- IPv6 Addressing -

IPv6 Basics

The most widespread implementation of IP currently is IPv4, which utilizes a 32-bit address. Mathematically, a 32-bit address can provide roughly 4 billion unique IP addresses (2^{32} = 4,294,967,296). Practically, the number of usable IPv4 addresses is much lower, as many addresses are reserved for diagnostic, experimental, or multicast purposes.

The explosive growth of the Internet and corporate networks quickly led to an IPv4 address shortage. Various solutions were developed to alleviate this shortage, including CIDR, NAT, and Private Addressing. However, these solutions could only serve as temporary fixes.

In response to the address shortage, **IPv6** was developed. IPv6 increases the address size to 128 bits, providing a nearly unlimited supply of addresses (340,282,366,920,938,463,463,374,607,431,768,211,456 to be exact). This provides roughly 50 *octillion* addresses *per person alive* on Earth today, or roughly 3.7 x 10^{21} addresses per square inch of the Earth's surface.

(References: http://cc.uoregon.edu/cnews/spring2001/whatsipv6.html; http://en.wikipedia.org/wiki/IPv6)

IPv6 offers the following features:

- **Increased Address Space and Scalability –** *providing the absurd number of possible addresses stated previously.*
- **Simplified Configuration –** *allows hosts to auto-configure their IPv6 addresses, based on network prefixes advertised by routers.*
- **Integrated Security –** *provides built-in authentication and encryption into the IPv6 network header*
- **Compatibility with IPv4 –** *simplifies address migration, as IPv6 is backward-compatible with IPv4*

* * *

The IPv6 Address

The IPv6 address is **128 bits**, as opposed to the 32-bit IPv4 address. Also unlike IPv4, the IPv6 address is represented in hexadecimal notation, separate by colons.

An example of an IPv6 address would be:

1254:1532:26B1:CC14:0123:1111:2222:3333

Each "grouping" (from here on called **fields**) of hexadecimal digits is 16 bits, with a total of eight fields. The hexadecimal values of an IPv6 address are **not case-sensitive.**

We can drop any leading zeros in each field of an IPv6 address. For example, consider the following address:

1423:0021:0C13:CC1E:3142:0001:2222:3333

We can condense that address to: 1423:21:C13:CC1E:3142:1:2222:3333

Only leading zeros can be condensed. If we have an entire field comprised of zeros, we can further compact the following address:

F12F:0000:0000:CC1E:2412:1111:2222:3333

The condensed address would be: F12F::CC1E:2412:1111:2222:3333

Notice the double colons (::). We can only condense one set of contiguous zero fields. Thus, if we had the following address:

F12F:0000:0000:CC1E:2412:0000:0000:3333

We could not condense that to: F12F::CC1E:2412::3333

The address would now be ambiguous, as we wouldn't know how many "0" fields were compacted in each spot. Remember that we can only use one set of double colons in an IPv6 address!

* * *

The IPv6 Prefix

IPv4 utilizes a *subnet mask* to define the network "prefix" and "host" portions of an address. This subnet mask can also be represented in Classless Inter-Domain Routing (CIDR) format.

IPv6 always use **CIDR** notation to determine what bits **notate the prefix** of an address:

Full Address:	1254:1532:26B1:CC14:123:1111:2222:3333/64
Prefix ID:	1254:1532:26B1:CC14:
Host ID:	123:1111:2222:3333

The */64* indicates that the first 64 bits of this address identify the prefix.

The IPv6 Interface ID and EUI-64 Format

The host portion of an IPv4 address is not based on the hardware address of an interface. IPv4 relies on **Address Resolution Protocol (ARP)** to map between the logical IP address and the **48-bit** hardware **MAC address**.

IPv6 unicasts generally allocate the first 64 bits of the address to identify the network (**prefix**), and the last 64 bits to identify the host (referred to as the **interface ID**). The interface ID *is* based on the interface's hardware address.

This interface ID adheres to the IEEE **64-bit Extended Unique Identifier** (**EUI-64**) format. Since most interfaces still use the 48-bit MAC address, the MAC must be converted into the EUI-64 format.

Consider the following MAC address: 1111.2222.3333. The first 24 bits, the Organizationally Unique Identifier (OUI), identify the manufacturer. The last 24 bits uniquely identify the host. To convert this to EUI-64 format:

1. The **first 24 bits** of the MAC (the **OUI**), become the first 24 bits of the EUI-64 formatted interface ID.
2. The ***seventh*** bit of the OUI is changed from a "**0**" to a "**1**".
3. The next 16 bits of the interface ID are **FFFE.**
4. The **last 24 bits** of the MAC (the **host ID**), become the last 24 bits of the interface ID.

Thus, the MAC address 1111.2222.3333 in EUI-64 format would become **1311:22FF:FE22:3333**, which becomes the interface ID.

* * *

The IPv6 Address Hierarchy

IPv4 separated its address space into specific **classes.** The class of an IPv4 address was identified by the high-order bits of the first octet:

- **Class A** - (00000001 – 01111111, or 1 - 127)
- **Class B -** (10000000 – 10111111, or 128 - 191)
- **Class C -** (11000000 – 11011111, or 192 - 223)
- **Class D -** (11100000 – 11101111, or 224 - 239)

IPv6's addressing structure is far more scalable. Less than 20% of the IPv6 address space has been designated for use, currently. The potential for growth is enormous.

The address space that *has* been allocated is organized into several types, determined by the high-order bits of the first field:

- **Special Addresses –** addresses begin **00*xx*:**
- **Link Local –** addresses begin **FE8*x*:**
- **Site Local –** addresses begin **FEC*x*:**
- **Aggregate Global –** addresses begin **2*xxx*:** or **3*xxx*:**
- **Multicasts –** addresses begin **FF*xx*:**
- **Anycasts**

(Note: an "x" indicates the value can be any hexadecimal number)

There are **no broadcast addresses** in IPv6. Thus, any IPv6 address that is not a *multicast* is a *unicast* address.

Anycast addresses identify a group of interfaces on multiple hosts. Thus, multiple hosts are configured with an *identical* address. Packets sent to an anycast address are sent to the *nearest* (i.e., least amount of hops) host. Anycasts are indistinguishable from any other IPv6 unicast address.

Practical applications of anycast addressing are a bit murky. One possible application would be a server farm providing an identical service or function, in which case anycast addressing would allow clients to connect to the nearest server.

Special (Reserved) IPv6 Addresses

The first field of a **reserved** or **special** IPv6 address will always begin **00*xx*.** Reserved addresses represent 1/256th of the available IPv6 address space.

Various reserved addresses exist, including:

- **0:0:0:0:0:0:0:0** (or **::**) – is an **unspecified** or **unknown** address. It is the equivalent of the IPv4 0.0.0.0 address, which indicates the absence of a configured or assigned address. In routing tables, the unspecified address is used to identify **all** or **any** possible hosts or networks.
- **0:0:0:0:0:0:0:1 (**or **::1**) – is the **loopback** or **localhost** address. It is the equivalent of the IPv4 127.0.0.1 address.

Reserved Addresses - IPv4 and IPv6 Compatibility

To alleviate the difficulties of immediately migrating from IPv4 to IPv6, specific reserved addresses can be used to *embed* an IPv4 address into an IPv6 address.

Two types of addresses can be used for IPv4 embedding, **IPv4-compatible IPv6 addresses**, and **IPv4-mapped IPv6 addresses**.

- **0:0:0:0:0:0:a.b.c.d (**or **::a.b.c.d)** – is an **IPv4-*compatible* IPv6 address**. This address is used on devices that support both IPv4 *and* IPv6. A prefix of /96 is used for IPv4-compatible IPv6 addresses:

  ::192.168.1.1/96

- **0:0:0:0:0:FFFF:a.b.c.d** (or **::FFFF:a.b.c.d) –** is an **IPv4-*mapped* IPv6 address.** This address is used by IPv6 routers and devices to identify *non*-IPv6 capable devices. Again, a prefix of /96 is used for IPv4-mapped IPv6 addresses:

  ::FFFF:192.168.1.1/96

* * *

Link-Local IPv6 Addresses

Link-local IPv6 addresses are used only on a single link (subnet). Any packet that contains a link-local source or destination address is *never routed* to another link. Every IPv6-enabled interface on a host (or router) is assigned a link-local address. This address can be manually assigned, or auto-configured.

The first field of a **link-local** IPv6 address will always begin ***FE8x (1111 1110 10).*** Link-local addresses are **unicasts,** and represent 1/1024th of the available IPv6 address space. A prefix of **/10** is used for link-local addresses.

FE80::1311:22FF:FE22:3333/10

There is no hierarchy to a link-local address:

- The first 10 bits are fixed (**FE8**), known as the **Format Prefix (FP).**
- The next 54 bits are set to **0.**
- The final 64 bits are used as the **interface ID**.

Site Local IPv6 Addresses

Site-local IPv6 addresses are the equivalent of "private" IPv4 addresses. Site-local addresses can be routed within a *site* or *organization*, but cannot be globally routed on the Internet. Multiple private subnets within a "site" are allowed.

The first field of a **site-local** IPv6 address will always begin ***FECx (1111 1110 11).*** Site-local addresses are **unicasts,** and represent 1/1024th of the available IPv6 address space.

FEC0::2731:E2FF:FE96:C283/64

Site-local addresses do adhere to a hierarchy:

- The first 10 bits are the fixed FP (**FEC**).
- The next 38 bits are set to **0.**
- The next 16 bits are used to identify the **private subnet ID**.
- The final 64 bits are used as the **interface ID**.

To identify two separate subnets (*1111* and *2222*):

FEC0::1111:2731:E2FF:FE96:C283/64
FEC0::2222:97A4:E2FF:FE1C:E2D1/64

* * *

Aggregate Global IPv6 Addresses

Aggregate Global IPv6 addresses are the equivalent of "public" IPv4 addresses. Aggregate global addresses can be routed publicly on the Internet. Any device or site that wishes to traverse the Internet must be uniquely identified with an aggregate global address.

Currently, the first field of an **aggregate global** IPv6 address will always begin **2*xxx* (001).** Aggregate global addresses are **unicasts,** and represent 1/8th of the available IPv6 address space.

2000::2731:E2FF:FE96:C283/64

Aggregate global addresses adhere to a very strict hierarchy:

- The first 3 bits are the fixed FP.
- The next 13 bits are the **top-level aggregation identifier (TLA ID).**
- The next 8 bits are **reserved** for future use.
- The next 24 bits are the **next-level aggregation identifier (NLA ID).**
- The next 16 bits are the **site-level aggregation identifier (SLA ID).**
- The final 64 bits are used as the **interface ID**.

By have multiple **levels**, a consistent, organized, and scalable hierarchy is maintained. High level registries are assigned ranges of TLA IDs. These can then be subdivided in the NLA ID field, and passed on to lower-tiered ISPs.

Such ISPs allocate these prefixes to their customers, which can further subdivide the prefix using the SLA ID field, to create whatever local hierarchy they wish. The 16-bit SLA field provides up to 65535 networks for an organization.

Note: Do not confuse the SLA ID field of a global address field, with a site-local address. Site-local addresses cannot be routed publicly, where as SLA ID's are just a subset of the publicly routable aggregate global address.

* * *

Multicast IPv6 Addresses

Multicast IPv6 addresses are the equivalent of IPv4 multicast addresses. Interfaces can belong to one or more multicast **groups**. Interfaces will accept a multicast packet only if they belong to that group. Multicasting provides a much more efficient mechanism than **broadcasting**, which requires that every host on a link accept and process each broadcast packet.

The first field of a **multicast** IPv6 address will always begin **FF*xx* (1111 1111).** The full multicast range is **FF00** through **FFFF**. **Multicasts** represent 1/256th of the available IPv6 address space.

FF01:0:0:0:0:0:0:1

Multicast addresses follow a specific format:

- The first 8 bits **identify the address** as a **multicast** (1111 1111)
- The next 4 bits are a **flag value**. If the flag is set to all zeroes (0000), the multicast address is considered *well-known*.
- The next 4 bits are a **scope value**:
  - 0000 (0) = Reserved
  - 0001 (1) = Node Local Scope
  - 0010 (2) = Link Local Scope
  - 0101 (5) = Site Local Scope
  - 1000 (8) = Organization Local Scope
  - 1110 (e) = Global Scope
  - 1111 (f) = Reserved
- The final 112 bits identify the actual **multicast group**.

IPv4 multicast addresses had no mechanism to support multiple "**scopes."** IPv6 scopes allow for a multicast hierarchy, a way to *contain* multicast traffic.

* * *

Common IPv6 Multicast Addresses

The following is a list of common, well-known IPv6 multicast addresses:

Node-Local Scope Multicast Addresses

- FF01::1 – All-nodes address
- FF01::2 – All-routers address

Link-Local Scope Multicast Addresses

- FF02::1 – All-nodes address
- FF02::2 – All-routers address
- FF02::5 – OSPFv3 (OSPF IPv6) All SPF Routers
- FF02::6 – OSPFv3 Designated Routers
- FF02::9 – RIPng Routers
- FF02::13 – PIM Routers

Site-Local Scope Multicast Addresses

- FF05::2 – All-routers address

All hosts must join the **all-nodes** multicast group, for both the node-local and link-local scopes. All routers must join the **all-routers** multicast group, for the node-local, link-local, and site-local scopes.

Every site-local and aggregate global address is assigned a **solicited-node multicast** address. This solicited-node address is created by appending the last 24 bits of the interface ID to the following prefix: FF02::1:FF/103.

Thus, if you have a site-local address of:

FEC0::1111:2731:E2FF:FE96:C283

The corresponding solicited-node multicast address would be:

FF02::1:FF96:C283

Solicited-node multicast addresses are most often used for neighbor discovery (covered in an upcoming section in this guide).

* * *

Required IPv6 Addresses

At a minimum, each IPv6 interface on a **host** must recognize the following IPv6 addresses:

- The loopback address
- A link-local address
- Any configured site-local or aggregate global addresses
- Any configured multicast groups
- The all-nodes multicast address (both node-local and link-local scopes)
- The solicited-node multicast address for any configured unicast addresses

In *addition* to the above addresses, each IPv6 interface on a **router** must recognize the following IPv6 addresses:

- The subnet-router anycast address
- Any configured multicast groups
- The all-routers multicast address (node-local, link-local, and site-local scopes)

IPv6 Addresses and URLs

IPv6 addresses can also be referenced in **URLs** (Uniform Resource Locator). URL's, however, use the colon to represent a specific TCP "port". This is not an issue with IPv4 addresses, which can easily be referenced using a URL:

http://192.168.1.1/index.html

Because IPv6 fields are separated by colons, the IPv6 address must be placed in brackets, to conform to the URL standard:

http://[FEC0::CC1E:2412:1111:2222:3333]/index.html

* * *

The IPv6 Header

The IPv6 header has **8 fields** and is **320 bits** long. It has been considerably streamlined compared to its IPv4 counterpart, which has **12 fields** and is **160 bits** long.

Field	*Length*	*Description*
Version	4 bits	*Version of IP (in this case, IPv6)*
Traffic Class	8 bits	*Classifies traffic for QoS*
Flow Label	20 bits	*Identifies a flow between a source and destination*
Payload Length	16 bits	*Length of data in packet*
Next Header	8 bits	*Specifies the next upper-layer or extension header*
Hop Limit	8 bits	*Decremented by each router traversed*
Source Address	128 bits	*Source IPv6 address*
Destination Address	128 bits	*Destination IPv6 address*

The *Next Header* field is of some importance. This field can identify either the next upper-layer header (for example, UDP, TCP or ICMP), or it can identify a special **Extension Header**, which placed in between the IPv6 and upper layer header.

Several such extension headers exist, and are usually processed in the following order:

- **Hop-by-Hop Options** – *specifies options that should be processed by every router in the path. Directly follows the IPv6 header.*
- **Destination Options –** *specifies options that should be processed by the destination device.*
- **Routing Header** – *specifies each router the packet must traverse to reach the destination (source routing)*
- **Fragment Header** – *used when a packet is larger than the MTU for the path*
- **Authentication Header –** *used to integrate IPSEC Authentication Header (AH) into the IPv6 packet*
- **ESP Header –** *used to integrate IPSEC Encapsulating Security Payload (ESP) into the IPv6 packet*

(Reference: http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t2/ipv6/ftipv6o.htm#1004285)

* * *

ICMPv6

ICMP Version 6 (**ICMPv6**) is a core component of IPv6. All devices employing IPv6 must also integrate ICMPv6.

ICMPv6 provides many services, including (but not limited to):

- Error Messages
- Informational messages (such as *echo* replies for IPv6 ping)
- MTU Path Discovery
- Neighbor Discovery

There are four key ICMPv6 error messages:

- **Destination Unreachable (**ICMP packet type 1**) –** *indicates that the packet cannot be forwarded to its destination. The node sending this message includes an explanatory code:*
  - **0 -** No route to destination
  - **1** - Access is administratively prohibited
  - **3** - Address unreachable
  - **4** - Port unreachable

- **Packet Too Big (**ICMP packet type 2**)** – *indicates the packet is larger than the MTU of the link. IPv6* ***routers do not fragment*** *packets. Instead, the Packet Too Big message is sent to the source (sending) device, which then reduces (or fragments) the size of the packet to the reported MTU. This message is used for* ***Path MTU Discovery (PMTUD)****.*

- **Time Exceeded (**ICMP packet type 3**)** – *indicates that the hop count limit has been reached, usually indicating a routing loop*

- **Parameter Problem (**ICMP packet type 4**) –** *indicates an error in the IPv6 header, or an IPv6 extension header. The node sending this message includes an explanatory code:*
  - **0 -** Erroneous header field
  - **1 -** Unrecognized next-header type
  - **2 -** Unrecognized IPv6 option

*(*Reference: *http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a0080113b1c.shtml)*

* * *

Neighbor Discovery Protocol (NDP) and ICMPv6

The **neighbor discovery protocol (NDP)** provides a multitude of services for IPv6 enabled devices, including:

- Automatic address configuration, and prefix discovery
- Duplicate address detection
- MTU discovery
- Router discovery
- Address resolution

NDP replaces many IPv4 specific protocols, such as DHCP and ARP. NDP utilizes **ICMPv6** to provide the above services.

Periodically, IPv6 routers send out **Router Advertisements (RA's)** to both announce their presence on a link, and to provide auto-configuration information for hosts. This **RA** (ICMP packet type 134) is sourced from the link-local address of the sending router, and sent to the link-scope all-nodes multicast group. The sending router sets a **hop limit** of **255** on a RA; however, the RA packet *must not* be forwarded outside the local link.

Hosts use RA's to configure themselves, and add the router to its local default router list. A host can request an RA by sending out a **Router Solicitation (RS,** ICMP packet type 133) to the link-local all-routers multicast address. A RS is usually sent when a host is *not* currently configured with an IP address.

The RA messages contain the following information for hosts:

- The **router's link-layer address** (to be added to the host's default router list)
- One or more **network prefixes**
- A **lifetime** (measured in seconds) for the prefix(es)
- The link **MTU**

Routers send **Redirect** messages to hosts, indicating a *better* route to a destination. Hosts can have multiple routers in its default router list, but one is chosen as the *true* default router. If this default router deems that another router has a better route to the destination, it forwards the Redirect message to the sending host.

* * *

Neighbor Discovery Protocol (NDP) and ICMPv6 (continued)

Neighbor Solicitations (NS's, ICMP packet type 135**)** are sent by hosts to identify the link-layer address of a neighbor, and ensure its reachability. A NS message's source address is the link-local address of the sending host, and the destination is the **solicited-node multicast** address of the destination host.

A neighbor will reply to a NS with a **Neighbor Advertisement (NA,** ICMP packet type 136). This process replaces the Address Resolution Protocol (ARP) used by IPv4, and provides a far more efficient means to learn neighbor address information.

Hosts additionally use the NS messages to **detect duplicate addresses**. Before a host assigns itself an IPv6 address, it sends out a NS to ensure no other host is configured with that address.

Autoconfiguration of Hosts

Hosts can be assigned IPv6 addresses one of two ways: manually, or using autoconfiguration. Hosts learn how to autoconfigure themselves from **Router Advertisements (RA's).**

Two types of autoconfiguration exist, **stateless** and **stateful**.

When using **Stateless Autoconfiguration**, a host first assigns itself a link-local IPv6 address. It accomplishes this by combining the link-local prefix (FE8) with its interface ID (MAC address in EUI-64 format).

The host then sends a **Router Solicitation** multicast to the all-routers multicast address, which provides one or more network prefixes. The host combines these prefixes with its interface ID to create its site-local (or aggregate global) IPv6 addresses.

Stateful Autoconfiguration is used in conjunction with stateless autoconfiguration. Stateful Autoconfiguration utilizes DHCPv6 to provide additional information to the host, such as DNS servers. DHCPv6 can also be used in the event that there is no router on the link, to provide stateless autoconfiguration.

* * *

Configuring IPv6 Addresses

IPv6 support is **disabled** by default on Cisco routers, and must be enabled globally:

Router(config)# *ipv6 unicast-routing*

To configure an interface to auto-configure a link-local IPv6 address:

Router(config)# *interface e0*
Router(config-if)# *ipv6 enable*

To manually configure a site-local IPv6 address on an interface:

Router(config)# *interface e0*
Router(config-if)# *ipv6 address FEC0::/64 eui-64*

The *eui-64* parameter will append interface ID (MAC address in EUI-64 format) to the site-local prefix. Otherwise, we could have specified the full IPv6 address:

Router(config-if)# *ipv6 address FEC0::1:1234:23FF:FE21:1212 eui-64*

Recall that we can configure multiple subnets for our site-local address space:

Router(config)# *interface e0*
Router(config-if)# *ipv6 address FEC0::2222:0:0:0:0/64 eui-64*

To configure a router interface to advertise a specific prefix to hosts on the link:

Router(config)# *interface e0*
Router(config-if)# *ipv6 nd prefix-advertisement 2002:1111::/48 2000 1000 onlink autoconfig*

The router will *advertise* a *prefix* of *2002:1111::/48* with a **valid lifetime** of *2000* seconds and a **preferred lifetime** of *1000* seconds. The clients will *autoconfig* themselves based on this prefix.

To view IPv6 specific information about an interface:

Router# *show ipv6 interface e0*

To create a static host entry for an IPv6 address:

Router(config)# *ipv6 host MYHOST FEC0::1111:2731:E2FF:FE96:C283*

* * *

Configuring IPv6 Static Routes

The syntax to configure an IPv6 static route is simple:

Router(config)# *ipv6 route FEC0::2222/64 FEC0::1111:3E5F:2E5B:A3D1*

The above command creates an *ipv6 route* to the *FEC0::2222/64* network, with a next-hop of *FEC0::1111:3E5F:2E5B:A3D1*.

To create an IPv6 default route:

Router(config)# *ipv6 route ::/0 FE80::2*

The above command creates an *ipv6* default *route*, with a next hop of *FE80::2*. The *::/0* designation indicates all zeros in the address field, and a mask of zero bits (the **unspecified** address).

To view the IPv6 routing table:

Router(config)# *show ipv6 route*

* * *

Configuring IPv6 RIPng

A version of RIP for IPv6 was developed called **RIPng** (RIP Next Generation). Functionally, RIPng is the equivalent of RIPv2, with the additional support for IPv6 addresses. However, RIPng is *not* backwards with earlier version of RIP, and does not support IPv4 addressing.

Basic RIPng characteristics:

- Administrative distance of 120
- Maximum hopcount of 16
- Updates are sent every 30 seconds as multicasts

To configure RIPng, we must first enable the RIP process globally:

Router(config)# *ipv6 router rip MYPROCESS*

We are enabling an *ipv6 rip* process called *MYPROCESS.* Next, we must enable RIPng on each participating interface:

Router(config)# *interface e0*
Router(config-if)# *ipv6 rip MYPROCESS enable*

RIPng, by default, utilizes **UDP** port **521** and multicast group **FF02::9,** but these parameters can be changed globally:

Router(config)# *ipv6 rip MYPROCESS port 555 multicast-group FF02::1111*

We can adjust RIPng's timers:

Router(config)# *ipv6 rip MYPROCESS timers 30 180 180 120*

In order, the above timers are *update, expire, holddown,* and *garbage-collect*. The above values are default.

To control inbound or outbound RIPng updates, using an access-list:

Router(config)# *interface e0*
Router(config-if)# *ipv6 rip MYPROCESS input-filter MYACCESSLIST*
Router(config-if)# *ipv6 rip MYPROCESS output-filter MYACCESSLIST*

To view configuration and status information for RIPng:

Router# *show ipv6 protocols*
Router# *show ipv6 rip*

* * *

Configuring IPv6 OSPF (OSPFv3)

OSPFv2 is a widely used link-state routing protocol in IPv4 environments. To support IPv6, **OSPFv3** was developed. Its function is very similar to OSPFv2.

First, we must first enable the OSPF process globally:

Router(config)# *ipv6 router ospf 1*

The *1* indicates the process ID. Next, we must place the participating interfaces in their appropriate areas:

Router(config)# *interface e0*
Router(config-if)# *ipv6 ospf 1 area 0*

Router(config)# *interface s0*
Router(config-if)# *ipv6 ospf 1 area 1*

Please note: the Router ID for OSPFv3 is still a 32-bit value. Thus, the highest IPv4 loopback address will be chosen first, then the highest IPv4 physical address. If neither exist, a 32-bit Router ID must be manually specified:

Router(config)# *ipv6 router ospf 1*
Router(config-router)# *router-id 1.1.1.1*

To create a summarized route on an area boundary:

Router(config)# *ipv6 router ospf 1*
Router(config-router)# *area range 2001:1111::/48*

To view configuration and status information for OSPFv3:

Router# *show ipv6 ospf neighbor*
Router# *show ipv6 ospf interface*

To clear an OSPFv3 process:

Router# *clear ipv6 ospf 1*

* * *

Configuring IPv6 BGP

BGP-4 does not natively support IPv6. Support for IPv6 and other protocols (such as IPX) are included in the **BGP Multi-protocol Extensions**.

Basic BGP configuration using IPv6 is identical to that of IPv4:

Router(config)# *router bgp 100*
Router(config-router)# *neighbor 2005:2222::1 remote-as 200*

Notice the use of an aggregate global IPv6 address in the *neighbor* statement.

Additional information is required - we must *activate* the neighbor. This allows the neighbor to share IPv6 routes with the local router:

Router(config)# *router bgp 100*
Router(config-router)# *address-family ipv6*
Router(config-router-af)# *neighbor 2005:2222::1 activate*

To advertise an IPv6 prefix into BGP:

Router(config)# *router bgp 100*
Router(config-router)# *address-family ipv6*
Router(config-router-af)# *network 2005:1111:: /24*

To view configuration and status information for IPv6 BGP:

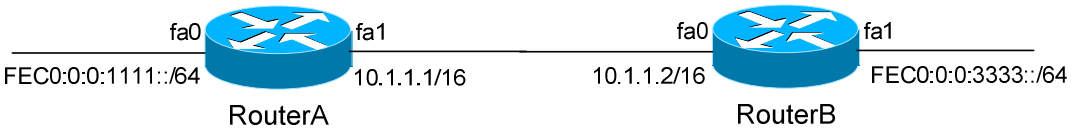
Router# *show bgp ipv6*

Router# *show bgp ipv6 summary*

(Reference: http://www.cisco.com/en/US/products/sw/iosswrel/ps5187/products_configuration_guide_chapter09186a00801d65f7.html)

* * *

Configuring an IPv6 Tunnel



We can configure an IPv6 "tunnel" across an IPv4 link. To accomplish this, we create a virtual tunnel interface on both RouterA and RouterB.

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

RouterA(config)#  interface fa0
RouterA(config-if)#  ipv6 address FEC0:0:0:1111::/64 eui-64

RouterA(config)#  interface fa1
RouterA(config-if)#  ip address 10.1.1.1 255.255.0.0

RouterA(config)#  interface tunnel0
RouterA(config-if)#  no ip address
RouterA(config-if)#  ipv6 address FEC0:0:0:2222::1/124
RouterA(config-if)#  tunnel source fa1
RouterA(config-if)#  tunnel destination 10.1.1.2
RouterA(config-if)#  tunnel mode ipv6ip
```

Configuration on Router B:

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

RouterB(config)#  interface fa0
RouterB(config-if)#  ip address 10.1.1.2 255.255.0.0

RouterB(config)#  interface fa1
RouterB(config-if)#  ipv6 address FEC0:0:0:3333::/64 eui-64

RouterB(config)#  interface tunnel0
RouterB(config-if)#  no ip address
RouterB(config-if)#  ipv6 address FEC0:0:0:2222::2/124
RouterB(config-if)#  tunnel source fa1
RouterB(config-if)#  tunnel destination 10.1.1.1
RouterB(config-if)#  tunnel mode ipv6ip
```

We've applied an IPv6 address on the *FEC0:0:0:2222::/124* network. IPv6 traffic can now route across the 10.1.x.x/16 IPv4 network. Any routing protocol configuration for IPv6 should be completed on the tunnel interfaces.

* * *

IPv6 Access-Lists

Cisco IOS 12.0(23) or later supports IPv6 access-lists. The configuration is similar to that of IPv4 named access-lists (All IPv6 access-lists are **named**; there are *no* IPv6 numbered access-lists).

```
Router(config)# ipv6 access-list MYLIST
Router(config-access-list)# deny ipv6 any 2001:1111::/64
Router(config-access-list)# permit ipv6 any any

Router(config)# interface fa0/0
Router(config-if)# ipv6 traffic-filter MYLIST in
```

Notice the use of a */prefix*, as opposed to a *wildcard* mask.

Also, notice the use of the *ipv6 traffic-filter* command to apply the ACL to the interface, as opposed to *ip access-group*.

Hurray for consistency!

(Reference: *http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t2/ipv6/ftipv6c.htm#1064881*)

* * *