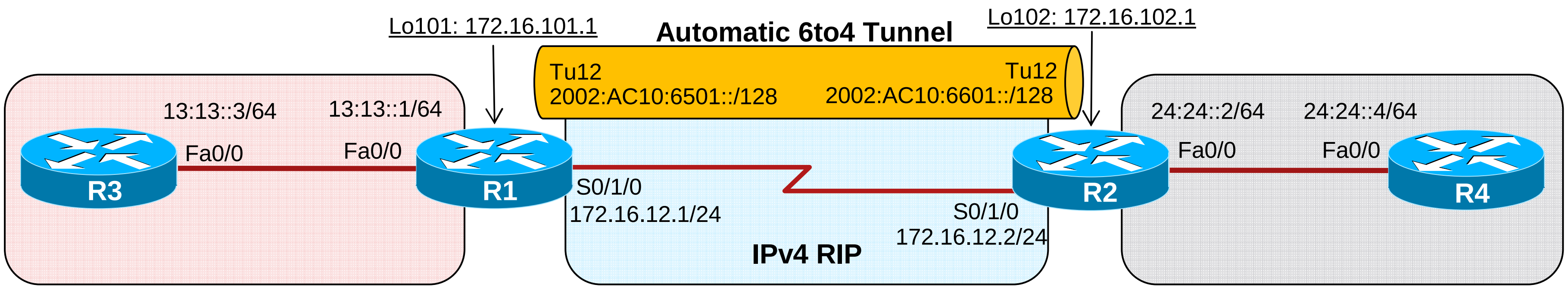
6to4 Tunnel Example




```
R1(config)# interface tunnel 12
R1(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down
R1(config-if)# no ip address
R1(config-if)# ipv6 address 2002:AC10:6501::/128
R1(config-if)# tunnel source loopback 101
R1(config-if)# tunnel mode ipv6ip 6to4
R1(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up
R1(config-if)# exit
```

- R1 is configured with the 6to4 tunnel.
  - Notice that the configuration is similar to the manual tunnel configurations except that the tunnel destination is not specified.

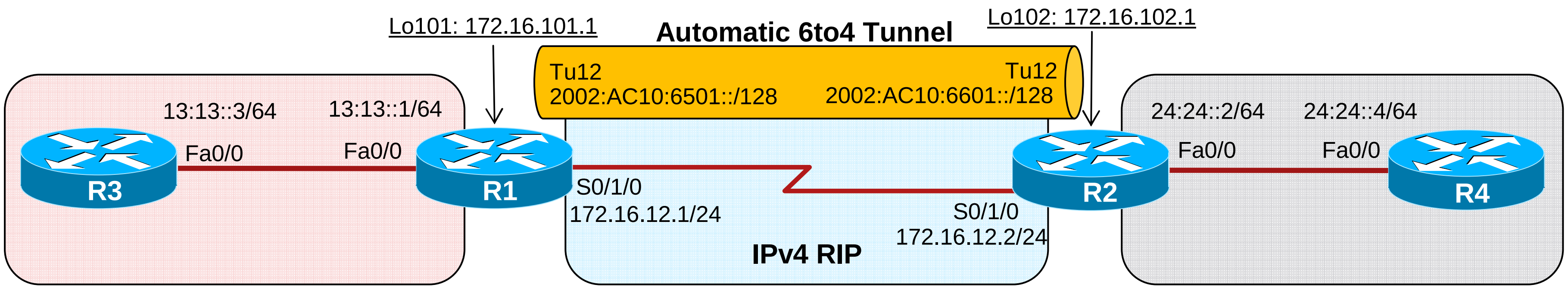
6to4 Tunnel Example




```
R1(config)# ipv6 route 2002::/16 tunnel 12
R1(config)# ipv6 route 24::/64 2002:AC10:6601::
R1(config)#
```

- R1 is configured with static routes.

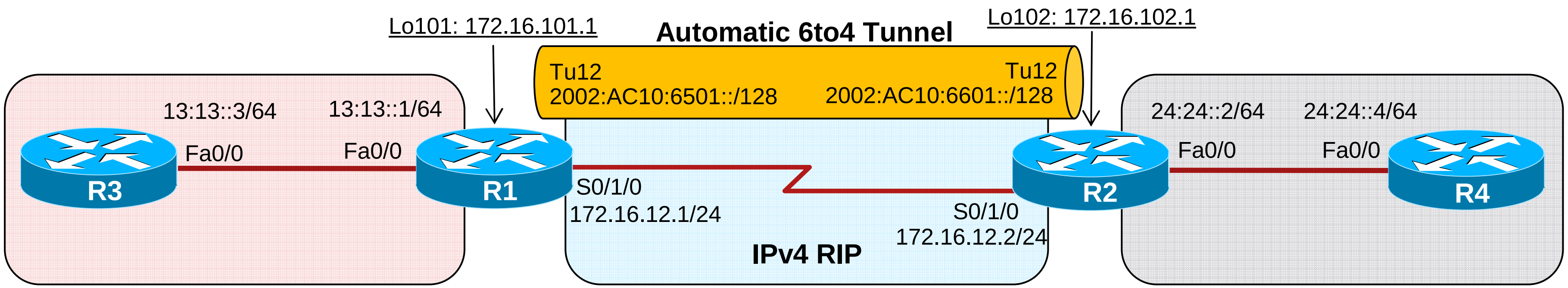
6to4 Tunnel Example




```
R2(config)# interface tunnel 12
R2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down
R2(config-if)# no ip address
R2(config-if)# ipv6 address 2002:AC10:6601::/128
R2(config-if)# tunnel source loopback 102
R2(config-if)# tunnel mode ipv6ip 6to4
R2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up
R2(config-if)# exit
```

- R2 is configured with the 6to4 tunnel.

6to4 Tunnel Example




```
R2(config)# ipv6 route 2002::/16 tunnel 12
R2(config)# ipv6 route 13::/64 2002:AC10:6501::
R2(config)#
```

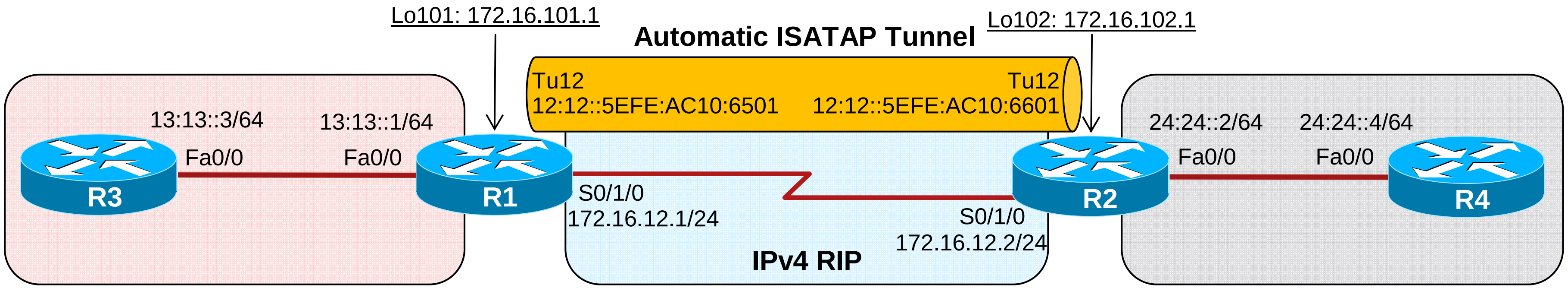
- R2 is configured with static routes.

ISATAP Tunnels

ISATAP Tunnels

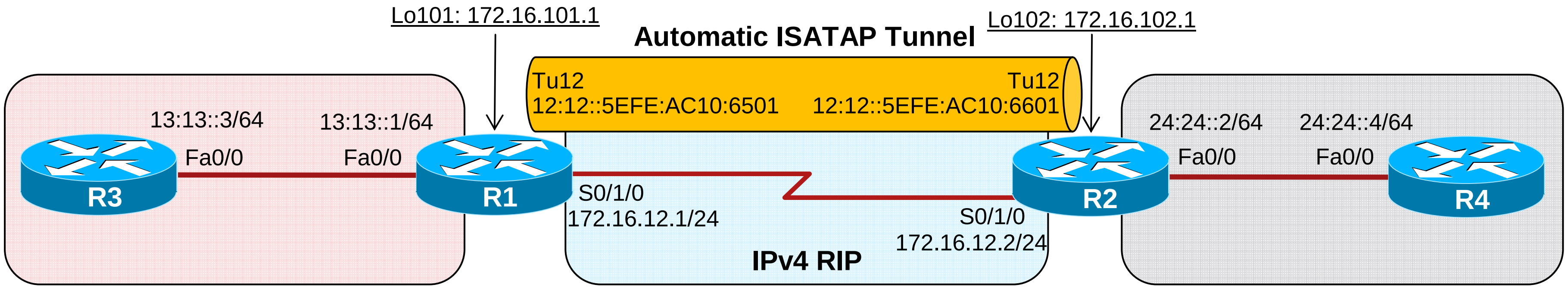
- An Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) tunnel is very similar to a 6to4 IPv6 tunnel.
  - It is used to connect IPv6 domains over an IPv4 network.
  - It embeds an IPv4 address within the IPv6 address.
- The goal of ISATAP is to provide connectivity for IPv6 hosts to a centralized IPv6-capable router, over an IPv4-only access network.
- ISATAP was designed to transport IPv6 packets within a site (hence the “intra-site” part of its name).
  - It can still be used between sites, but its purpose is within sites.
- ISATAP tunnels use IPv6 addresses consisting of a 64-bit prefix concatenated to a 64-bit interface ID in EUI-64 format.

ISATAP Tunnel Example



- In this example, there are two IPv6 networks separated by an IPv4 network.
- The objective of this example is to again provide full connectivity between the IPv6 islands over the IPv4-only infrastructure.
- The first step is to configure routers R1 and R2 so that they can establish the ISATAP tunnel between them.

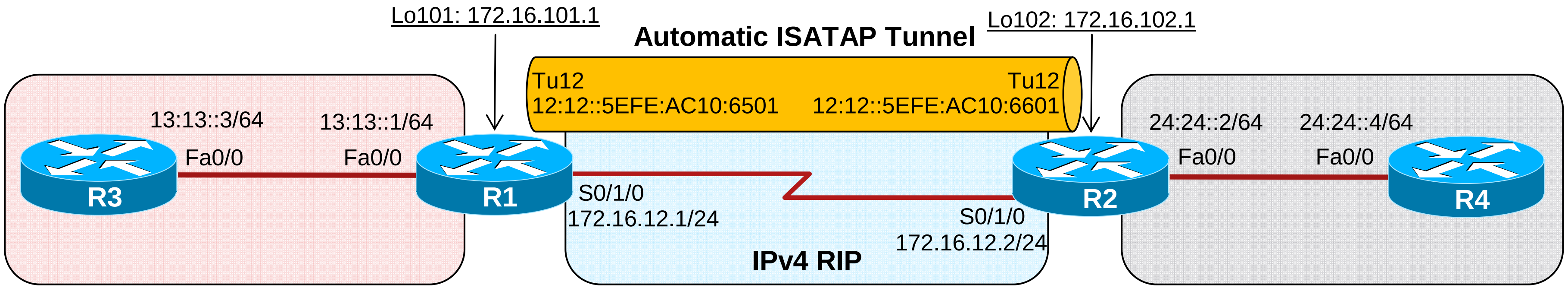
ISATAP Tunnel Example




```
R1(config)# interface tunnel 12
R1(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down
R1(config-if)# no ip address
R1(config-if)# ipv6 address 12:12::/64 eui-64
R1(config-if)# tunnel source loopback 101
R1(config-if)# tunnel mode ipv6ip isatap
R1(config-if)# exit
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up
R1(config)# ipv6 route 24::/64 tunnel12 FE80::5EFE:AC10:6601
R1(config)#
```

- R1 is configured with the ISATAP tunnel and a static route.
  - Notice that the configuration is similar to the manual and GRE tunnel configurations except that the tunnel destination is not specified.

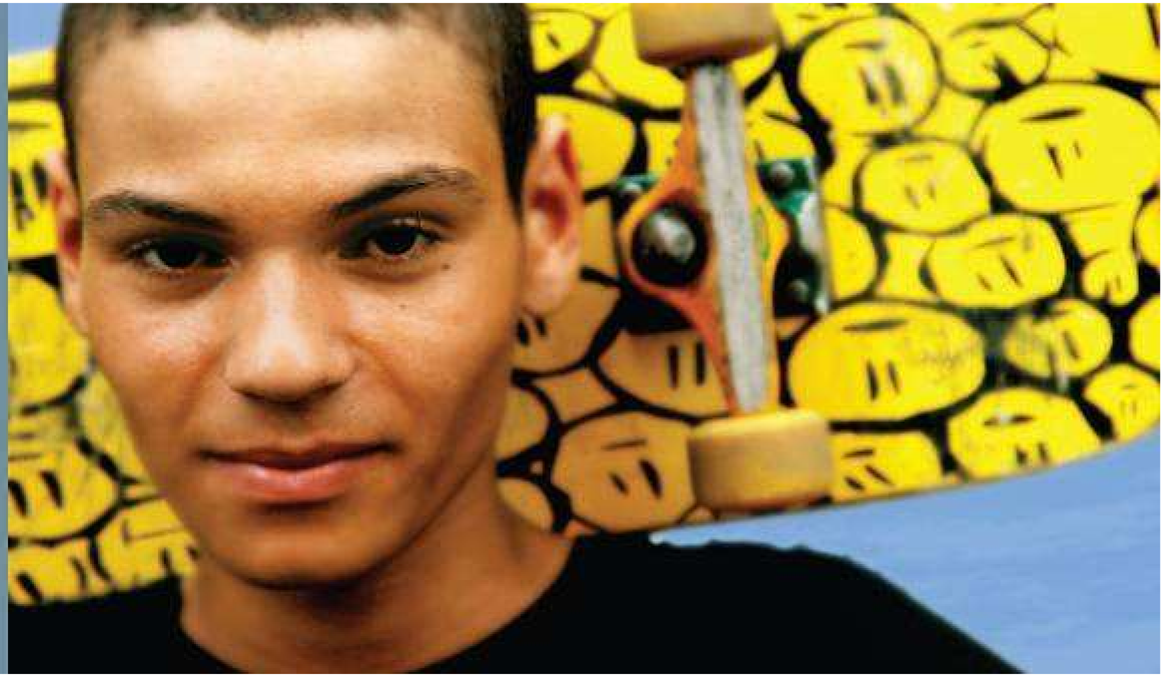
ISATAP Tunnel Example




```
R2(config)# interface tunnel 12
R2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to down
R2(config-if)# no ip address
R2(config-if)# ipv6 address 12:12::/64 eui-64
R2(config-if)# tunnel source loopback 102
R2(config-if)# tunnel mode ipv6ip isatap
R2(config-if)# exit
%LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel12, changed state to up
R2(config)# ipv6 route 13::/64 tunnel12 FE80::5EFE:AC10:6501
R2(config)#
```

- R2 is configured with the ISATAP tunnel and a static route.

Translation Using NAT-PT



NAT-PT

- NAT-PT is a transition technique, but is not a replacement for dual stack or tunneling.
  - It can be used in situations where direct communication between IPv6-only and IPv4-only networks is desired.
  - It would not be appropriate in situations where connectivity between two IPv6 networks is required, because two points of translation would be necessary, which would not be efficient or effective.
- With NAT-PT, all configuration and translation is performed on the NAT-PT router.
  - The other devices in the network are not aware of the existence of the other protocol's network, nor that translations are occurring.
- **Note**: NAT-PT has been moved to historical status with RFC 4966.

Summary

- This presentation covered transition mechanisms to aid in the transition from IPv4 to IPv6.
- **Dual Stack**
  - A device or network on which two protocol stacks have been enabled at the same time operates in dual-stack mode.
  - The primary advantage of dual-stack is that it does not require tunneling within the campus network. Dual-stack runs the two protocols as “ships-in-the-night”.
- **Tunneling**
  - A manually configured tunnel is equivalent to a permanent link between two IPv6 domains over an IPv4 backbone.
  - An automatic 6to4 tunnel allows isolated IPv6 domains to be connected over an IPv4 network to remote IPv6 networks. The key difference between automatic 6to4 tunnels and manually configured tunnels is that the tunnel is not point-to-point; it is point-to-multipoint.
  - ISATAP tunneling mechanism is similar to other automatic tunneling mechanisms, such as IPv6 6to4 tunneling; however, ISATAP is designed for transporting IPv6 packets within a site, not between sites.
- **NAT-PT**
  - NAT-PT is designed to be deployed to allow direct communication between IPv6-only networks and IPv4-only networks.
  - One of the benefits of NAT-PT is that no changes are required to existing hosts, because all the NAT-PT configurations are performed at the NAT-PT router.

Resources

- Cisco IPv6

  http://www.cisco.com/web/solutions/netsys/ipv6/index.html

- Cisco IOS IPv6 Configuration Guide

  http://www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/12_4/ipv6_12_4_book.html

- Dual-Stack At-A-Glance

  http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6553/at_a_glance_c45-625859.pdf

- Implementing Tunneling for IPv6

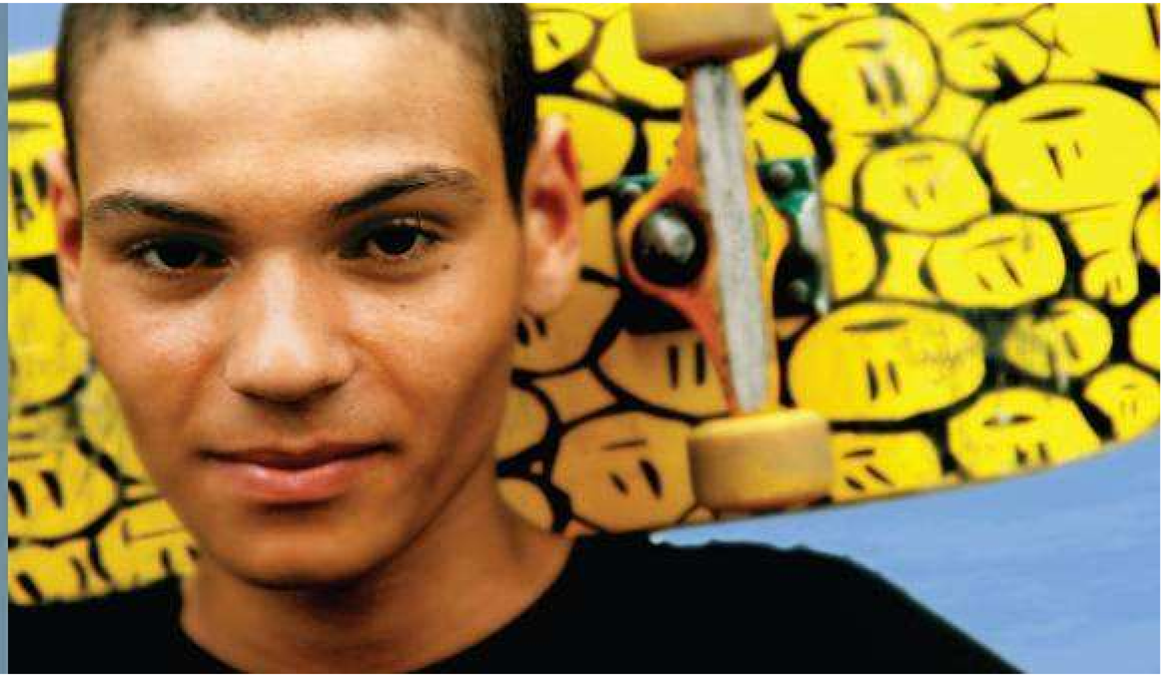
  http://www.cisco.com/en/US/docs/ios/ipv6/configuration/guide/ip6-tunnel.html

- RFC 4966

  http://www.apps.ietf.org/rfc/rfc4966.html

Cisco | Networking Academy®
Mind Wide Open™

Appendix A: Translation Using NAT-PT

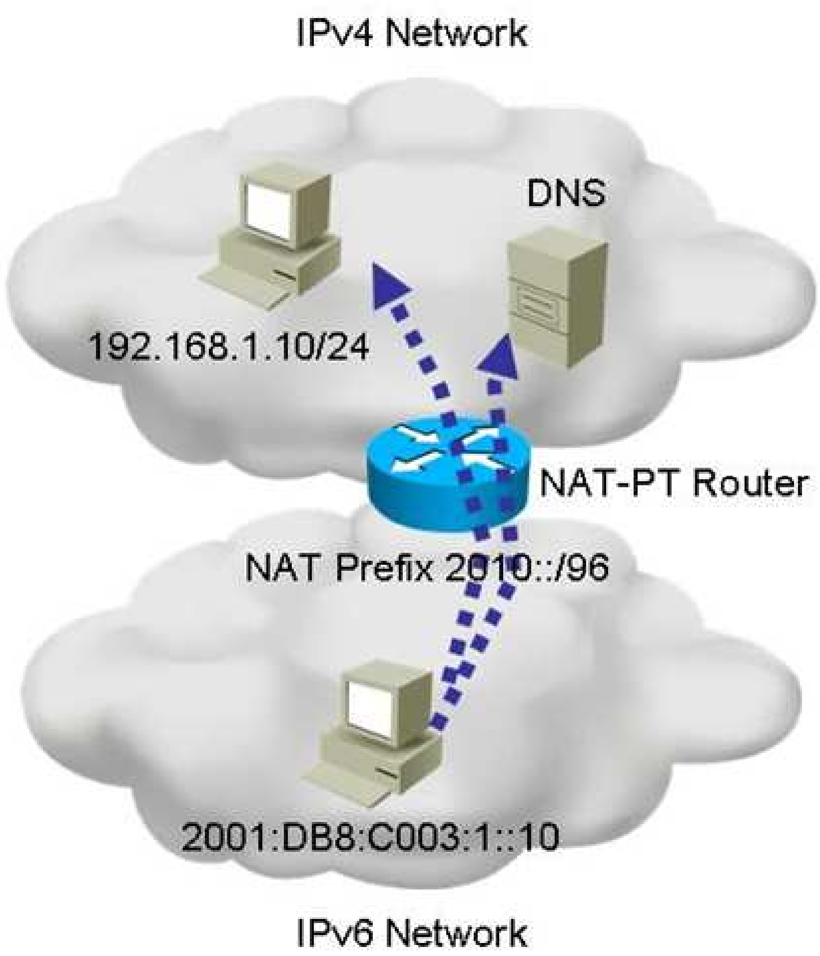


NAT-PT

- The NAT-PT router translates source and destination addresses and other packet header fields in both directions:
  - From the IPv4 network to the IPv6 network
  - From the IPv6 network to the IPv4 network.
- For this reason, this router is dual stacked and must have two sets of translation entries for this bidirectional translation.

NAT-PT Operation

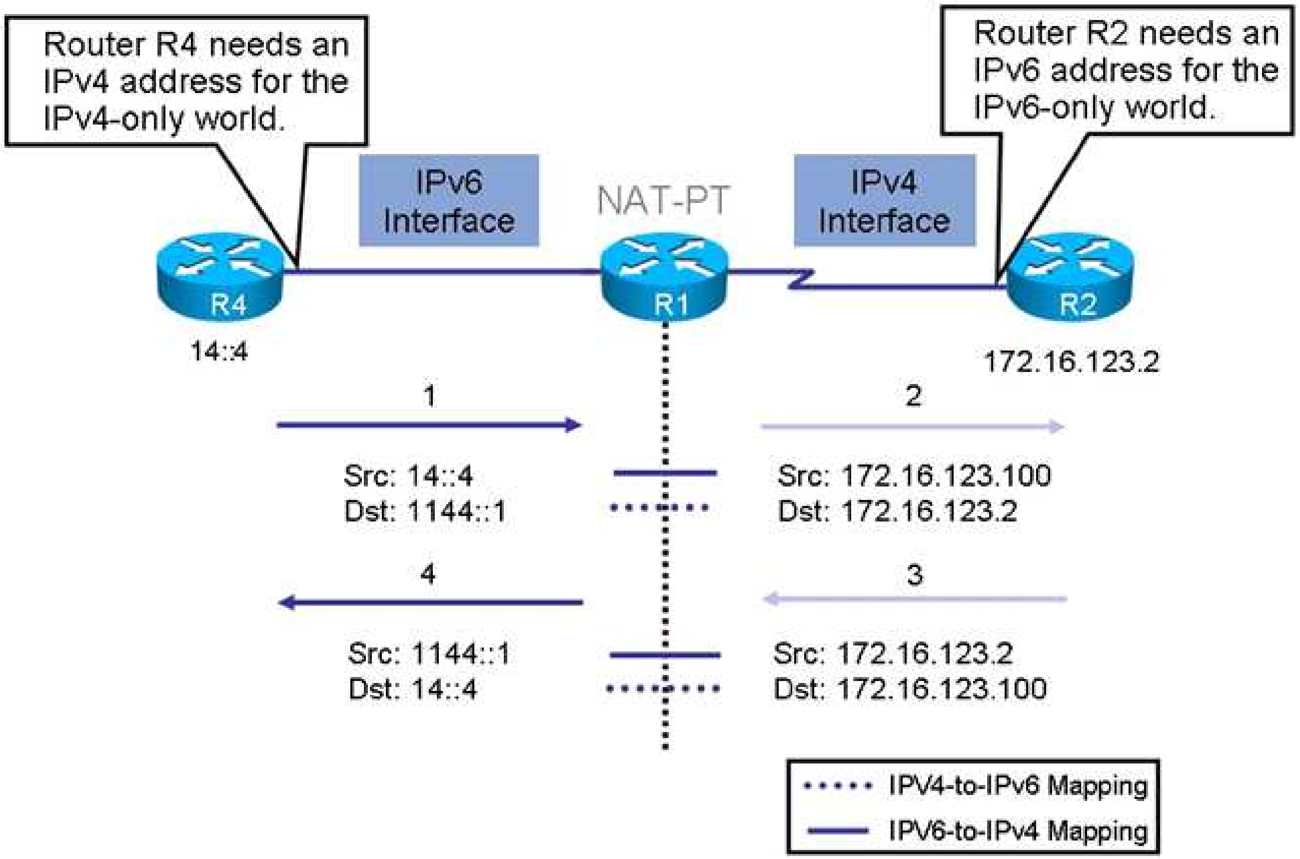
- A DNS is required in NAT-PT architectures.
  - Applications initiate traffic from hosts, and DNS translates domain names to IP addresses.
- Because DNS requests may cross the NAT-PT router, a DNS application layer gateway (ALG) is typically implemented to facilitate the name-to-address mapping.
  - The DNS-ALG translates IPv6 addresses in DNS queries and responses into their IPv4 address bindings, and vice versa.



NAT-PT

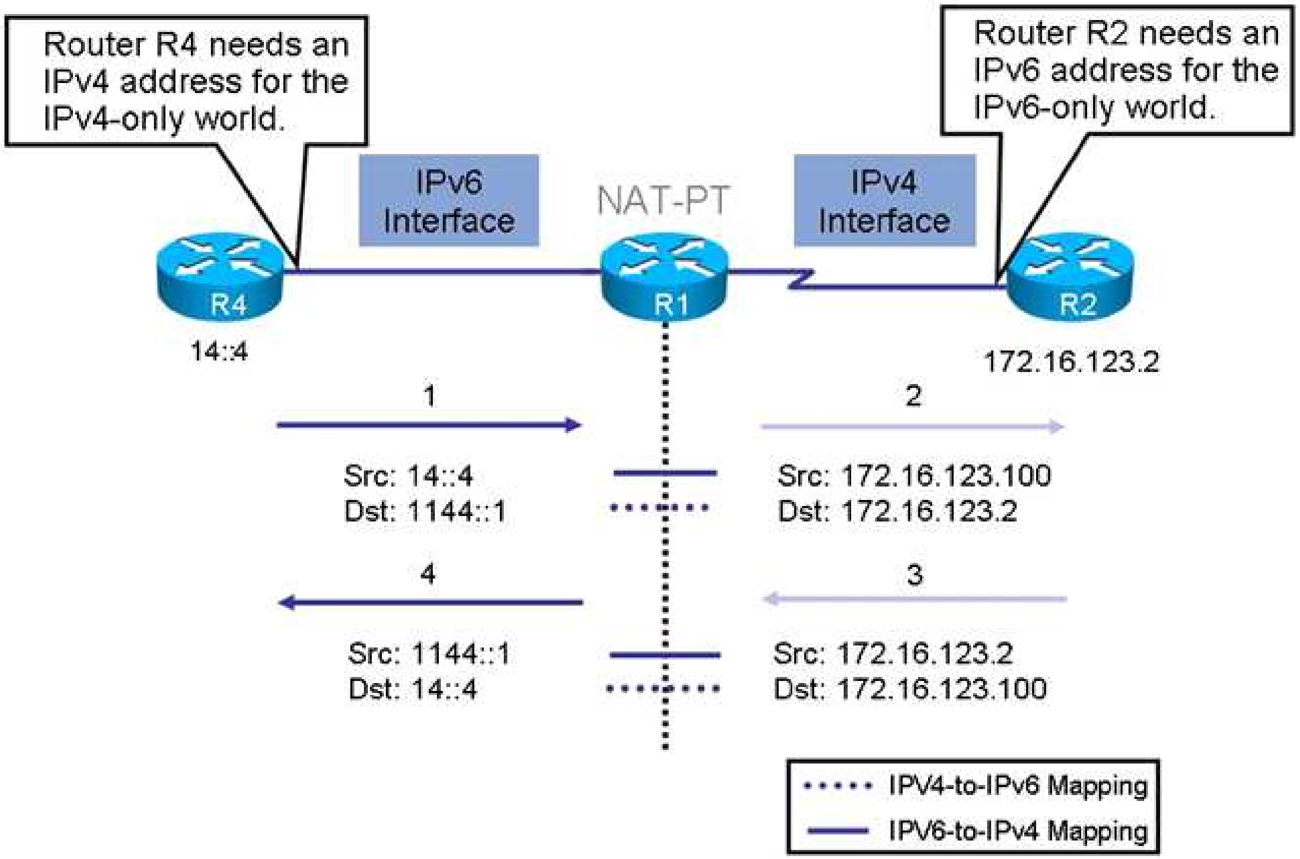
- NAT-PT uses a 96-bit IPv6 network prefix to direct all IPv6 traffic that needs to be translated to the NAT-PT router.
  - This prefix can be any routable prefix within the IPv6 domain.
  - IPv6 routing must be configured such that all IPv6 packets addressed to this prefix are routed to the NAT-PT device.
- When the NAT-PT router receives an IPv6 packet destined for the NAT-PT prefix, it translates the packet according to the configured mapping rules.
  - This prefix is also used in the translation of IPv4 addresses into IPv6 addresses.
- Within the IPv6 domain, external IPv4 addresses are mapped to IPv6 addresses.
  - This mapping is done statically or dynamically.
  - Similarly, static and dynamic mapping can be configured for translating internal IPv6 addresses to external IPv4 addresses.

Static NAT-PT for IPv6 Example



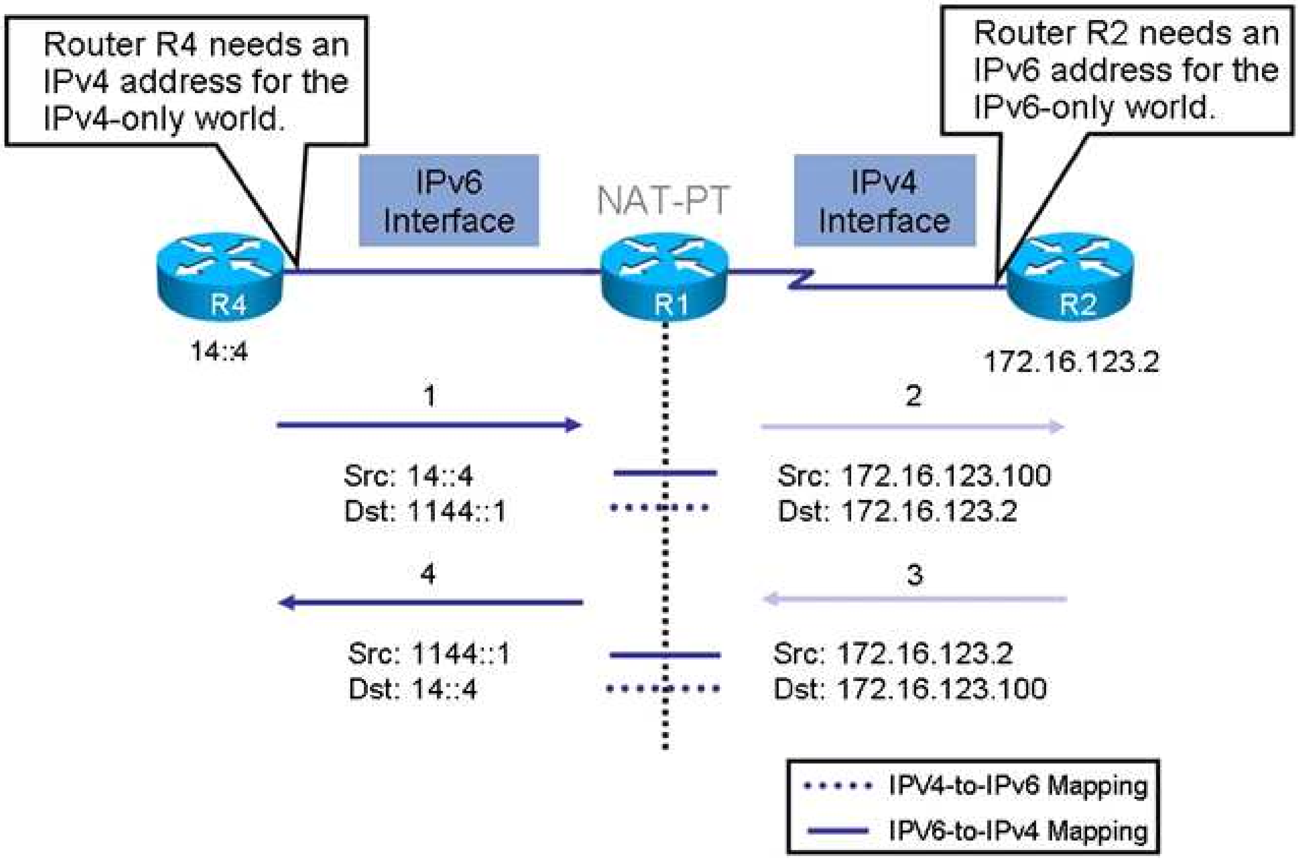
1. When R4 wants to communicate with R2, it sends an IPv6 packet (the only type it knows) with its own source address (**14::4**) and a destination address (**1144::1**) within the NAT-PT prefix.

Static NAT-PT for IPv6 Example



2. When R1 receives the IPv6 packet:
   - It looks for a static translation for the destination IPv6 address and if found, it translates it to R2's IPv4 address.
   - It also translates R4's IPv6 source address (14::4) to 172.16.123.100.

Static NAT-PT for IPv6 Example



3. When R2 replies to R4, traffic travels in the other direction.
4. When R1 receives the packet it translates the IPv4 source address (172.16.123.2) to IPv6 (1144::1) and the IPv4 destination address (172.16.123.100) to IPv6 (14::4).

Configure Static NAT-PT

- Configure IPv4-to-IPv6 static address translation using NAT-PT.

```
Router(config)#
```

```
ipv6 nat v4v6 source ipv4-address ipv6-address
```

- Configure IPv6-to-IPv4 static address translation using NAT-PT.

```
Router(config)#
```

```
ipv6 nat v6v4 source ipv6-address ipv4-address
```

Define the NAT-PT Prefix

- Define the network prefix that NAT-PT will translate.

```
Router(config)#    or   Router(config-if)#
```

```
ipv6 nat prefix ipv6-prefix/prefix-length
```

- The *ipv6-prefix/prefix-length* specifies that packets matching that address will be translated.
- It is important to note that the prefix-length must be 96.

Identify the NAT-PT Interfaces

- Identify the participating NAT-PT interfaces.

```
Router(config-if)#
```

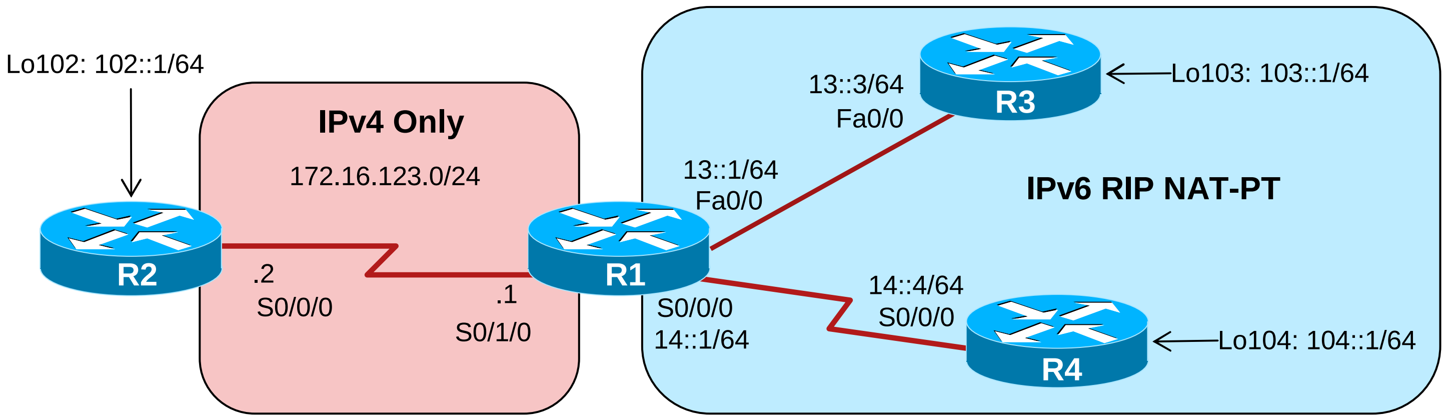
```
ipv6 nat
```

- Creates the NAT virtual interface (NVI0) and designates that traffic originating from or destined for the interface is subject to NAT-PT.
- Notice that unlike IPv4 NAT, the **`inside`** and **`outside`** keywords are not required.

Verifying and Troubleshooting NAT-PT

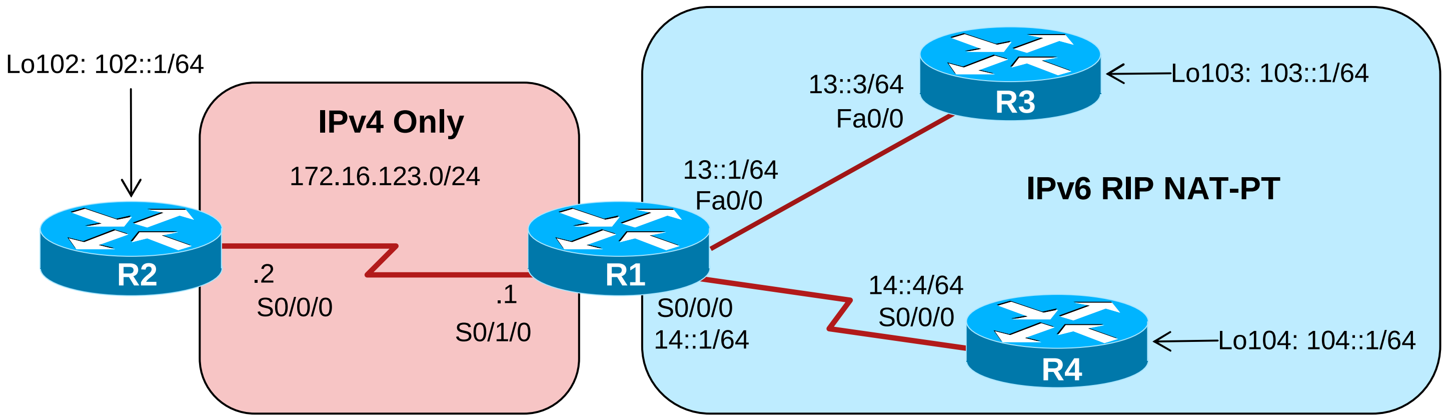
Command	Description
`show ipv6 nat translations`	Displays active NAT-PT translations. Each translation is displayed over two lines.
`show ipv6 nat statistics`	Displays NAT-PT statistics.
`debug ip icmp`	Displays ICMPv6 events in real time.
`debug ipv6 nat`	Displays debug messages for NAT-PT translation events

Static NAT-PT Example



- In this example, R3 and R4 are IPv6-only devices, and R2 is an IPv4-only device.
- R1 is the NAT-PT router.
- R4 and R2 need to communicate, therefore two static translation entries are required in R1 to allow this bidirectional communication.

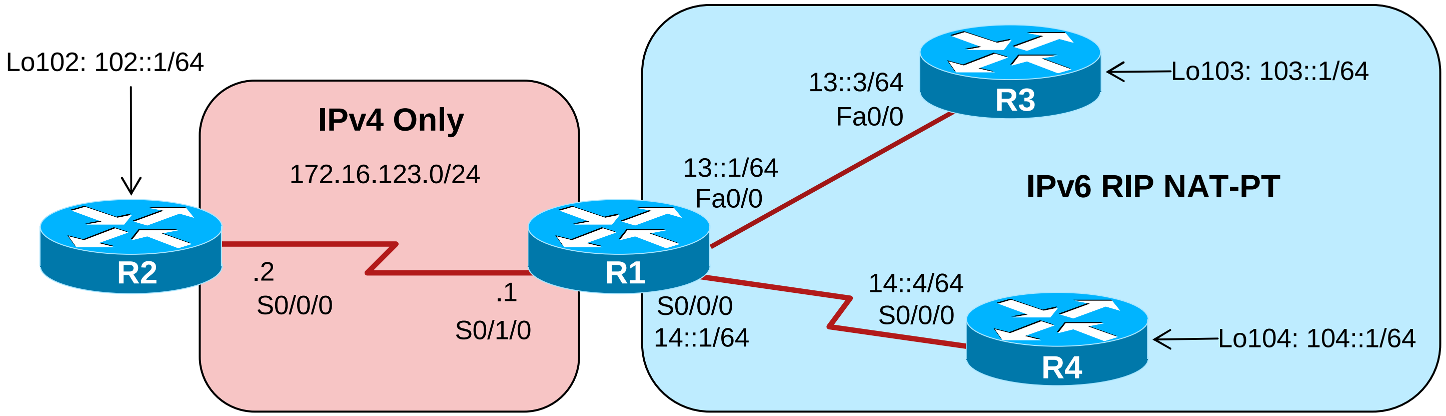
Static NAT-PT Example




```
R1(config)# interface s0/0/0
R1(config-if)# ipv6 nat
%LINEPROTO-5-UPDOWN: Line protocol on Interface NVI0, changed state to up
R1(config-if)# interface s0/1/0
R1(config-if)# ipv6 nat
R1(config-if)# exit
R1(config)#
```

- NAT-PT is enabled on the two interfaces pointing to R2 and R4.
  - Notice that the NAT virtual interface (NVI0) has been created and is active.

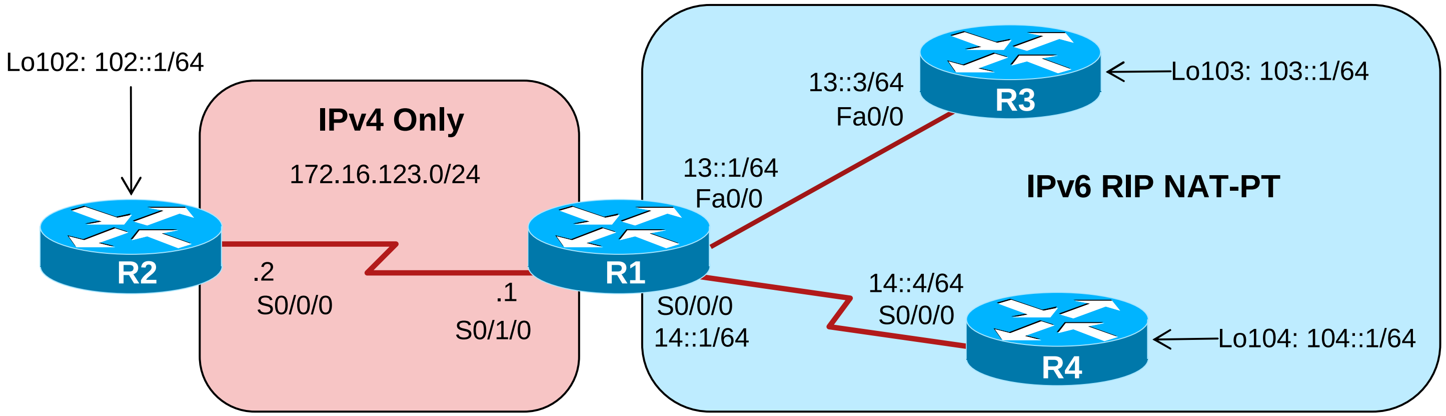
Static NAT-PT Example




```
R1(config)# ipv6 nat v6v4 source 14::4 172.16.123.100
R1(config)# ipv6 nat v4v6 source 172.16.123.2 1144::1
R1(config)#
R1(config)# ipv6 nat prefix 1144::/96
R1(config)#
```

- The IPv6-to-IPv4 and IPv4-to-IPv6 static mappings are configured.
- The last command identifies the traffic destined to the 1144::/96 prefix will be translated.

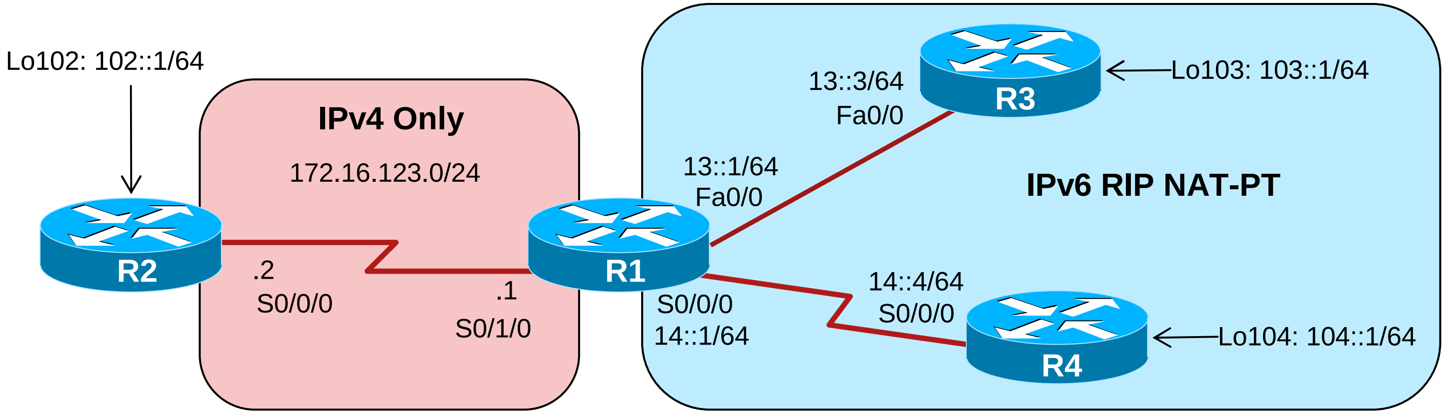
Static NAT-PT Example




```
R1# show ipv6 route connected

<output omitted>

C   13::64 [0/0]
     via FastEthernet0/0, directly connected
C   14::/64 [0/0]
     via Serial0/0/0, directly connected
C   1144::/96 [0/0]
     via NVI0, directly connected
R1#
```

- Notice that the 1144::/96 prefix appears as a directly connected route.

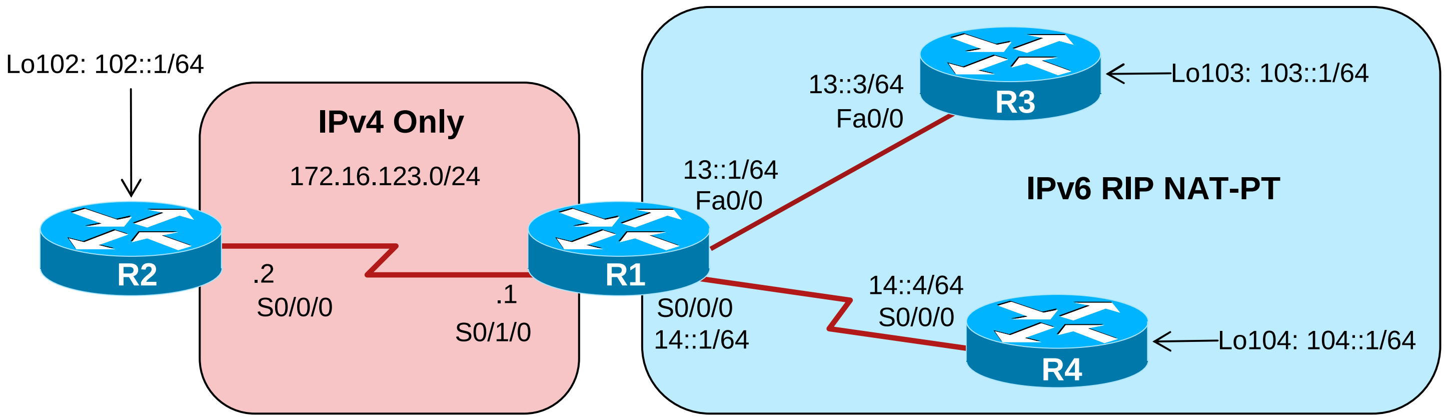
Static NAT-PT Example




```
R1# config t
R1(config)# ipv6 router rip NAT-PT
R1(config-rtr)# redistribute connected metric 3
R1(config-rtr)# exit
R1#
```

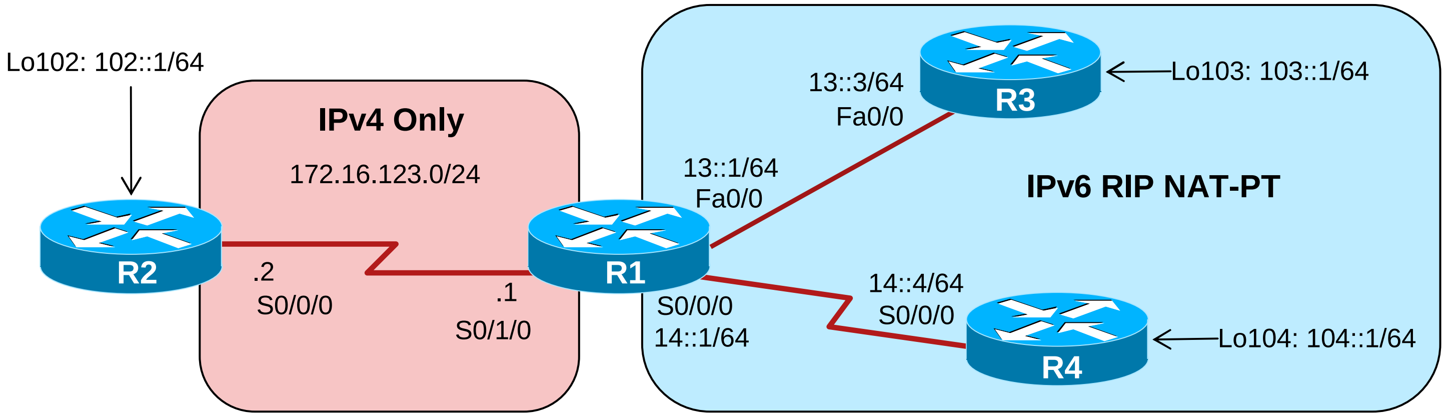
- To 1144::/96 prefix must be propagated to R4, therefore the route is redistributed with a metric of 3.

Static NAT-PT Example




```
R4# show ipv6 route rip

<ouput omitted>

R   13::/64 [120/2]
     via FE80::1, Serial 1/1.7
R   1144::/96 [120/4]
     via FE80::1, Serial 1/1.7
R4#
R4#ping 1144::1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1144::1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/70/73 ms
R4#
```

Static NAT-PT Example




```
R1# show ipv6 nat translations
Prot    IPv4 source                IPv6 source
        IPv4 destination           IPv6 destination
---     ---                         ---
        172.16.123.2                1144::1
icmp    172.16.123.100,7364         14::4, 7364
        172.16.123.2, 7364          1144::1, 7364
---     172.16.123.100              14::4
        ---                         ---
R1#
```

- Displaying the NAT translation table reveals the two static translation entries and the ICMPv6 entry created by the **`ping`** command.

Static NAT Summary

- Static NAT-PT is quite simple to configure and a good solution for one or two sites.
- Therefore a big drawback of static NAT is that it is not scalable.
  - It’s very cumbersome to create static entries for multiple sources communicating with multiple destinations.
- Dynamic NAT provides a far more scalable solution.

Dynamic NAT-PT for IPv6

- With dynamic NAT-PT, addresses are allocated from an address pool, the same as is done with IPv4 dynamic NAT.
  - Again, the commands have similar syntax to their IPv4 NAT.
- When the NAT-PT router receives a packet with an IPv6 destination address of an arbitrarily assigned 96-bit prefix (the NAT-PT prefix), it translates the IPv6 packet to an IPv4 address from an address pool.

Configure Dynamic NAT-PT

- Define a pool of IPv4 addresses for NAT-PT.

```
Router(config)#
```

```
ipv6 nat v6v4 pool name start-ipv4 end-ipv4 prefix-
  length prefix-length
```

```
R1(config)# ipv6 nat v6v4 pool POOL-12 172.16.12.100 172.16.12.101 prefix-length 24
R1(config)# ipv6 nat v6v4 pool POOL-123 172.16.123.100 172.16.123.101 prefix-length 24
R1(config)#
```

Configure Dynamic NAT-PT

- Bind an ACL with the NAT-PT pool.

Router(config)#

```
ipv6 nat v6v4 source {list {access-list-number | name}
  pool name}
```

```
R1(config)# ipv6 access-list LOOPBACK
R1(config-ipv6-acl)# permit ipv6 104::/64 any
R1(config-ipv6-acl)# permit ipv6 103::/64 any
R1(config-ipv6-acl)# exit
R1(config)# ipv6 access-list PHYSICAL
R1(config-ipv6-acl)# permit ipv6 13::/64 any
R1(config-ipv6-acl)# permit ipv6 14::/64 any
R1(config-ipv6-acl)# exit
R1(config)#
R1(config)# ipv6 nat v6v4 source list LOOPBACK pool POOL-12
R1(config)# ipv6 nat v6v4 source list PHYSICAL pool POOL-123
R1(config)#
```

Configure Dynamic NAT-PT

- Define a pool of IPv6 addresses for NAT-PT.

Router(config)#

```
ipv6 nat v4v6 pool name start-ipv6 end-ipv6 prefix-
  length prefix-length
```

```
R1(config)# ipv6 nat v4v6 pool POOL-1144 1144::1 1144::2 prefix-length 96
R1(config)#
```

Configure Dynamic NAT-PT

- Bind an ACL with the NAT-PT pool.

```
Router(config)#
```

```
ipv6 nat v4v6 source {list {access-list-number | name}
  pool name}
```

```
R1(config)# ip access-list standard IPV4
R1(config-std-nacl)# permit 172.16.123.0 0.0.0.255
R1(config-std-nacl)# permit 172.16.12.0 0.0.0.255
R1(config-std-nacl)# exit
R1(config)# ipv6 nat prefix 1144::/96
R1(config)#
R1(config)# ipv6 nat v4v6 source list IPV4 pool POOL-1144
R1(config)#
```