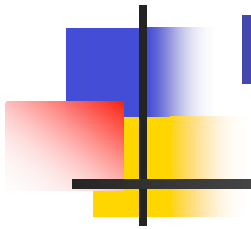


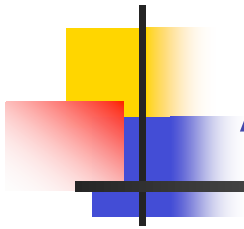
IPv6 Addressing & Routing Protocols

Philip Smith

MENOG 3

15th - 17th April 2008

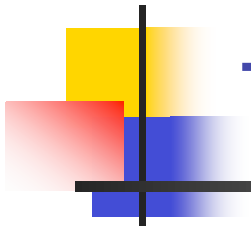
Salmiya, Kuwait



Acknowledgements

- Thanks to Ron Bonica of Juniper for the JunOS configuration examples

- Presentation slides are at:
  - ftp://ftp-eng.cisco.com/pfs/seminars/MENOG3-IPv6-Routing-Tutorial.pdf



Topics & Goals

- Addressing plans for IPv6
- Configuring IPv6
- IPv6 Routing Protocols

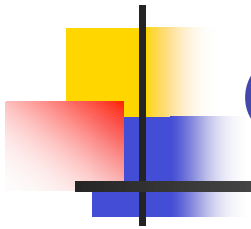
- Configuration examples including CLI from:
  - Cisco: IOS & IOS-XR
  - Juniper: JunOS

Addressing

Where to get IPv6 addresses

- The Regional Internet Registries:
  - Africa
    - AfriNIC – http://www.afrinic.net
  - Asia and the Pacific
    - APNIC – http://www.apnic.net
  - North America
    - ARIN – http://www.arin.net
  - Latin America and the Caribbean
    - LACNIC – http://www.lacnic.net
  - Europe and Middle East
    - RIPE NCC – http://www.ripe.net/info/ncc
- From your upstream ISP
- Use 6to4

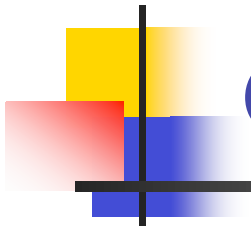
Internet Registry Regions



Getting IPv6 address space

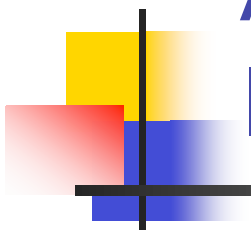
- Become a member of your Regional Internet Registry and get your own allocation
  - Require a plan for a year ahead
  - IPv6 allocation policies are documented on each RIR website
- There is plenty of IPv6 address space
  - The RIRs require high quality documentation

Getting IPv6 address space



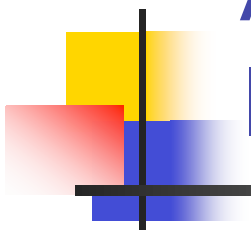
- From your upstream ISP
  - Get one /48 from your upstream ISP
  - More than one /48 if you have more than 65k subnets
- Use 6to4
  - Take a single public IPv4 /32 address
  - 2002:<ipv4 /32 address>::/48 becomes your IPv6 address block, giving 65k subnets
  - Requires a 6to4 gateway

Addressing Plans – ISP Infrastructure

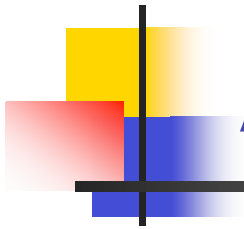


- Address block for router loop-back interfaces
  - Generally number all loopbacks out of **one** /64
  - /128 per loopback
- Address block for infrastructure
  - /48 allows 65k subnets
  - /48 per PoP or region (for large networks)
  - /48 for whole backbone (for small to medium networks)
  - Summarise between sites if it makes sense

Addressing Plans – ISP Infrastructure



- What about LANs?
  - /64 per LAN
- What about Point-to-Point links?
  - Expectation is that /64 is used
  - People have used /126s
    - Mobile IPv6 Home Agent discovery won't work
  - People have used /112s
    - Leaves final 16 bits free for node IDs
  - See RFC3627 for more discussion

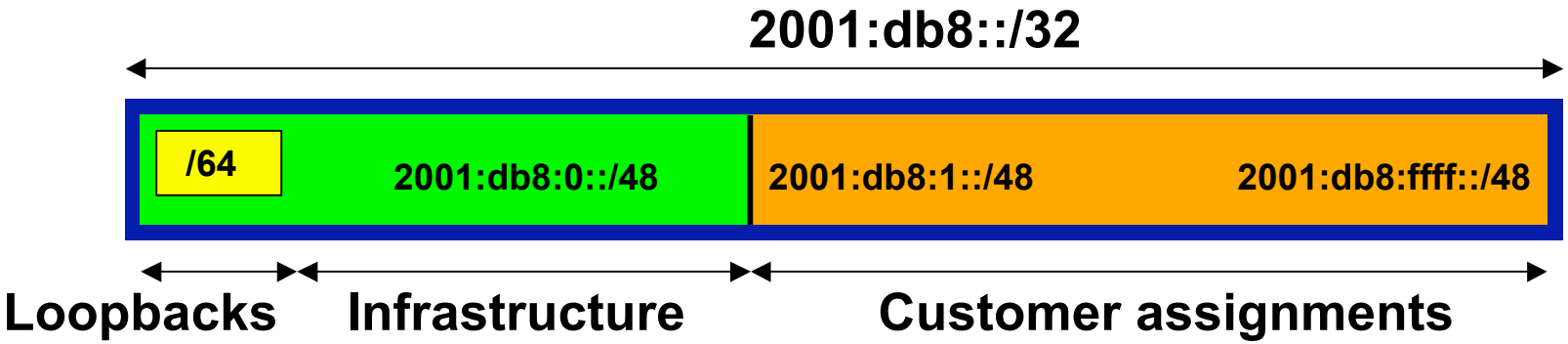


Addressing Plans – Customer

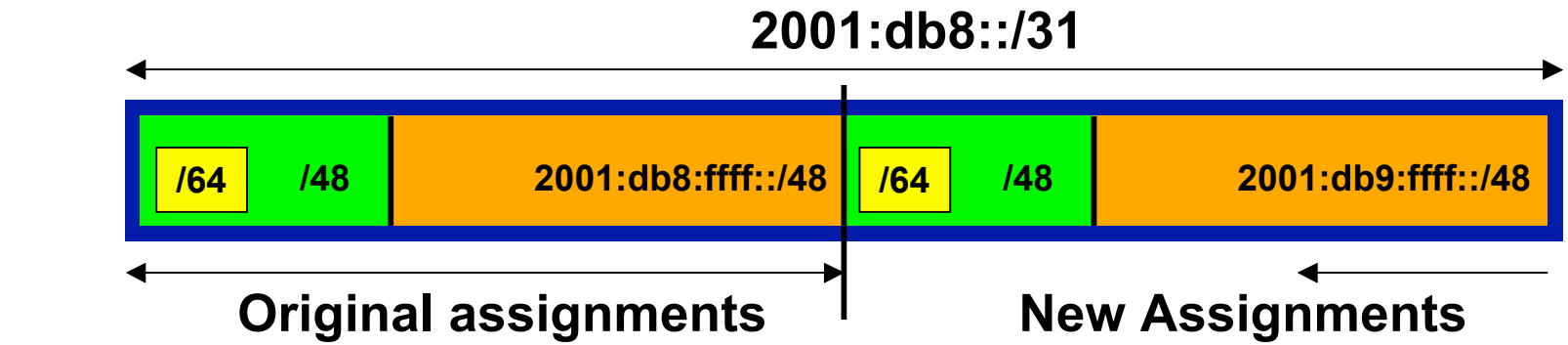
- Customers get **one** /48
  - Unless they have more than 65k subnets in which case they get a second /48 (and so on)
- Should not be reserved or assigned on a per PoP basis
  - ISP iBGP carries customer nets
  - Aggregation within the iBGP not required and usually not desirable
  - Aggregation in eBGP is very necessary

Addressing Plans – ISP Infrastructure

- Phase One

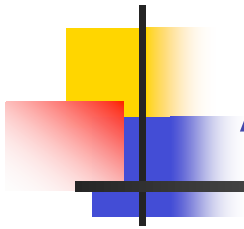


- Phase Two – second /32



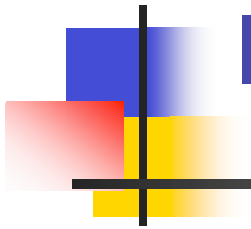
Addressing Plans Planning

- Registries will usually allocate the next block to be contiguous with the first allocation
  - Minimum allocation is /32
  - Very likely that subsequent allocation will make this up to a /31
  - So plan accordingly



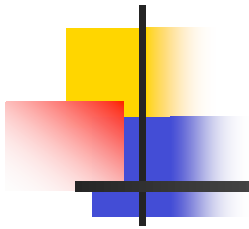
Addressing Plans (contd)

- Document infrastructure allocation
  - Eases operation, debugging and management
- Document customer allocation
  - Customers get /48 each
  - Prefix contained in iBGP
  - Eases operation, debugging and management
  - Submit network object to RIR Database



Initial IPv6 Configuration

Getting Started…



IPv6 Configuration – IOS/IOS-XR

- Enabling IPv6:
  - On by default in IOS-XR
  - For IOS, enable IPv6 using:

```
Router(config)# ipv6 unicast-routing
```

- Configuring interfaces:
  - A global or unique local IPv6 address:

```
Router(config-if)# ipv6 address X:X..X:X/prefix
```

  - An EUI-64 based IPv6 address (not so useful on a router):

```
Router(config-if)# ipv6 address X:X::/prefix eui-64
```

IPv6 Configuration – JunOS

- Enabling IPv6:
  - On by default
- Configuring interfaces:
  - A global or unique local IPv6 address:

```
interfaces {
  fe-3/0/0 {
    unit 0 {
      family inet6 {
        address 2001:db8:1::45c/64;
      }
    }
  }
}
```

IPv6 Configuration – JunOS

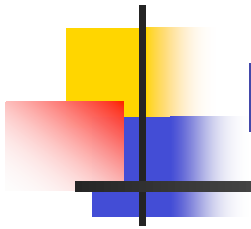
- Configuring interfaces:
  - Dual Stack:

```
interfaces {
  fe-3/0/0 {
    unit 0 {
      family inet {
        address 10.1.1.1/24;
      }
      family inet6 {
        address 2001:db8:1::45c/64;
      }
    }
  }
}
```

IPv6 Configuration – JunOS

- Configuring interfaces:
  - An EUI-64 based IPv6 address :

```
interfaces {
  fe-3/0/0 {
    unit 0 {
      family inet6 {
        address 2001:db8:1::45c/64 eui-64;
      }
    }
  }
}
```



IPv6 Configuration – Services

- Nameserver, syslog etc can be IPv6 accessible
  - IOS

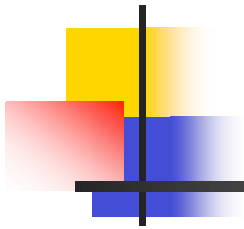
```
ip nameserver 2001:db8:2:1::2
ip nameserver 10.1.40.40
```

  - IOS-XR

```
domain name-server 2001:db8:2:1::2
domain name-server 10.1.40.40
```

  - JunOS:

```
system {
  name-server {
    2001:db8:2:1::2;
    10.1.40.40;
  }
}
```



IPv6 Configuration

- Note that by configuring an IPv6 address you will have a global or unique-local IPv6 address and a link-local IPv6 address which is

  `FE80::interface-id`

- The local-link IPv6 address is constructed automatically by concatenating FE80 with Interface ID as soon as IPv6 is enabled on the interface either by assigning an IPv6 address or simply by enabling IPv6 on the interface:

  `Router(config-if)# ipv6 enable`

IOS IPv6 Interface Status – Link Local

```
Router1# conf t
Router1(config)# ipv6 unicast-routing
Router1(config)# ^Z

Router1#sh ipv6 interface
Ethernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::A8BB:CCFF:FE00:1E00
  No global unicast address is configured
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1E00
  MTU is 1500 bytes
  ICMP error messages limited to one every 100 milliseconds
  ICMP redirects are enabled
```

IOS IPv6 Interface Status

```
Router1#sh ipv6 interface eth0/0
Ethernet0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80::A8BB:CCFF:FE00:1E00
  Global unicast address(es):
    2001:DB8::A8BB:CCFF:FE00:1E00, subnet is 2001:DB8::/64 [EUI]
  Joined group address(es):
    FF02::1
    FF02::2
    FF02::1:FF00:1E00
  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
  ND advertised reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  ND router advertisements are sent every 200 seconds
  ND router advertisements live for 1800 seconds
  Hosts use stateless autoconfig for addresses.
```

IOS-XR IPv6 Interface Status

```
RP/0/0/CPU0:as4byte#sh ipv6 interface gig 0/2/0/1
GigabitEthernet0/2/0/1 is Up, line protocol is Up, Vrfid is 0x60000000
  IPv6 is enabled, link-local address is fe80::204:6dff:fea2:90fd
  Global unicast address(es):
     2001:db8::204:6dff:fea2:90fd, subnet is 2001:db8::/64
  Joined group address(es): ff02::6 ff02::5 ff02::2
      ff02::1
  MTU is 1514 (1500 is available to IPv6)
  ICMP redirects are disabled
  ICMP unreachables are enabled
  ND DAD is enabled, number of DAD attempts 1
  ND reachable time is 0 milliseconds
  ND advertised retransmit interval is 0 milliseconds
  Hosts use stateless autoconfig for addresses.
  Outgoing access list is not set
  Inbound  access list is not set
```

JunOS IPv6 Interface Status

```
regress@UI-J6300-2> show interfaces fe-3/0/0
 Logical interface fe-3/0/0.0 (Index 68) (SNMP ifIndex 42)
                    . . .
   Flags: SNMP-Traps Encapsulation: ENET2
   Input packets : 70
   Output packets: 79
   Protocol inet, MTU: 1500
     Flags: None
     Addresses, Flags: Is-Preferred Is-Primary
       Destination: 1.1.1/24, Local: 1.1.1.2, Broadcast: 1.1.1.255
   Protocol inet6, MTU: 1500
     Flags: Is-Primary
     Addresses, Flags: Is-Preferred
       Destination: fe80::/64, Local: fe80::205:85ff:fec7:683c
     Addresses, Flags: Is-Default Is-Preferred Is-Primary
       Destination: 2001:db8:2:1::/64, Local: 2001:db8:2:1::2
```

Routing Protocols

Static Routing – IOS

- Syntax is:

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

- Static Route

```
ipv6 route 2001:db8::/64 2001:db8:0:CC00::1  110
```

  - Routes packets for network 2001:db8::/64 to a networking device at 2001:db8:0:CC00::1 with an administrative distance of 110

Static Routing – Cisco IOS-XR

- Syntax is:

```
router static
 address-family ipv6 unicast
  ipv6-prefix/prefix-length {ipv6-address |
 interface-type interface-number} [admin-distance]
```

- Static Route

```
router static
 address-family ipv6 unicast
  2001:db8::/64 2001:db8:0:CC00::1  110
```

  - Routes packets for network 2001:db8::/64 to a networking device at 2001:db8:0:CC00::1 with an administrative distance of 110

Static Routing – Juniper JunOS

- Syntax is:

```
[edit routing-options rib inet6.0 ]
   static {
       defaults {
           static-options;
       }
       rib-group group-name;
       route destination-prefix {
           next-hop;
           qualified-next-hop address {
               metric metric;
               preference preference;
           }
           static-options;
       }
   }
```

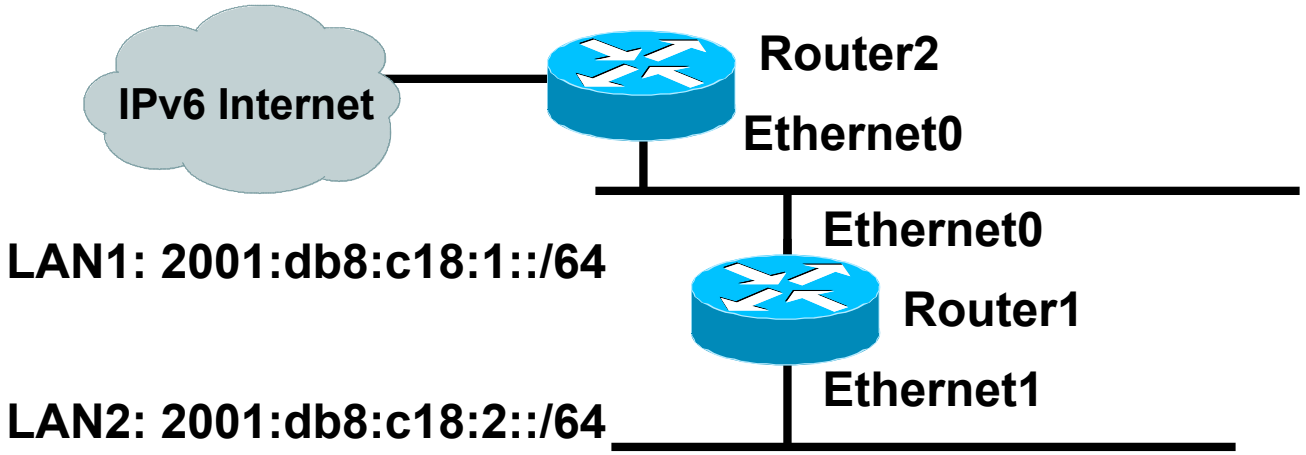
Static Routing – Juniper JunOS

- Static route:

```
[edit routing-options]
   rib inet6.0 {
        static {
             route 2001:db8::/64 {
                  next-hop 2001:db8:0:cc00::1;
                  metric 110;
             }
        }
   }
```

  - Routes packets for network 2001:db8::/64 to a networking device at 2001:db8:0:CC00::1 with an administrative distance of 110

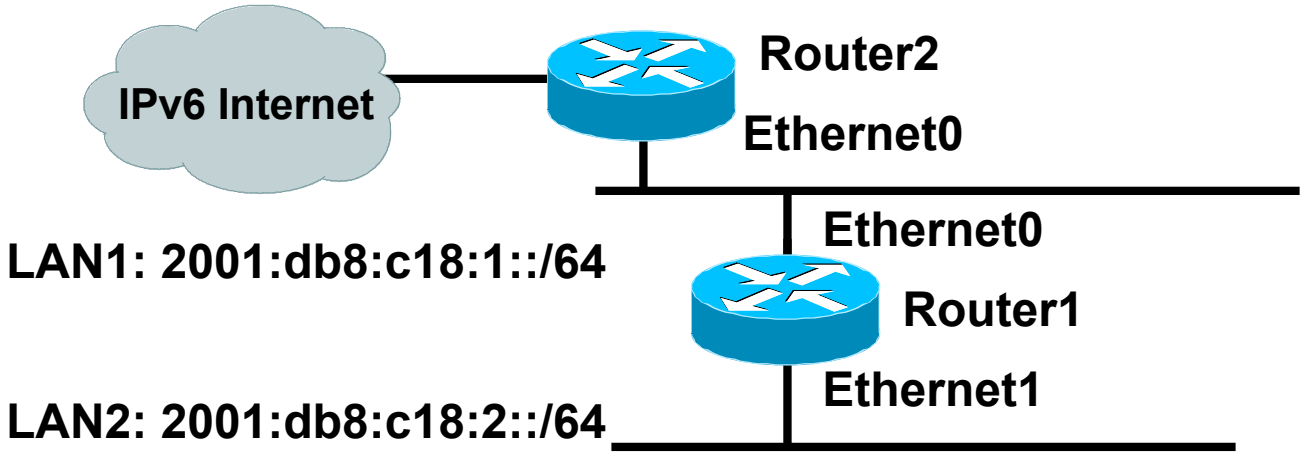
Default Routing Example – IOS




```
ipv6 unicast-routing
!
interface Ethernet0
 ipv6 address 2001:db8:c18:1::a/64
!
interface Ethernet1
 ipv6 address 2001:db8:c18:2::a/64
!
ipv6 route ::/0 <address of R2 ethernet0>
```

Default Route to Router2

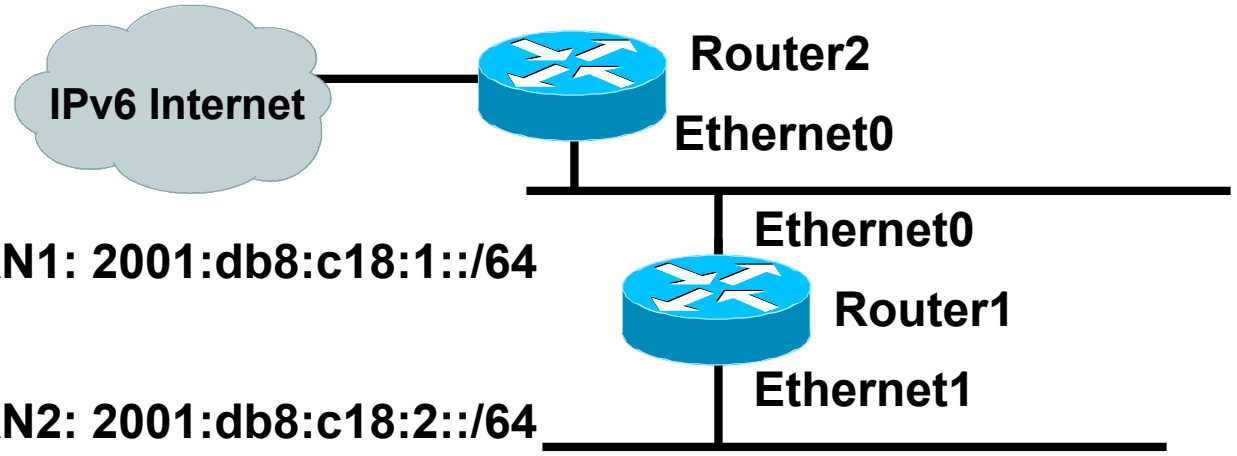
Default Routing Example – IOS-XR




```
interface Ethernet0
 ipv6 address 2001:db8:c18:1::a/64
!
interface Ethernet1
 ipv6 address 2001:db8:c18:2::a/64
!
router static
 address-family ipv6 unicast
  ::/0 <address of R2 ethernet0>
```

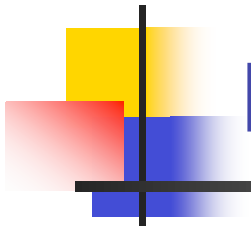
Default Route to Router2

Default Routing Example – JunOS




```
routing-options {
    rib inet6.0 {
        static {
            route ::/0 next-hop <address of R2 Ethernet0>;
        }
    }
}
```

Default Route to Router2



Dynamic Routing Protocols in IPv6

- Dynamic Routing in IPv6 is unchanged from IPv4:
  - IPv6 has 2 types of routing protocols: IGP and EGP
  - IPv6 still uses the longest-prefix match routing algorithm
- IGP
  - RIPng (RFC 2080)
  - Cisco EIGRP for IPv6
    - Juniper does not support EIGRP
  - OSPFv3 (RFC 2740)
  - Integrated IS-ISv6 (draft-ietf-isis-ipv6-06)
- EGP
  - MP-BGP4 (RFC 4760 and RFC 2545)

Configuring Routing Protocols – IOS

- Dynamic routing protocols require router-id
  - Router-id is a 32 bit integer
  - IOS auto-generates these from loopback interface address if configured, else highest IPv4 address on the router
  - **Most ISPs will deploy IPv6 dual stack** – so router-id will be automatically created
- Early adopters choosing to deploy IPv6 in the total absence of any IPv4 addressing need to be aware:
  - Router-id needs to be manually configured:

```
ipv6 router ospf 100
 router-id 10.1.1.4
```

Configuring Routing Protocols – IOS-XR

- For IPv4 routing protocols, IOS-XR auto-generates the router-id as per IOS rules
- But:
  - BGP requires router-id to be manually configured
    - Example:

```
router bgp 2.4
 router-id 10.1.1.4
```

  - OSPFv3 requires router-id to be manually configured:
    - Example:

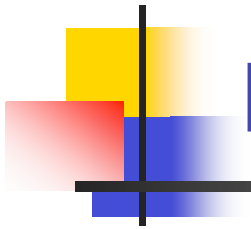
```
router ospfv3 ISP-BB
 router-id 10.1.1.4
```

Configuring Routing Protocols – JunOS

- Dynamic routing protocols require router-id
  - User can (and should!) configure router-id explicitly:

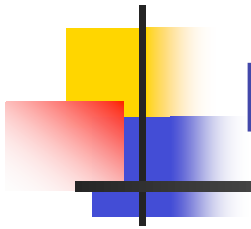
```
routing-options {
    router-id ipv4address;
}
```

  - If user does not explicitly configure router-id, loopback id is used



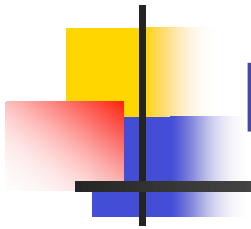
RIPng

- For the ISP industry, simply don’t go here
- ISPs do not use RIP in any form unless there is absolutely no alternative
  - And there usually is
- RIPng was used in the early days of the IPv6 test network
  - Sensible routing protocols such as OSPF and BGP rapidly replaced RIPng when they became available



EIGRP for IPv6

- Cisco EIGRP has had IPv6 protocol support added
  - Just another protocol module (IP, IPX, AppleTalk) with three new TLVs:
    - IPv6_REQUEST_TYPE (0X0401)
    - IPv6_METRIC_TYPE (0X0402)
    - IPv6_EXTERIOR_TYPE (0X0403)
  - Router-ID is still 32-bit, protocol is still 88
- Uses similar CLI to existing IPv4 protocol support
- Easy deployment path for existing IPv4 EIGRP users
- In IOS Release 12.4 onwards
- Not in IOS-XR yet



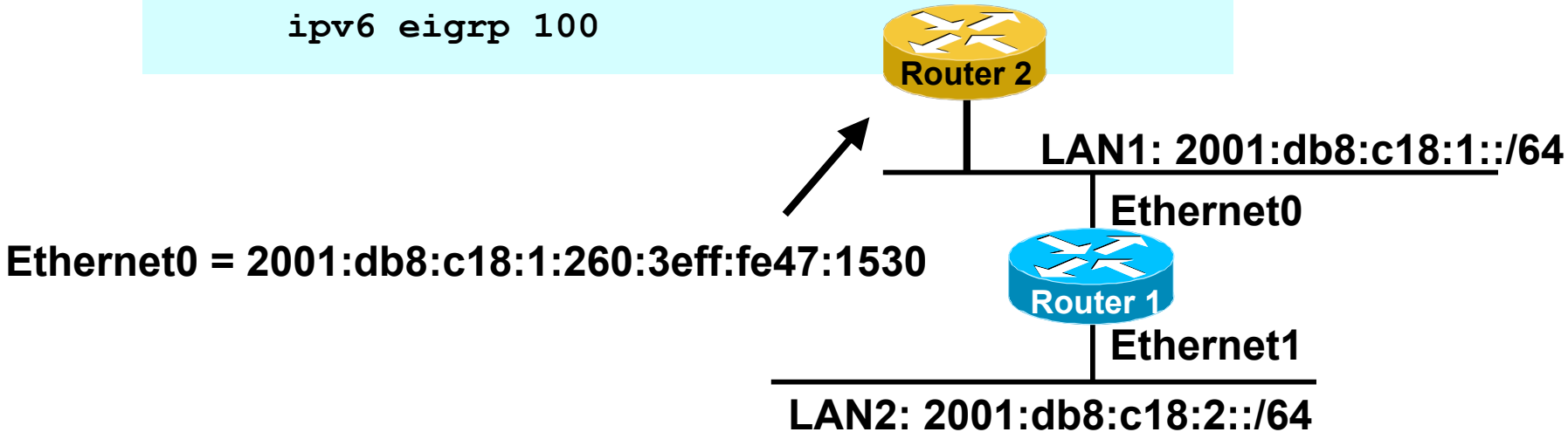
EIGRP for IPv6

- Some differences:
  - Hellos are sourced from the link-local address and destined to FF02::A (all EIGRP routers). This means that neighbors do not have to share the same global prefix (with the exception of explicitly specified neighbours where traffic is unicasted).
  - Automatic summarisation is disabled by default for IPv6 (unlike IPv4)
  - No split-horizon in the case of EIGRP for IPv6 (because IPv6 supports multiple prefixes per interface)

EIGRP for IPv6 – Configuration

- Router 2 configuration:

```
ipv6 router eigrp 100
!
interface Ethernet0
 ipv6 address 2001:db8:c18:1::/64 eui-64
 ipv6 enable
 ipv6 eigrp 100
```



EIGRP for IPv6 – Display

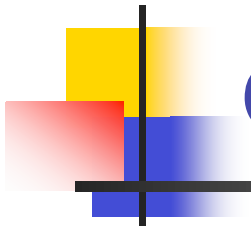
```
Router1#show ipv6 eigrp neighbor
IPv6-EIGRP neighbors for process 100
H Address                 Int Hold Uptime   SRTT   RTO Q   Seq
                            (sec)           (ms)       Cnt Num
0 FE80::260:3eff:fe47:1530 E0    14 00:01:43    1 4500 0   1
```

Neighbour Identified by Link-Local Address

Note Router-ID is 32 bit integer

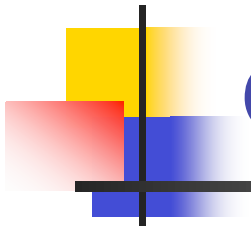
```
Router1#show ipv6 eigrp topology all-links
IPv6-EIGRP Topology Table for AS(100)/ID(10.10.10.1)
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply
       r - reply Status, s - sia Status
P 2001:db8:c18:1::/64, 1 successors, FD is 28160, serno 1
    via Connected, Ethernet0
   via FE80::260:3eff:fe47:1530 (30720/28160), Ethernet0
```

OSPFv3



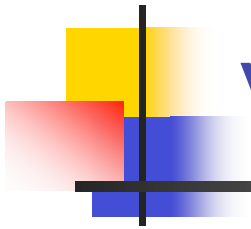
OSPFv3 overview

- OSPF for IPv6
- Based on OSPFv2, with enhancements
- Distributes IPv6 prefixes
- Runs directly over IPv6
- Ships-in-the-night with OSPFv2



OSPFv3 / OSPFv2 Similarities

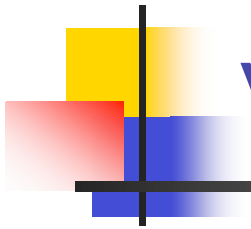
- Basic packet types
  - Hello, DBD, LSR, LSU, LSA
- Mechanisms for neighbor discovery and adjacency formation
- Interface types
  - P2P, P2MP, Broadcast, NBMA, Virtual
- LSA flooding and aging
- Nearly identical LSA types



V2, V3 Differences

OSPFv3 runs on a Link instead of per IP Subnet

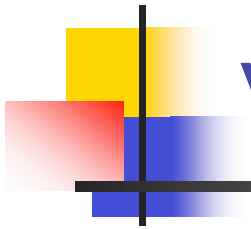
- A link by definition is a medium over which two nodes can communicate at link layer
- In IPv6 multiple IP subnet can be assigned to a link and two nodes in different subnet can communicate at link layer therefore OSPFv3 is running per link instead of per IP subnet
- An Interface connect to a link and multiple interface can be connected to a link



V2, V3 Differences (Cont.)

Support of Multiple Instances per Link

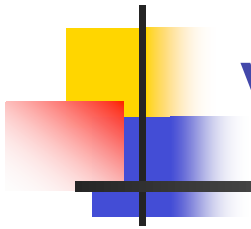
- New field (instance) in OSPF packet header allow running multiple instance per link
- Instance ID should match before packet being accepted
- Useful for traffic separation, multiple areas per link and AF (see later)



V2, V3 Differences (Cont.)

Address Semantic Change in LSA

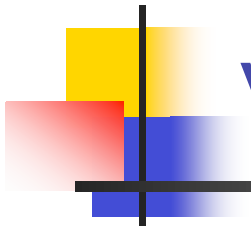
- Router and Network LSA carry only topology information
- Router LSA can be split across multiple LSAs; Link State ID in LSA header is a fragment ID
- Intra area prefix are carried in a new LSA payload called intra-area-prefix-LSAs
- Prefix are carried in payload of inter-area and external LSA



V2, V3 Differences (Cont.)

Generalization of Flooding Scope

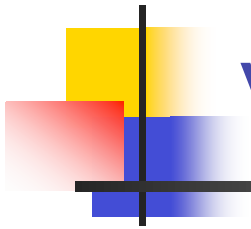
- In OSPFv3 there are three flooding scope for LSAs (link-local scope, area scope, AS scope) and they are coded in LS type explicitly
- In OSPFv2 initially only area and AS wide flooding was defined; later opaque LSAs introduced link local scope as well



V2, V3 Differences (Cont.)

Explicit Handling of Unknown LSA

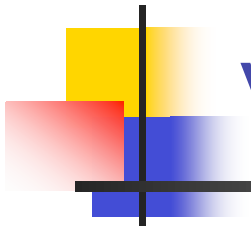
- The handling of unknown LSA is coded via U-bit in LS type
- When U bit is set, the LSA is flooded with the corresponding flooding scope, as if it was understood
- When U bit is clear, the LSA is flooded with link local scope
- In v2 unknown LSA were discarded



V2, V3 Differences (Cont.)

Authentication is Removed from OSPF

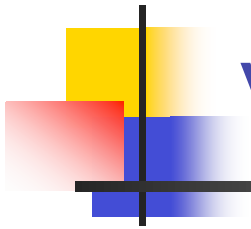
- Authentication in OSPFv3 has been removed
  - OSPFv3 relies on IPv6 authentication header since OSPFv3 runs over IPv6
- Autype and Authentication field in the OSPF packet header have been suppressed



V2, V3 Differences (Cont.)

OSPF Packet format has been changed

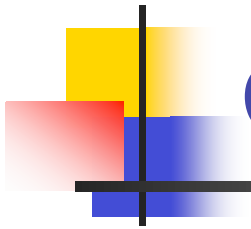
- The mask field has been removed from Hello packet
- IPv6 prefix is only present in payload of Link State update packet



V2, V3 Differences (Cont.)

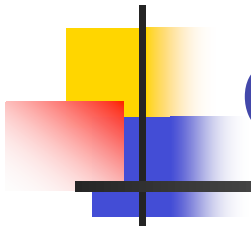
Two New LSAs Have Been Introduced

- Link-LSA has a link local flooding scope and has three purposes:
  - Provides router link-local address
  - Lists all IPv6 prefixes attached to link
  - Assert collection of option bits for Router LSA
- Intra-area-prefix-LSA to advertise router's IPv6 address within the area



Configuring OSPFv3 in Cisco IOS

- Similar to OSPFv2
  - Prefixing existing Interface and Exec mode commands with "**`ipv6`**"
- Interfaces configured directly
  - Replaces **`network`** command
  - (Also available in OSPFv2 from IOS 12.4)
- "Native" IPv6 router mode
  - Not a sub-mode of **`router ospf`**



Configuration modes in OSPFv3

- Entering router mode

```
[no] ipv6 router ospf <process ID>
```

- Entering interface mode

```
[no] ipv6 ospf <process ID> area <area ID>
```

- Exec mode

```
show ipv6 ospf [<process ID>]
clear ipv6 ospf [<process ID>]
```

OSPFv3 Specific Attributes – IOS

- Configuring area range

  `[no] area <area ID> range <prefix>/<prefix length>`

- Showing new LSA

  `show ipv6 ospf [<process ID>] database link`

  `show ipv6 ospf [<process ID>] database prefix`

- Configuring authentication
  - Under **`ipv6 router ospf`**:

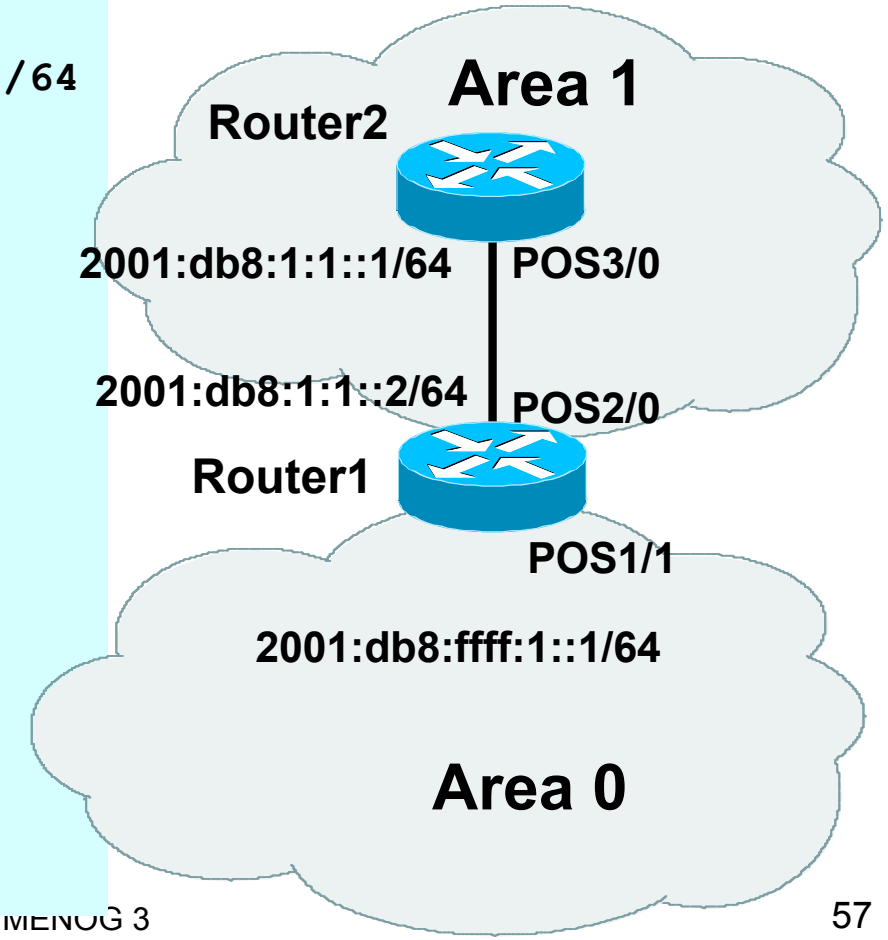
    `area 0 authentication ipsec spi 256 md5 <passwd>`

  - Under interface:

    `ipv6 ospf authentication ipsec spi 256 md5 <passwd>`

OSPFv3 Configuration Example – IOS

```
Router1#
interface POS1/1
 ipv6 address 2001:db8:FFFF:1::1/64
 ipv6 ospf 100 area 0
!
interface POS2/0
 ipv6 address 2001:db8:1:1::2/64
 ipv6 ospf 100 area 1
!
ipv6 router ospf 100


Router2#
interface POS3/0
 ipv6 address 2001:db8:1:1::1/64
 ipv6 ospf 100 area 1
!
ipv6 router ospf 100
```



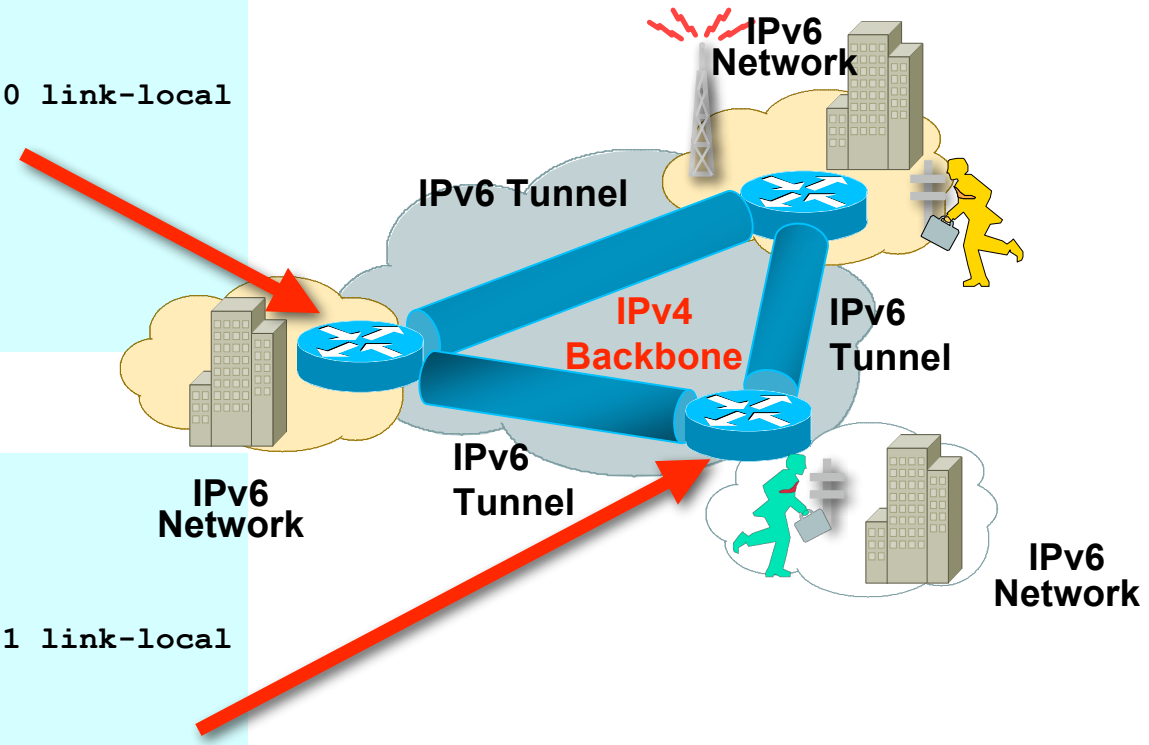
OSPFv3 entries in Routing Table – IOS

```
Router2#sh ipv6 route
IPv6 Routing Table - 5 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
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
OI  2001:db8:FFFF:1::/64 [110/2]
     via FE80::2D0:FFFF:FE60:DFFF, POS3/0
C   2001:db8:1:1::/64 [0/0]
     via ::, POS3/0
L   2001:db8:1:1::1/128 [0/0]
     via ::, POS3/0
L   FE80::/10 [0/0]
     via ::, Null0
L   FF00::/8 [0/0]
     via ::, Null0
```

OSPFv3 on IPv6 Tunnels over IPv4 – IOS

```
On Router1:
interface Tunnel0
 no ip address
 ipv6 address 2001:db8:1::1/64
 ipv6 address FE80::10:7BC2:ACC9:10 link-local
 ipv6 router ospf 1 area 0
 tunnel source 10.42.1.1
 tunnel destination 10.42.2.1
 tunnel mode ipv6ip
!
ipv6 router ospf 1
```

```
On Router2:
interface Tunnel0
 no ip address
 ipv6 address 2001:db8:1::2/64
 ipv6 address FE80::10:7BC2:B280:11 link-local
 ipv6 router ospf 1 area 0
 tunnel source 10.42.2.1
 tunnel destination 10.42.1.1
 tunnel mode ipv6ip
!
ipv6 router ospf 1
```



Configuring OSPFv3 in IOS-XR

- Similar to OSPFv2
  - Routing process is called **`ospfv3`** rather than just **`ospf`**
- Interfaces configured directly, as for OSPFv2
- Entering router mode

```
[no] router ospfv3 <process ID>
```

- Activating interfaces done in router mode:

```
area <number>
  [no] interface <interface name>
```

- Exec mode

```
show ospfv3 [<process ID>]
clear ospfv3 [<process ID>]
```

OSPFv3 Specific Attributes – IOS-XR

- Configuring area range

```
area <number>
  [no] range <prefix>/<prefix length>
```

- Showing new LSA

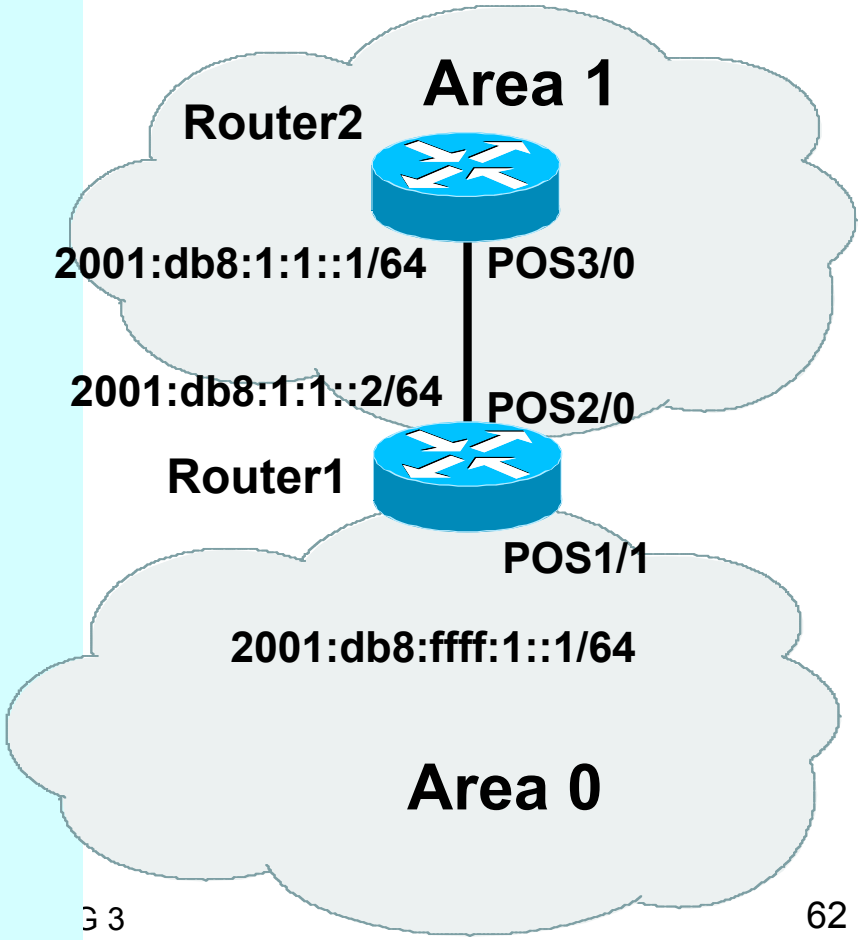
```
show ospfv3 [<process ID>] database link
show ospfv3 [<process ID>] database prefix
```

- Configuring authentication
  - All done under **router ospfv3**:

```
area <number>
 authentication ipsec spi 256 md5 <passwd>
 interface <interface-name> authentication ipsec
  spi 256 md5 <passwd>
```

OSPFv3 Configuration Example – IOS-XR

```
Router1#
interface POS1/1
 ipv6 address 2001:db8:FFFF:1::1/64
!
interface POS2/0
 ipv6 address 2001:db8:1:1::2/64
!
router ospfv3 ISP-BB
 address-family ipv6 unicast
 area 0
  interface POS1/1
 area 1
  interface POS2/0


Router2#
interface POS3/0
 ipv6 address 2001:db8:1:1::1/64
!
router ospfv3 ISP-BB
 address-family ipv6 unicast
 area 1
  interface POS3/0
```



OSPFv3 entries in Routing Table – IOS-XR

```
Router2#sh route ipv6

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
       O - OSPF intra, OI - OSPF inter, OE1 - OSPF ext 1, OE2 - OSPF ext 2
OI  2001:db8:FFFF:1::/64 [110/2]
     via FE80::2D0:FFFF:FE60:DFFF, 00:35:41, POS3/0
C   2001:db8:1:1::/64 is directly connected
     5w5d, POS3/0
L   2001:db8:1:1::1/128 is directly connected
     2w3d, POS3/0
```

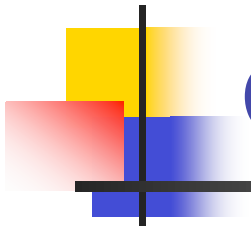
Configuring OSPFv3 on JunOS

- Configuration Mode

```
protocols {
    ospf3 {
        area 0.0.0.0 {
            interface fe-3/0/0.0;
        }
    }
}
```

- Command Mode

```
show ospf3 [ database interface io-statistics
  log neighbor overview route statistics ]
clear ospf3 [database io-statistics neighbor
  statistics ]
```

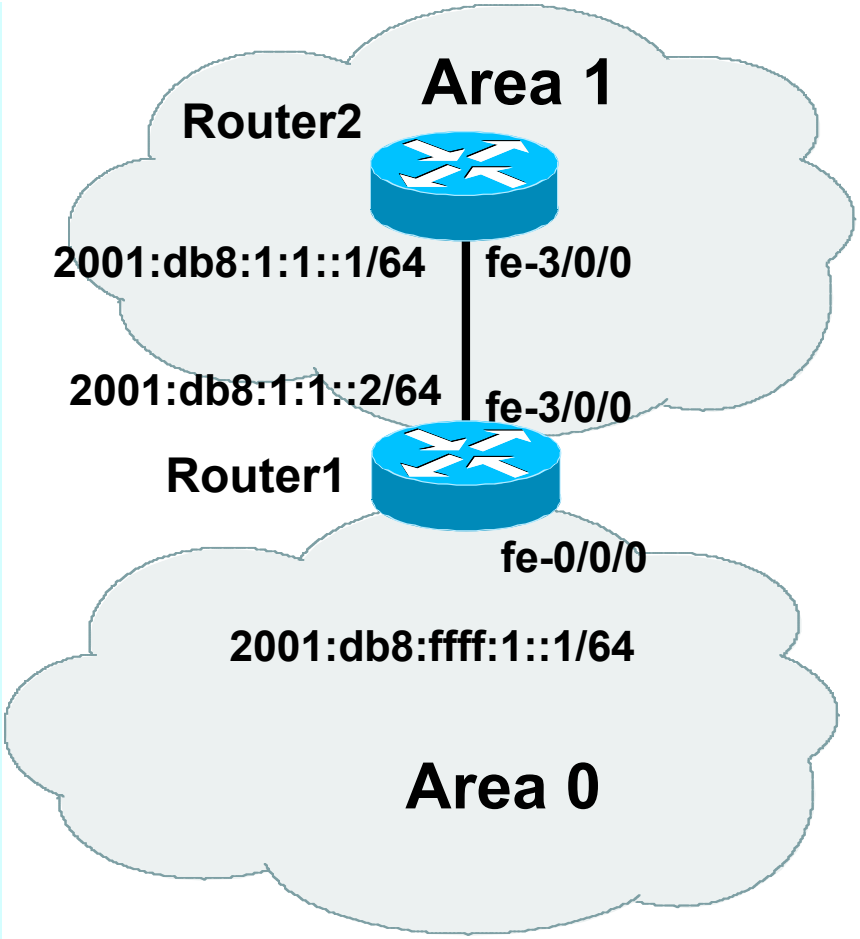


Configuring OSPFv3 on JunOS

- Differences from IOS:
  - No support for authentication (not specified in RFC 2740)
  - Replaced by IPv6 Authentication Header

OSPFv3 Configuration Example – JunOS

```
On Router2:
interfaces {
  fe-3/0/0 {
    unit 0 {
      family inet6 {
        address 2001:db8:1:1::1/64;
      }
    }
  }
}
routing-options {
  router-id 10.1.1.104;
}
protocols {
  ospf3 {
    area 0.0.0.1 {
      interface fe-3/0/0.0 {
        metric 100;
      }
    }
  }
}
```



OSPFv3 Configuration Example – JunOS

```
On Router1:
interfaces {
  fe-0/0/0 {
    unit 0 {
      family inet6 {
        address 2001:db8:ffff:1::1/64;
      }
    }
  }
  fe-3/0/0 {
    unit 0 {
      family inet6 {
        address 2001:db8:1:1::2/64;
      }
    }
  }
}

(Continued -->)
```

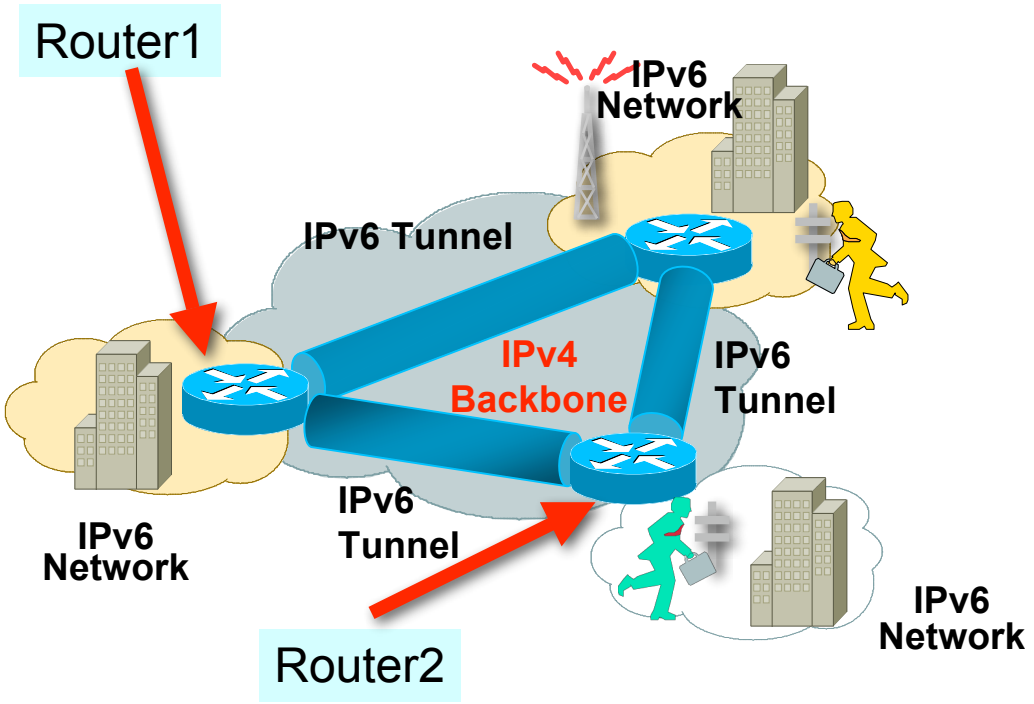
```
routing-options {
  router-id 10.1.1.103;
}
protocols {
  ospf3 {
    area 0.0.0.1 {
      interface fe-3/0/0.0 {
        metric 100;
      }
    }
    area 0.0.0.0 {
      interface fe-0/0/0.0 {
        metric 100;
      }
    }
  }
}
```

OSPFv3 entries in Routing Table – JunOS

```
regress@UI-J6300-2> show route
inet6.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8:1:1::/64  *[Direct/0] 00:56:36
                    > via fe-3/0/0.0
2001:db8:1:1::1/128*[Local/0] 00:56:36
                      Local via fe-3/0/0.0
2001:db8:ffff:1::/64
                   *[OSPF/10] 00:03:56, metric 200
                    > to fe80::205:85ff:fec7:a13c via fe-3/0/0.0
fe80::/64          *[Direct/0] 00:56:36
                    > via fe-3/0/0.0
fe80::205:85ff:fec7:683c/128
                   *[Local/0] 00:56:36
                      Local via fe-3/0/0.0
ff02::5/128        *[OSPF/10] 00:56:36, metric 1
                      MultiRecv
```

OSPFv3 on IPv6 Tunnels over IPv4 – JunOS

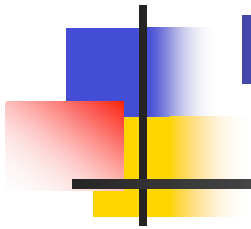
- Requires Tunnel Services PIC or Application Services PIC on both tunnel endpoints
- Configurations on next slide



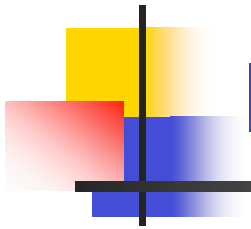
OSPFv3 on IPv6 Tunnels over IPv4 – JunOS

```
On Router1:
[edit]
interfaces {
  gr-1/0/0 {
    unit 0 {
      tunnel {
        source 10.42.1.1;
        destination 10.42.2.1;
      }
      family inet6 {
        address 2001:DB8:1::1/64;
      }
    }
  }
}
protocol {
  ospf3 {
    area 0.0.0.0 {
      interface gr-1/0/0;
    }
  }
}
```

```
On Router 2:
[edit]
interfaces {
  gr-1/0/0 {
    unit 0 {
      tunnel {
        source 10.42.2.1;
        destination 10.42.1.1;
      }
      family inet6 {
        address 2001:DB8:1::2/64;
      }
    }
  }
}
protocol {
  ospf3 {
    area 0.0.0.0 {
      interface gr-1/0/0;
    }
  }
}
```



ISIS for IPv6

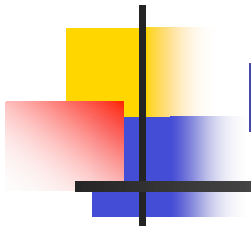


IS-IS Standards History

- ISO 10589 specifies OSI IS-IS routing protocol for CLNS traffic
  - Tag/Length/Value (TLV) options to enhance the protocol
  - A Link State protocol with a 2 level hierarchical architecture.
- RFC 1195 added IP support
  - I/IS-IS runs on top of the Data Link Layer
  - Requires CLNP to be configured
- Internet Draft defines how to add IPv6 address family support to IS-IS

  www.ietf.org/internet-drafts/draft-ietf-isis-ipv6-06.txt
- Internet Draft introduces Multi-Topology concept for IS-IS

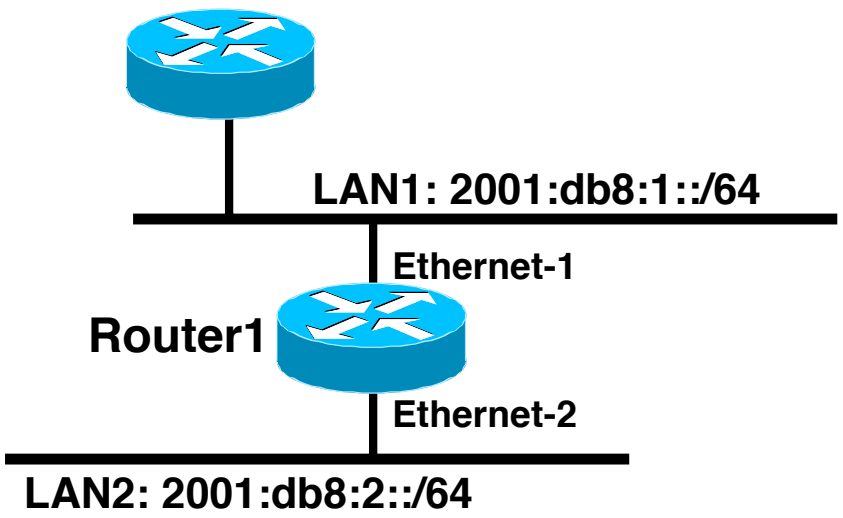
  www.ietf.org/internet-drafts/draft-ietf-isis-wg-multi-topology-11.txt



IS-IS for IPv6

- 2 Tag/Length/Values added to introduce IPv6 routing
- IPv6 Reachability TLV (0xEC)
  - External bit
  - Equivalent to IP Internal/External Reachability TLV's
- IPv6 Interface Address TLV (0xE8)
  - For Hello PDUs, must contain the Link-Local address
  - For LSP, must only contain the non-Link Local address
- IPv6 NLPID (0x8E) is advertised by IPv6 enabled routers

IOS IS-IS dual IP configuration



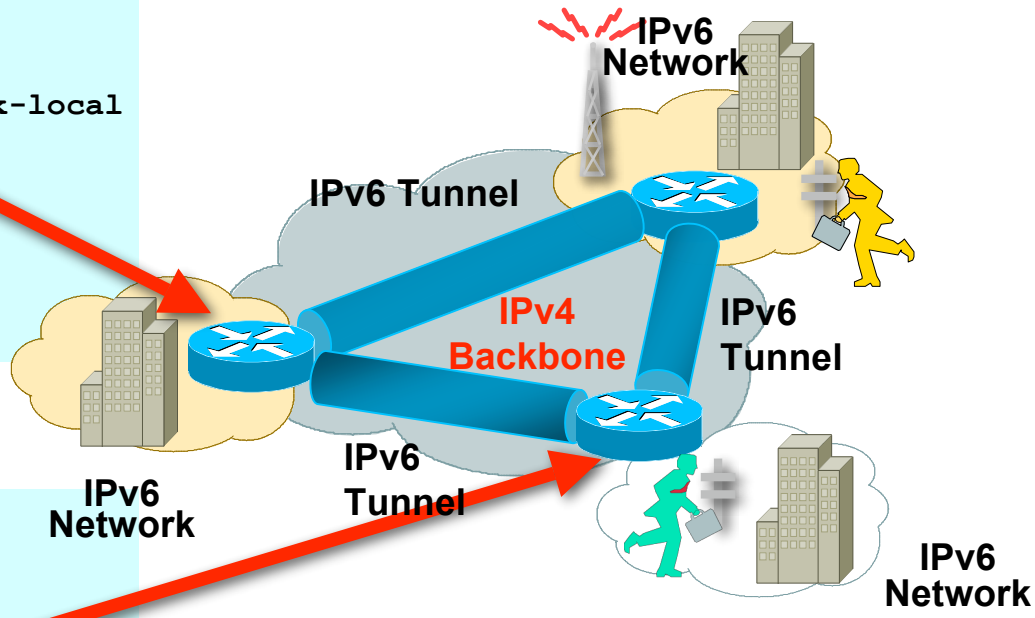
Dual IPv4/IPv6 configuration.
Redistributing both IPv6 static routes and IPv4 static routes.

```
Router1#
interface ethernet-1
 ip address 10.1.1.1 255.255.255.0
 ipv6 address 2001:db8:1::1/64
 ip router isis
 ipv6 router isis

interface ethernet-2
 ip address 10.2.1.1 255.255.255.0
 ipv6 address 2001:db8:2::1/64
 ip router isis
 ipv6 router isis

router isis
 address-family ipv6
  redistribute static
  exit-address-family
 net 42.0001.0000.0000.072c.00
 redistribute static
```

IOS Configuration for IS-IS for IPv6 on IPv6 Tunnels over IPv4

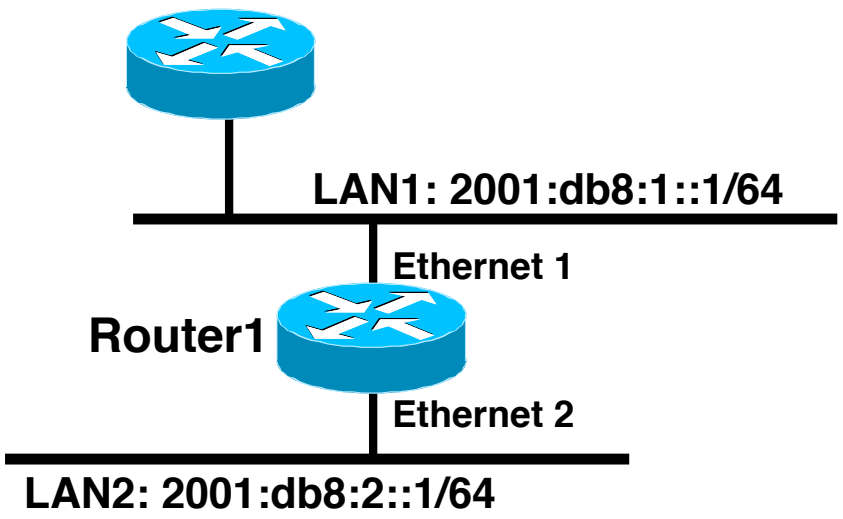
```
On Router1:
interface Tunnel0
 no ip address
 ipv6 address 2001:db8:1::1/64
 ipv6 address FE80::10:7BC2:ACC9:10 link-local
 ipv6 router isis
 tunnel source 10.42.1.1
 tunnel destination 10.42.2.1
!
router isis
 net 42.0001.0000.0000.0001.00
```

```
On Router2:
interface Tunnel0
 no ip address
 ipv6 address 2001:db8:1::2/64
 ipv6 address FE80::10:7BC2:B280:11 link-local
 ipv6 router isis
 tunnel source 10.42.2.1
 tunnel destination 10.42.1.1
!
router isis
 net 42.0001.0000.0000.0002.00
```



IS-IS for IPv6 on an IPv6 Tunnel requires GRE Tunnel; it can't work with IPv6 configured tunnel as IS-IS runs directly over the data link layer

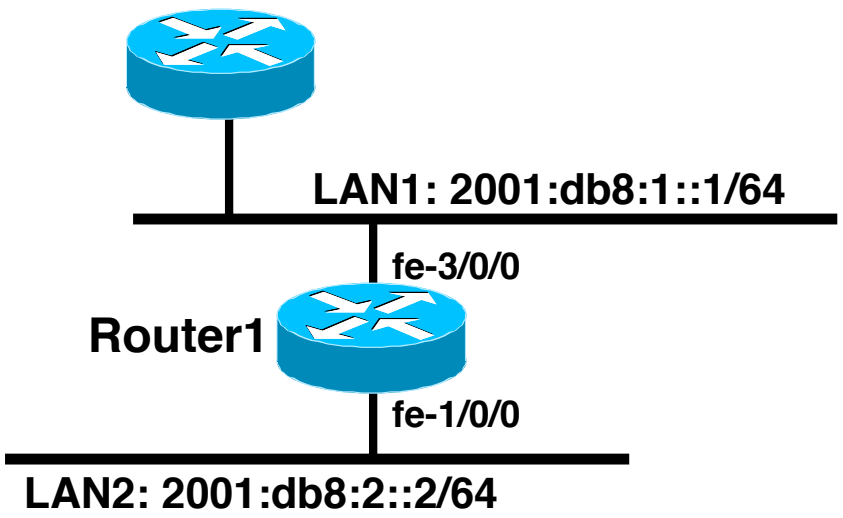
IOS-XR IS-IS dual IP configuration



Dual IPv4/IPv6 configuration - note **`single-topology`** command

```
Router1#
interface Ethernet 1
 ip address 10.1.1.1 255.255.255.0
 ipv6 address 2001:db8:1::1/64
!
interface Ethernet 2
 ip address 10.2.1.1 255.255.255.0
 ipv6 address 2001:db8:2::1/64
!
router isis ISP-BB
 net 42.0001.0000.0000.072c.00
 address-family ipv4 unicast
  redistribute static
 address-family ipv6 unicast
  redistribute static
  single-topology
 interface Ethernet 1
  address-family ipv4 unicast
 interface Ethernet 2
  address family ipv6 unicast
```

JunOS IS-IS dual IP configuration



Dual IPv4/IPv6 configuration.
Redistributing both IPv6 static routes and IPv4 static routes.

```
Router1:

interfaces {
  fe-3/0/0 {
    unit 0 {
      family inet {
        address 10.1.1.1/24;
      }
      family iso;
      family inet6 {
        address 2001:db8:1::1/64;
      }
    }
  }

(Continued -->)
```

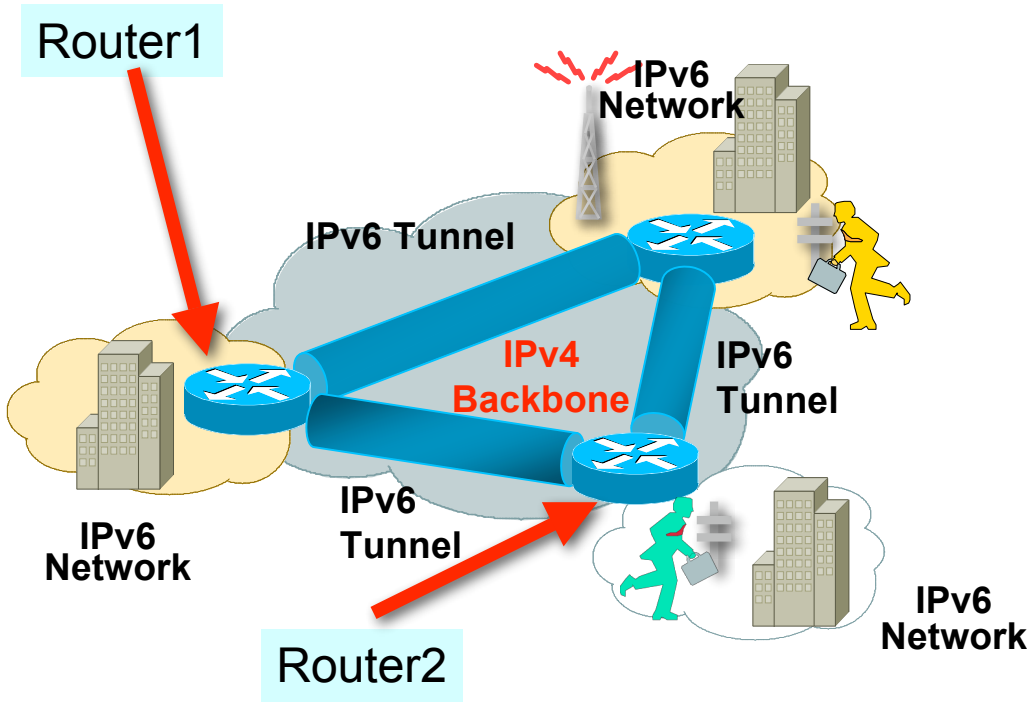
JunOS IS-IS dual IP Configuration

```
fe-1/0/0 {
 unit 0 {
  family inet {
   address 10.2.1.1/24;
  }
  family iso;
  family inet6 {
   address 2001:db8:2::1/64;
  }
 }
}
lo0 {
 unit 0 {
  family inet {
   address 10.1.1.103/32;
  }
  family inet6;

(Continued…)
```

```
   family iso {
    address 42.0001.0000.0000.072c.00;
   }
  }
 }
}
protocols {
 isis {
  export redistribute-static;
  interface fe-1/0/0.0;
  interface fe-3/0/0.0;
  interface lo0.0;
 }
}
policy-options {
 policy-statement redistribute-static
{
  term 1 {
   from protocol static;
   then accept;
  }
 }
}
```

ISIS for IPv6 on IPv6 Tunnels over IPv4 – JunOS

- Requires Tunnel Services PIC or Application Services PIC on both tunnel endpoints
- Configurations on following slides

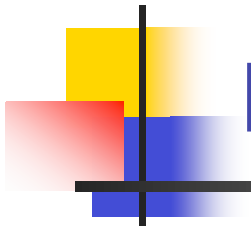


ISIS for IPv6 on IPv6 Tunnels over IPv4 – JunOS

On Router1:

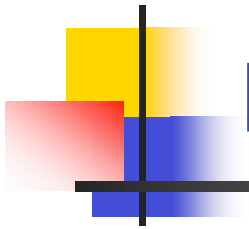
```
interfaces {
 gr-0/0/0 {
  unit 1 {
   tunnel {
    source 10.42.1.1;
    destination 10.42.2.1;
   }
   family inet {
    address 1.1.1.1/30;
   }
   family iso;
   family inet6 {
    address 2001:DB8:1::1/64;
   }
  }
 }

(Continued…)
```

```
 lo0 {
  unit 0 {
   family inet {
    address 10.1.1.103/32;
   }
   family iso {
    address 42.0001.0000.0000.072c.00;
   }
   family inet6;
  }
 }
}
protocols {
 isis {
  interface gr-0/0/0.1;
  interface fe-3/0/0.0 {
   no-ipv6-unicast;
  }
  interface lo0.0;
 }
}
```



Multi-Topology IS-IS extensions

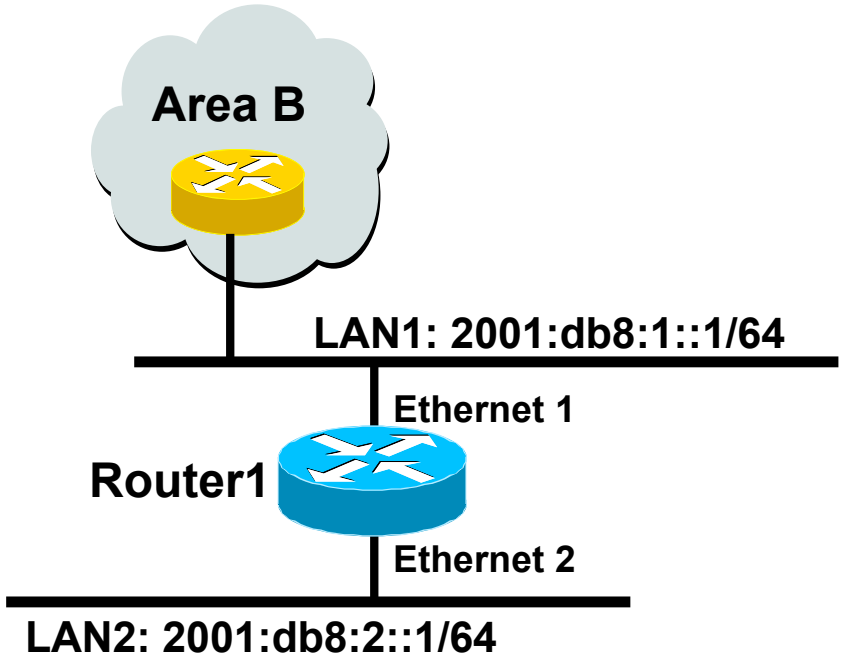
- IS-IS for IPv6 assumes that the IPv6 topology is the same as the IPv4 topology
  - Single SPF running, multiple address families
  - Some networks may be like this, but many others are not
- Multi-Topology IS-IS solves this problem
  - New TLV attributes introduced
  - New Multi-Topology ID #2 for IPv6 Routing Topology
  - Two topologies now maintained:
    - ISO/IPv4 Routing Topology (MT ID #0)
    - IPv6 Routing Topology (MT ID #2)



Multi-Topology IS-IS extensions

- New TLVs attributes for Multi-Topology extensions:
  - Multi-topology TLV: contains one or more multi-topology ID in which the router participates
  - MT Intermediate Systems TLV: this TLV appears as many times as the number of topologies a node supports
  - Multi-Topology Reachable IPv4 Prefixes TLV: this TLV appears as many times as the number of IPv4 announced by an IS for a given MT ID
  - Multi-Topology Reachable IPv6 Prefixes TLV: this TLV appears as many times as the number of IPv6 announced by an IS for a given MT ID

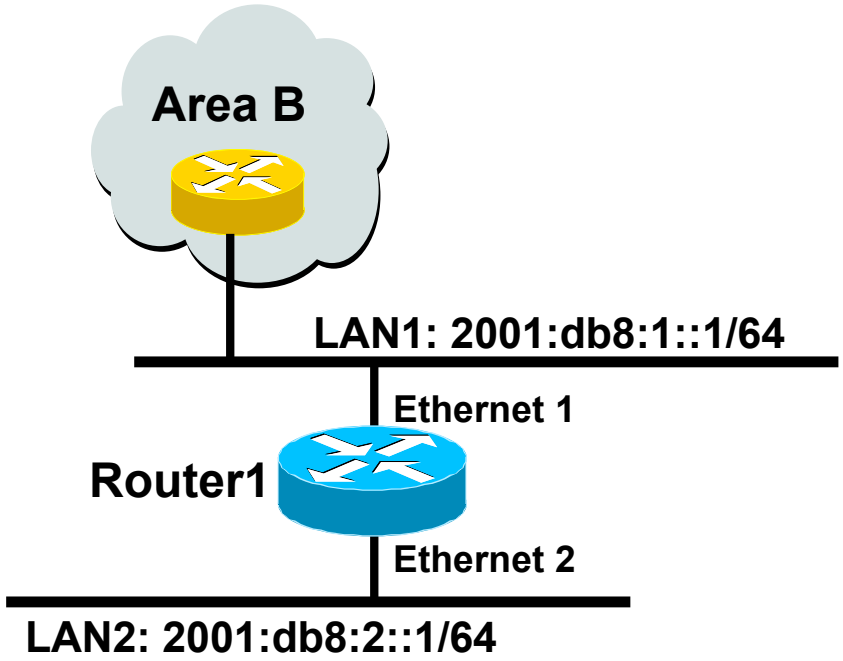
Multi-Topology ISIS configuration example (IOS)



- The optional keyword **`transition`** may be used for transitioning existing IS-IS IPv6 single SPF mode to MT IS-IS
- Wide metric is mandated for Multi-Topology to work

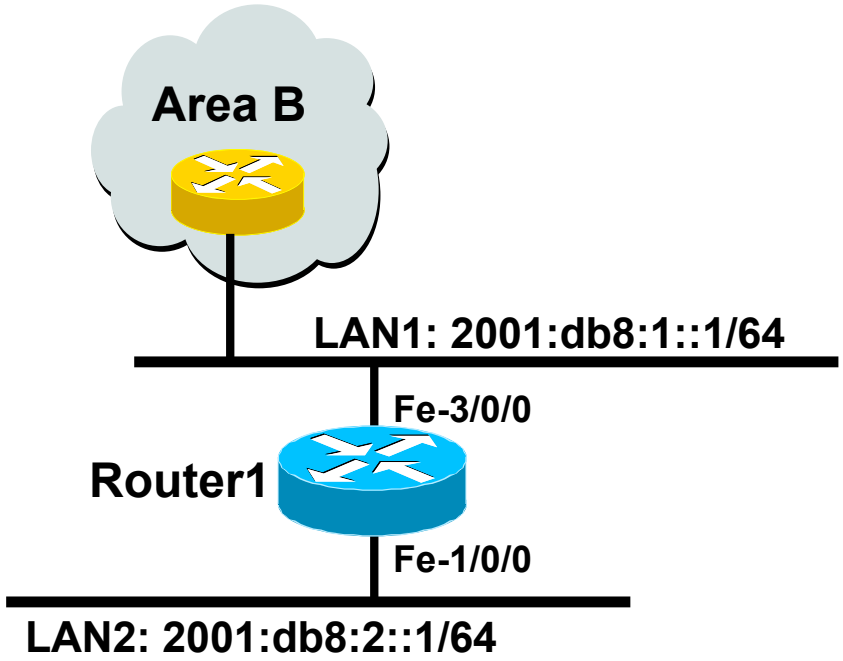
```
Router1#
interface Ethernet 1
 ip address 10.1.1.1 255.255.255.0
 ipv6 address 2001:db8:1::1/64
 ip router isis
 ipv6 router isis
 isis ipv6 metric 20

interface Ethernet 2
 ip address 10.2.1.1 255.255.255.0
 ipv6 address 2001:db8:2::1/64
 ip router isis
 ipv6 router isis
 isis ipv6 metric 20

router isis
 net 42.0001.0000.0000.072c.00
 metric-style wide
 !
 address-family ipv6
  multi-topology
 exit-address-family
```

Multi-Topology ISIS configuration example (IOS-XR)

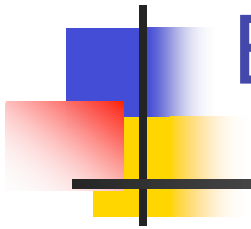



```
Router1#
interface Ethernet 1
 ip address 10.1.1.1 255.255.255.0
 ipv6 address 2001:db8:1::1/64

interface Ethernet 2
 ip address 10.2.1.1 255.255.255.0
 ipv6 address 2001:db8:2::1/64

router isis ISP-BB
 net 42.0001.0000.0000.072c.00
 address-family ipv4 unicast
  metric-style wide
  redistribute static
 !
 address-family ipv6 unicast
  metric-style wide
  redistribute static
 !
 interface Ethernet 1
  address-family ipv4 unicast
 !
 interface Ethernet 2
  address family ipv6 unicast
```

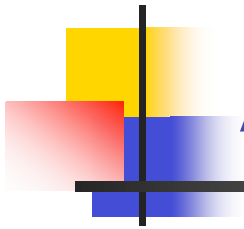
Multi-Topology ISIS configuration example (JunOS)




```
routing-instances {
  test {
    instance-type virtual-router;
    interface fe-3/0/0.0;
    protocols {
      isis {
        interface fe-3/0/0.0;
      }
    }
  }
}
```

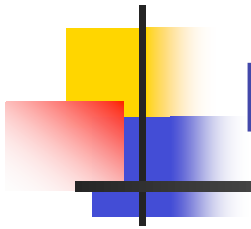


BGP for IPv6



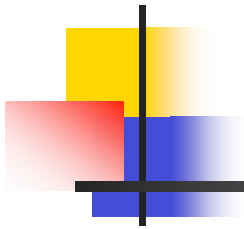
Adding IPv6 to BGP…

- RFC4760
  - Defines Multi-protocol Extensions for BGP4
  - Enables BGP to carry routing information of protocols other than IPv4
    - e.g. MPLS, IPv6, Multicast etc
  - Exchange of multiprotocol NLRI must be negotiated at session startup
- RFC2545
  - Use of BGP Multiprotocol Extensions for IPv6 Inter-Domain Routing



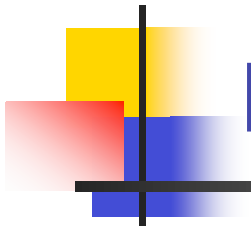
RFC4760

- New optional and non-transitive BGP attributes:
  - MP_REACH_ NLRI (Attribute code: 14)
    - Carry the set of reachable destinations together with the next-hop information to be used for forwarding to these destinations (RFC4760)
  - MP_UNREACH_NLRI (Attribute code: 15)
    - Carry the set of unreachable destinations
- Attribute contains one or more Triples:
  - AFI Address Family Information
  - Next-Hop Information (must be of the same address family)
  - NLRI Network Layer Reachability Information



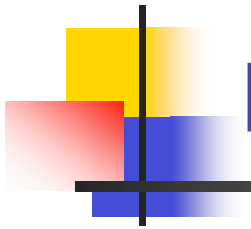
RFC2545

- IPv6 specific extensions
  - Scoped addresses: Next-hop contains a global IPv6 address and/or potentially a link-local address
  - NEXT_HOP and NLRI are expressed as IPv6 addresses and prefix
  - Address Family Information (AFI) = 2 (IPv6)
    - Sub-AFI = 1 (NLRI is used for unicast)
    - Sub-AFI = 2 (NLRI is used for multicast RPF check)
    - Sub-AFI = 3 (NLRI is used for both unicast and multicast RPF check)
    - Sub-AFI = 4 (label)



BGP Considerations

- Rules for constructing the NEXTHOP attribute:
  - When two peers share a common subnet, the NEXTHOP information is formed by a global address and a link local address
  - Redirects in IPv6 are restricted to the usage of link local addresses

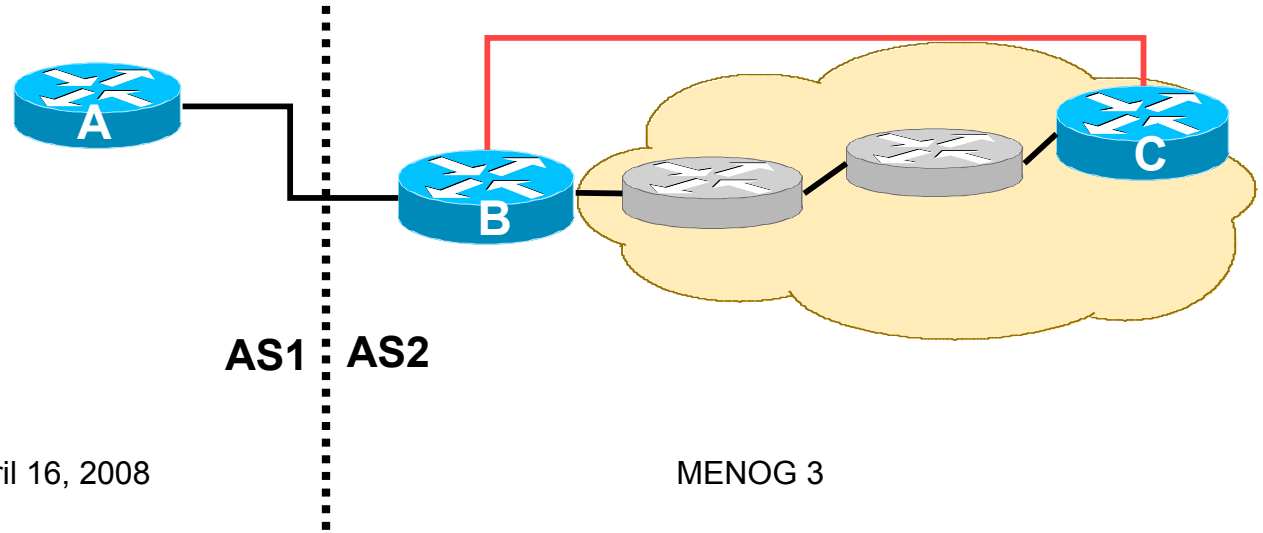


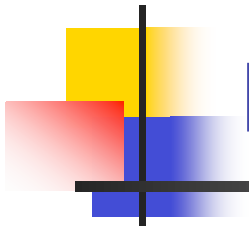
Routing Information

- Independent operation
  - One RIB per protocol
    - e.g. IPv6 has its own BGP table
  - Distinct policies per protocol
- Peering sessions **can** be shared when the topology is congruent

BGP next-hop attribute

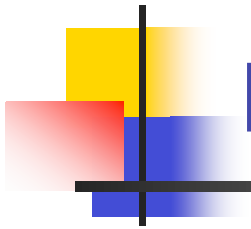
- Next-hop contains a global IPv6 address (or potentially a link local address)
- Link local address as a next-hop is only set if the BGP peer shares the subnet with both routers (advertising and advertised)





More BGP considerations

- TCP Interaction
  - BGP runs on top of TCP
  - This connection could be set up either over IPv4 or IPv6
- Router ID
  - When no IPv4 is configured, an explicit bgp router-id needs to be configured
    - BGP identifier is a 32 bit integer currently generated from the router identifier – which is generated from an IPv4 address on the router
  - This is needed as a BGP identifier, this is used as a tie breaker, and is sent within the OPEN message



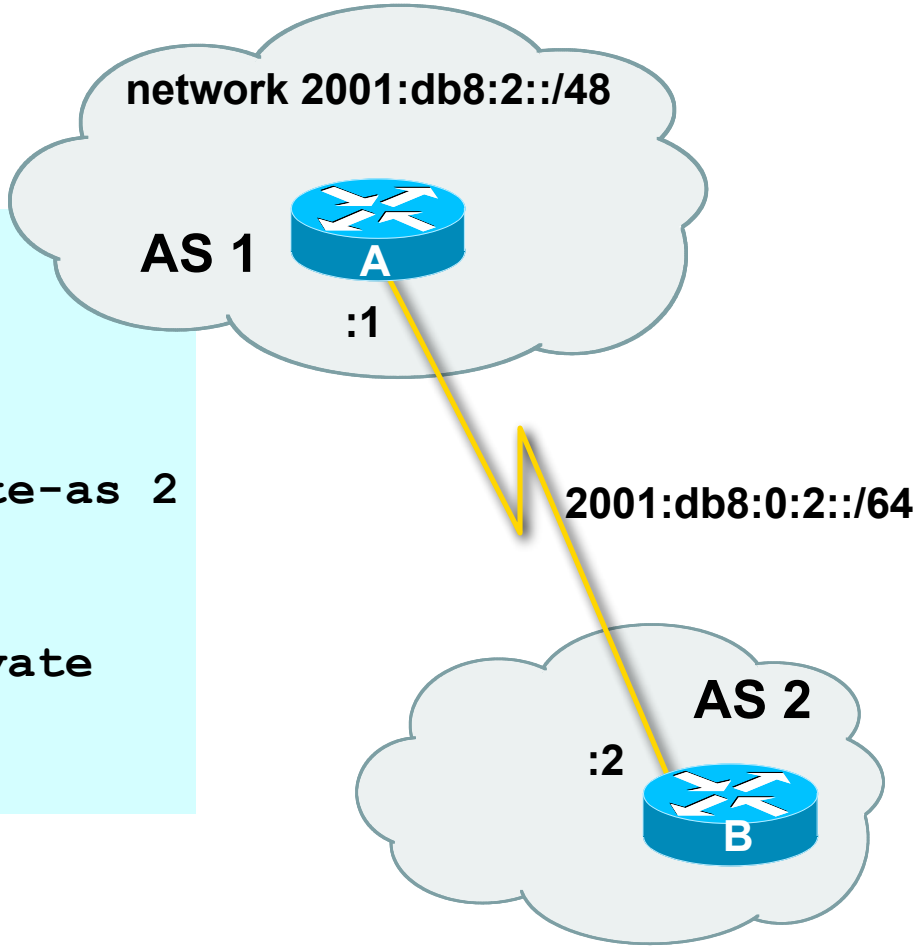
BGP Configuration

- Two options for configuring BGP peering
- Using link local addressing
  - ISP uses FE80:: addressing for BGP neighbours
  - **NOT RECOMMENDED**
    - There are plenty of IPv6 addresses
    - Unnecessary configuration complexity
- Using global unicast addresses
  - As with IPv4
  - **RECOMMENDED**

Regular BGP Peering – IOS

Router A

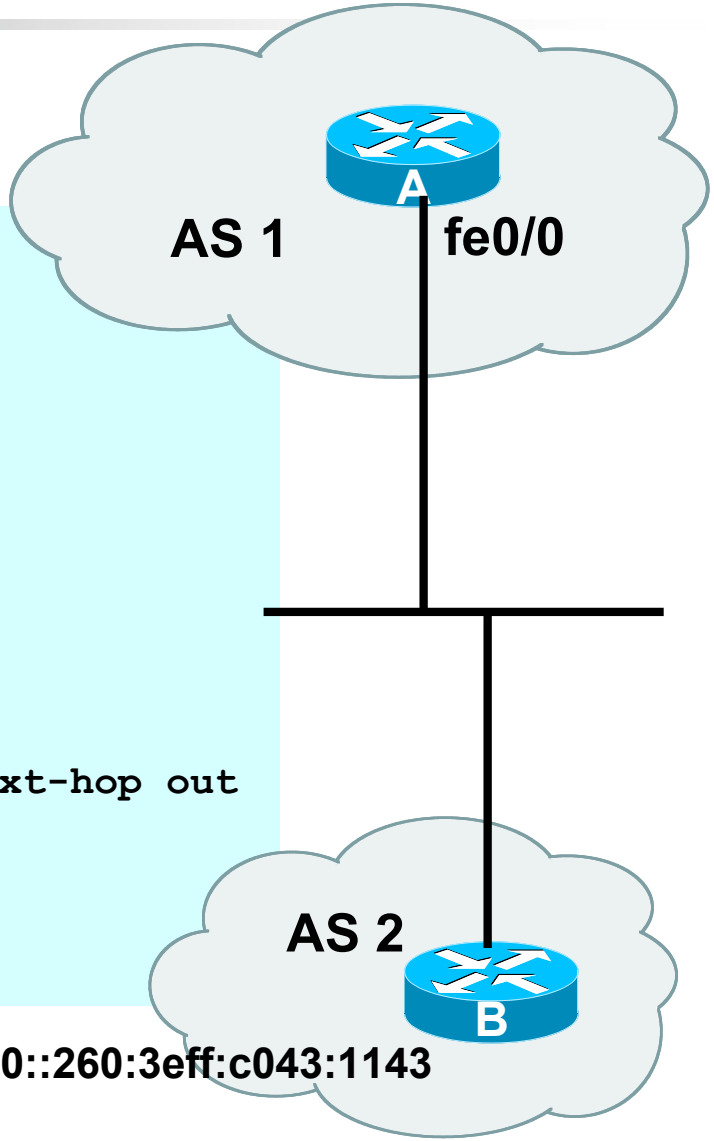
```
router bgp 1
 no bgp default ipv4 unicast
 neighbor 2001:db8:0:2::2 remote-as 2
!
 address-family ipv6
 neighbor 2001:db8:0:2::2 activate
 network 2001:db8:2::/48
!
```



Link Local Peering – IOS

Router A

```
interface fastethernet 0/0
 ipv6 address 2001:db8:0:1::1/64
!
router bgp 1
 no bgp default ipv4 unicast
 neighbor fe80::260:3eff:c043:1143 remote-as 2
!
address-family ipv6
 neighbor fe80::260:3eff:c043:1143 activate
 neighbor fe80::260:3eff:c043:1143 route-map next-hop out
!
route-map next-hop permit 5
 set ipv6 next-hop 2001:db8:0:1::1
!
```

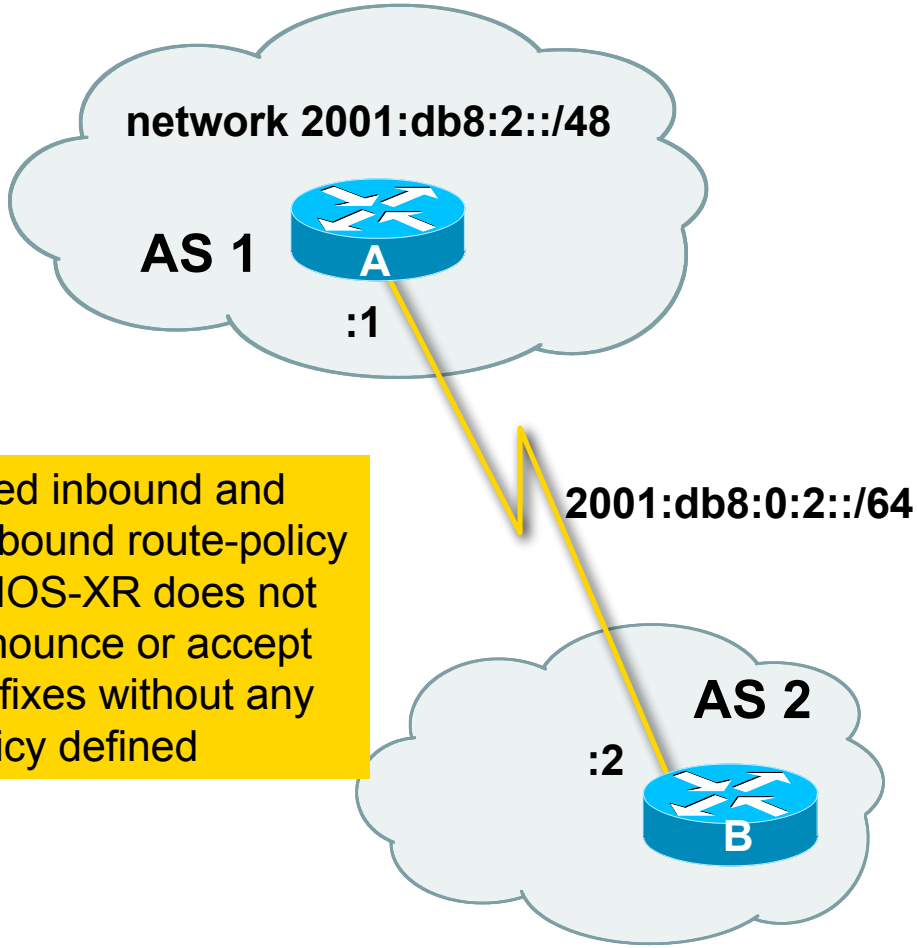


Regular BGP Peering – IOS-XR

Router A

```
router bgp 1
 bgp router-id 10.1.1.4
 !
 address-family ipv6 unicast
 network 2001:db8:2::/48
 !
 neighbor 2001:db8:0:2::2
  remote-as 2
  route-policy all-v6-in in
  route-policy my-v6-out out
!
! all-v6-in <snipped>
route-policy my-v6-out
 if destination in my-v6 then pass
 endif
end-policy
!
prefix-set my-v6
 2001:db8:2::/48
end-set
!
```

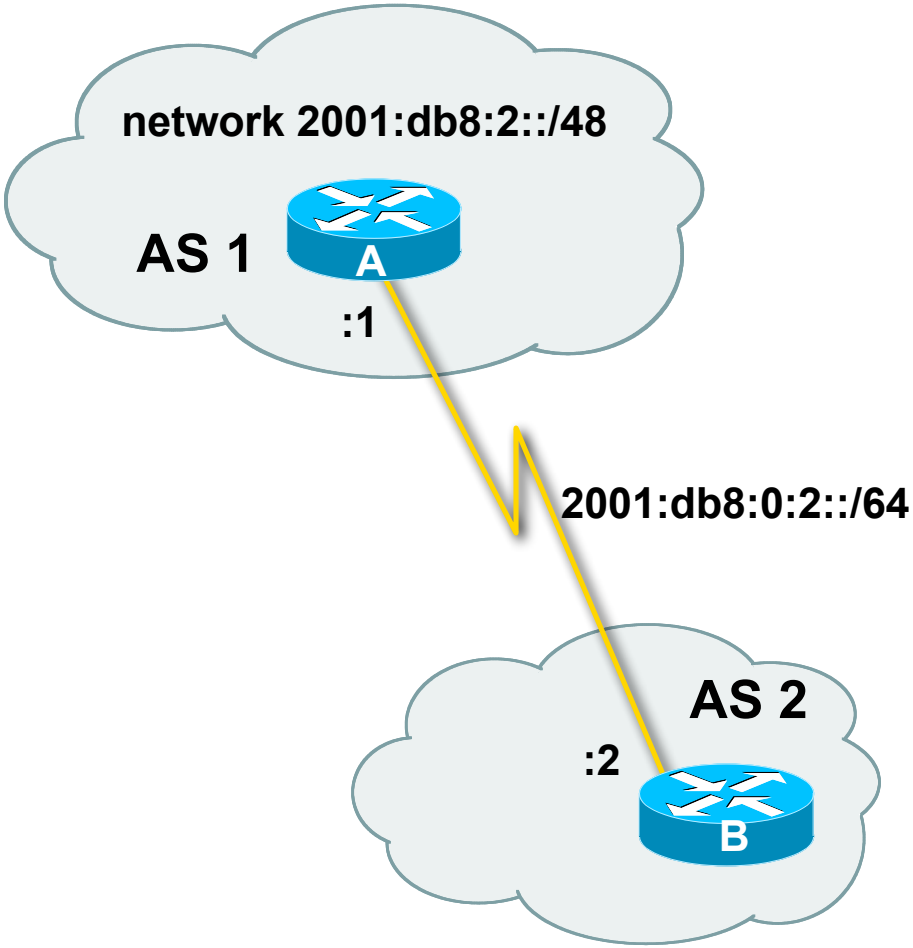
Need inbound and outbound route-policy as IOS-XR does not announce or accept prefixes without any policy defined



Regular BGP Peering – JunOS

Router A

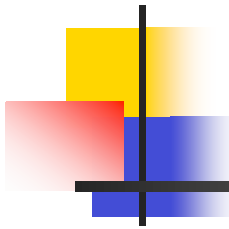
```
interfaces {
  fe-3/0/0 {
    unit 0 {
      family inet6 {
        address 2001:db8:0:2::1/64;
      }
    }
  }
}
routing-options {
  rib inet6.0 {
    static {
      route 2001:db8:2::/48 discard;
    }
  }
  router-id 10.1.1.103;
}

(Continued -->)
```



Regular BGP Peering – Juniper JunOS

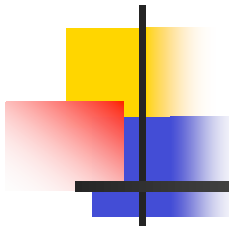
Router A

```
protocols {
    bgp {
        local-as 1;
        group as2 {
            export export-static;
            peer-as 2;
            neighbor 2001:db8:0:2::2;
        }
    }
}
policy-options {
    policy-statement export-static {
        term 1 {
            from protocol static;
            then accept;
        }
    }
}
```



IPv4 and IPv6 – IOS

```
router bgp 10
 no bgp default ipv4-unicast
 neighbor 2001:db8:1:1019::1 remote-as 20
 neighbor 172.16.1.2 remote-as 30
!
 address-family ipv4
 neighbor 172.16.1.2 activate
 neighbor 172.16.1.2 prefix-list ipv4-ebgp in
 neighbor 172.16.1.2 prefix-list v4out out
 network 172.16.0.0
 exit-address-family
!
 address-family ipv6
 neighbor 2001:db8:1:1019::1 activate
 neighbor 2001:db8:1:1019::1 prefix-list ipv6-ebgp in
 neighbor 2001:db8:1:1019::1 prefix-list v6out out
 network 2001:db8::/32
 exit-address-family
!
! Continued -->
```



IPv4 and IPv6 – IOS

```
ip prefix-list ipv4-ebgp permit 0.0.0.0/0 le 32
!
ip prefix-list v4out permit 172.16.0.0/16
!
ipv6 prefix-list ipv6-ebgp permit ::/0 le 128
!
ipv6 prefix-list v6out permit 2001:db8::/32
!
```

- Compare IPv4 prefix filters with IPv6 prefix filters
  - **ip prefix-list <name> permit|deny <ipv4 address>**
  - **ipv6 prefix-list <name> permit|deny <ipv6 address>**

IPv4 and IPv6 – IOS-XR

```
router bgp 10
 bgp router-id 10.1.1.4
 !
 address-family ipv4 unicast
  network 172.16.0.0
 !
 address-family ipv6 unicast
  network 2001:db8::/32
 !
 neighbor 2001:db8:1:1019::1
  remote-as 20
  address-family ipv6 unicast
   route-policy ipv6-ebgp in
   route-policy v6out out
 !
 neighbor 172.16.1.2
  remote-as 30
  address-family ipv4 unicast
   route-policy ipv4-ebgp in
   route-policy v4out out

! Continued -->
```

```
route-policy ipv6-ebgp
 if destination in full-v6 then
   pass
 endif
end-policy
!
prefix-set full-v6
 ::/0 le 128
!
route-policy v6out
 if destination in v6out then
   pass
 endif
end-policy
!
prefix-set v6out
 2001:db8::/32
end-set



! Continued -->
```

IPv4 and IPv6 – IOS-XR

```
route-policy ipv4-ebgp
 if destination in full-v4 then
   pass
 endif
end-policy
!
prefix-set full-v4
 0.0.0.0/0 le 32
end-set
!
route-policy v4out
 if destination in v4out then
   pass
 endif
end-policy
!
prefix-set v4out
 172.16.0.0/16
end-set
```

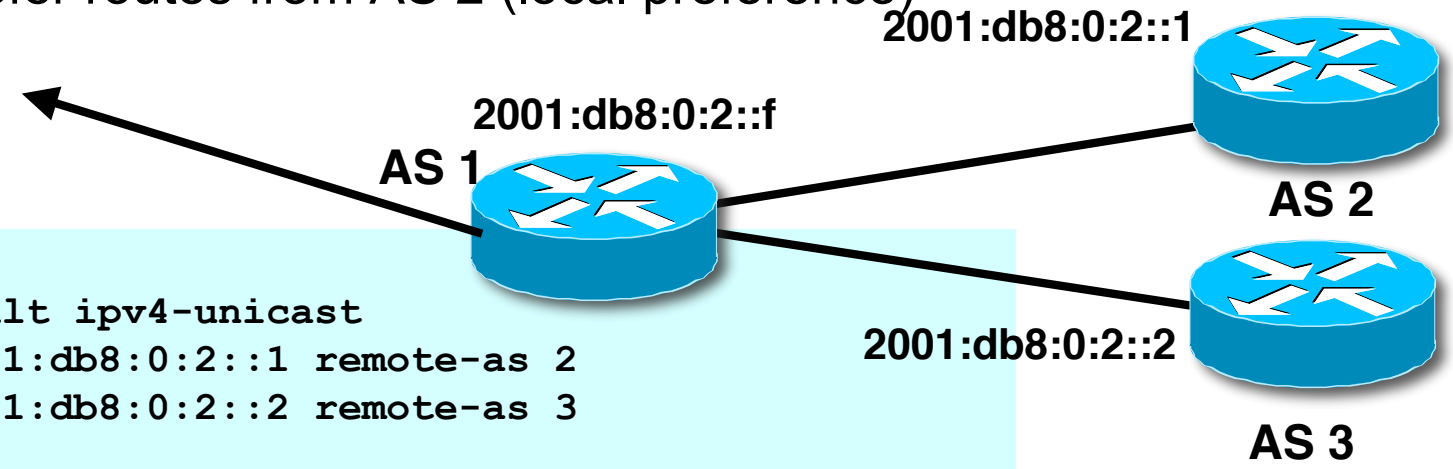
- Note the per address family configuration per neighbour
- Prefix-sets are similar to IOS prefix-lists, but no distinction between IPv4 and IPv6

IPv4 and IPv6 – JunOS

```
interfaces {
  fe-3/0/0 {
    unit 0 {
      family inet {
        address 10.1.1.1/24;
      }
      family inet6 {
       address 2001:db8:1::45c/64;
      }
    }
  }
}
routing-options {
  rib inet6.0 {
    static {
      route 2001:db8::/32 discard;
    }
  }
  router-id 10.1.1.103;
}
protocols {
  bgp {
    local-as 10;
    group as20 {
      export export-static;
      peer-as 20;
      neighbor 10.1.1.2;
    }
    group as30 {
      export export-static;
      peer-as 30;
      neighbor 2001:db8:1:1019::1;
    }
  }
}
policy-options {
  policy-statement export-static {
    term 1 {
      from protocol static;
      then accept;
    }
  }
}
```

Manipulating Attributes – IOS

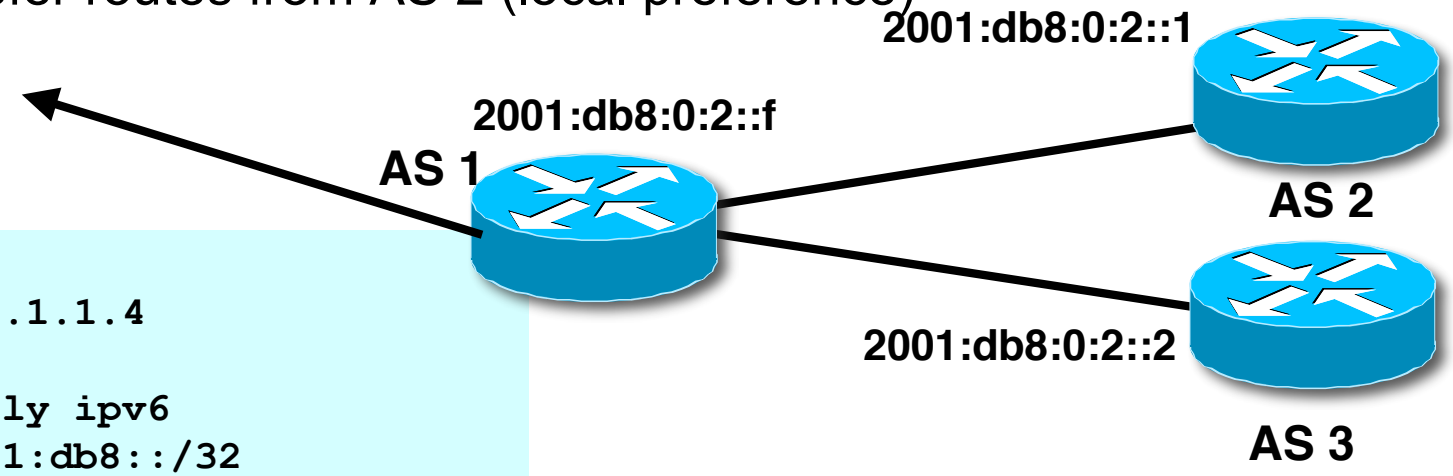
- Prefer routes from AS 2 (local preference)




```
router bgp 1
 no bgp default ipv4-unicast
 neighbor 2001:db8:0:2::1 remote-as 2
 neighbor 2001:db8:0:2::2 remote-as 3
!
 address-family ipv6
 neighbor 2001:db8:0:2::1 activate
 neighbor 2001:db8:0:2::1 prefix-list in-filter in
 neighbor 2001:db8:0:2::1 route-map fromAS2 in
 neighbor 2001:db8:0:2::2 activate
 neighbor 2001:db8:0:2::2 prefix-list in-filter in
 network 2001:db8::/32
 exit-address-family
!
route-map fromAS2 permit 10
 set local-preference 120
```

Manipulating Attributes – IOS-XR

- Prefer routes from AS 2 (local preference)




```
router bgp 1
 router-id 10.1.1.4
 !
 address-family ipv6
  network 2001:db8::/32
 !
 neighbor 2001:db8:0:2::1
  remote-as 2
   route-policy fromAS2 in
   route-policy toAS2 out
 !
 neighbor 2001:db8:0:2::2
  remote-as 3
   route-policy fromAS3 in
   route-policy toAS3 out
! Continued -->
```

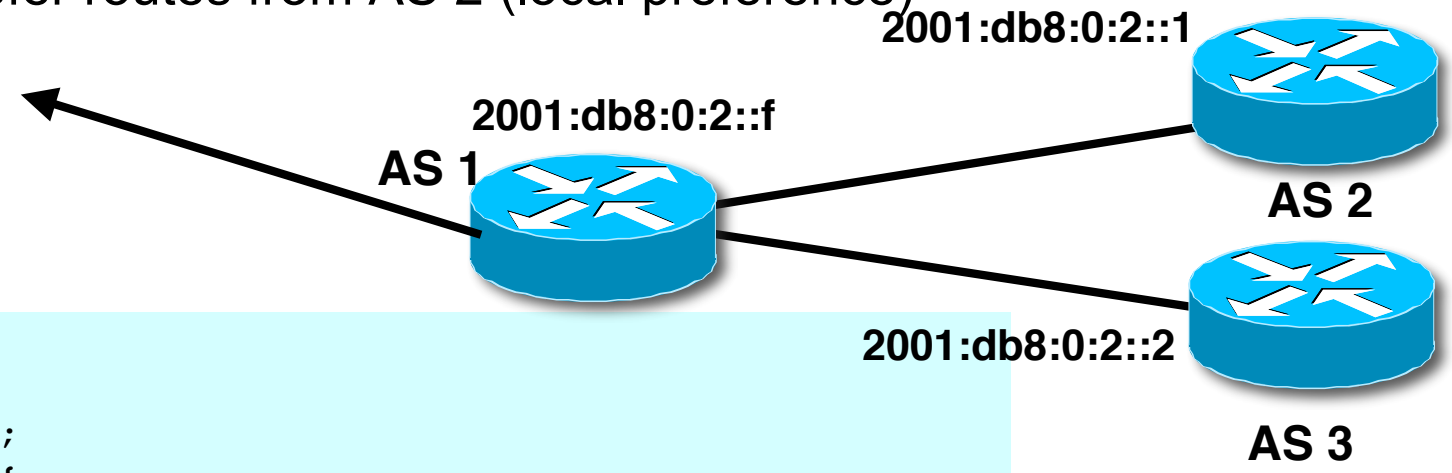
Manipulating Attributes – IOS-XR

```
route-policy fromAS2
 if destination in infilt then
   set local-preference 120
   pass
 endif
end-policy
!
prefix-set infilt
 <prefix>
end-set
!
route-policy toAS2
 if destination in glb-v6 then
   pass
 endif
end-policy
!
prefix-set glb-v6
  ::/0 le 128
end-set
! Continued -->
```

```
route-policy fromAS3
 if destination in infilt then
   pass
 endif
end-policy
!
route-policy toAS3
 if destination in glb-v6 then
   pass
 endif
end-policy
!
```

Manipulating Attributes – JunOS

- Prefer routes from AS 2 (local preference)




```
protocols {
  bgp {
    local-as 1;
    group as2 {
      export export-static2;
      peer-as 2;
      neighbor 2001:db8:0:2::1;
    }
    group as3 {
      export export-static3;
      peer-as 3;
      neighbor 2001:db8:0:2::2;
    }
  }
}
```

Manipulating Attributes – JunOS

```
policy-options {
  policy-statement export-static2 {
    term 1 {
      from protocol static;
      then {
        local-preference 120;
        accept;
      }
    }
  }
 policy-statement export-static3 {
    term 1 {
      from protocol static;
      then {
        local-preference 100;
        accept;
      }
    }
  }
}
```

Carrying IPv4 inside IPv6 peering

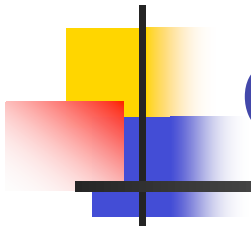
- IPv4 prefixes can be carried inside an IPv6 peering
  - Note that we need to “fix” the next-hop
- Example – IOS

```
router bgp 1
 neighbor 2001:db8:0:2::2 remote-as 2
!
address-family ipv4
 neighbor 2001:db8:0:2::2 activate
 neighbor 2001:db8:0:2::2 route-map ipv4 in
!
route-map ipv4 permit 10
 set ip next-hop 131.108.1.1
```

Carrying IPv4 inside IPv6 peering

- Example – IOS-XR

```
router bgp 1
 bgp router-id 10.1.1.2
 !
 neighbor 2001:db8:0:2::2
  remote-as 2
  address-family ipv4
   route-policy set-nexthop in
   route-policy glb-out out
!
route-policy set-nexthop
  set next-hop 131.108.1.1
  pass
end-policy
!
route-policy glb-out
  pass
end-policy
```



Carrying IPv4 inside IPv6 peering

- JunOS:
  - Can carry IPv6 prefixes in an IPv4 peering
  - Cannot carry IPv4 prefixes in an IPv6 peering
  - When IPv4 prefixes are present, an IPv4 peering is required.

BGP Status Commands IOS & IOS-XR

- IPv6 BGP show commands take ipv6 as argument
  - (Also works for IPv4)

```
show bgp ipv6 unicast <parameter>
```

```
Router1#show bgp ipv6 unicast 2017::/32
BGP routing table entry for 2017::/32, version 11
Paths: (1 available, best #1)
Local
2001:db8:c18:2:1::1 from 2001:db8:c18:2:1::1 (10.10.20.2)
Origin incomplete, localpref 100, valid, internal, best
```

BGP Status Summary – IOS

- Display summary information regarding the state of the BGP neighbours: **`show bgp ipv6 unicast summary`**

```
BGP router identifier 128.107.240.254, local AS number 109
BGP table version is 9030, main routing table version 9030
900 network entries using 134100 bytes of memory
3838 path entries using 291688 bytes of memory
3520/799 BGP path/bestpath attribute entries using 436480 bytes of memory
3464 BGP AS-PATH entries using 91744 bytes of memory
89 BGP community entries using 2152 bytes of memory
1 BGP extended community entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
5715 BGP filter-list cache entries using 68580 bytes of memory
BGP using 1024768 total bytes of memory
BGP activity 2083/1124 prefixes, 11377/7423 paths, scan interval 60 secs

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
2001:5A0:500::9 4  6453   14954    6270     9030    0    0 4d06h           822
```

resource utilisation by the BGP process

Neighbour Information

BGP Messages Activity

BGP Status Summary – IOS-XR

- Display summary information regarding the state of the BGP neighbours: **`show bgp ipv6 unicast summary`**

```
RP/0/0/CPU0:as4byte#sh bgp ipv6 uni sum
BGP router identifier 204.69.200.25, local AS number 2.4
BGP generic scan interval 60 secs
BGP table state: Active
BGP main routing table version 268
BGP scan interval 60 secs
BGP is operating in STANDALONE mode.

Process       RecvTblVer    bRIB/RIB  LabelVer  ImportVer  SendTblVer
Speaker              268         268       268        268         268

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  St/PfxRcd
2001:420:0:8001::1
                  0 65534   98629   58158      268    0    0 05:46:30        6
```

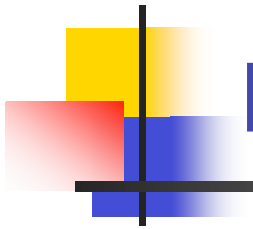
Neighbour Information

BGP Messages Activity

BGP Status Commands – JunOS

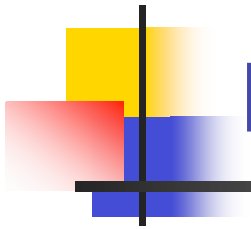
- Display Brief Summary information:

```
regress@UI-J6300-2> show bgp summary
Groups: 1 Peers: 2 Down peers: 0
Table            Tot Paths  Act Paths Suppressed    History Damp State   Pending
inet6.0                  1          1          0          0          0          0
inet6.2                  0          0          0          0          0          0
inet.0                   2          1          0          0          0          0
Peer                     AS  InPkt  OutPkt  OutQ  Flaps Last Up/Dwn State|#Active/Receive/
2001:db8:1::45c         103     48      48     0      1       20:22 Establ
  inet6.0: 1/1/0
10.1.1.1                103     34      35     0      0       14:40 Establ
  inet.0: 1/2/0
```



BGP Status Summary – JunOS

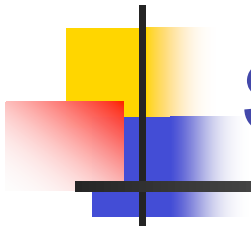
```
regress@UI-J6300-2> show bgp neighbor
Peer: 2001:db8:1::45c+2854 AS 103 Local: 2001:db8:1::45a+179 AS 104
  Type: External    State: Established    Flags: <Sync>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: Cease
  Options: <Preference PeerAS LocalAS Refresh>
  Holdtime: 90 Preference: 170 Local AS: 104 Local System AS: 0
  Number of flaps: 1
  Error: 'Cease' Sent: 1 Recv: 1
  Peer ID: 10.1.1.103       Local ID: 10.1.1.104       Active Holdtime: 90
  Keepalive Interval: 30         Peer index: 0
  BFD: disabled, down
  Local Interface: fe-3/0/0.0
  NLRI advertised by peer: inet6-unicast
  NLRI for this session: inet6-unicast
  Peer supports Refresh capability (2)
  Table inet6.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              1
    Received prefixes:            1
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 18   Sent 5    Checked 59
  Input messages:  Total 50     Updates 1       Refreshes 0     Octets 1023
  Output messages: Total 50     Updates 0       Refreshes 0     Octets 976
  Output Queue[0]: 0
```



BGP: Conclusion

- BGP extended to support multiple protocols
  - IPv6 is but one more address family
- Operators experienced with IPv4 BGP should have no trouble adapting
  - Configuration concepts and CLI is familiar format

Summary



Summary

- Routing Protocols in IPv6 behave as they do in IPv4
  - “96 more bits, no magic”
- Configuration concepts are very similar
- CLI is generally very similar
- Most organisations will deploy IPv6 dual stack with IPv4
  - Simple case of adding IPv6 functionality to existing network