Routing TCP/IP Vol 1 Notes

7 Jun 2008

Chapter 1: TCP/IP Review

Internet Protocol (IPv4)

Version (4 bits) - IP version (4 or 6)

Header lentgh (4 bits) - Length of header plus any options

Type of Service (TOS) (8 bits) - Used for QoS; can also be evaluated as *DiffServ Code Point (DSCP)*

Total length (16 bits) - Total packet size

Identifier (16 bits) - Identifies fragments belonging to a single original packet

Fragmentations flags (3 bits) - Three flags: unused, *Don't Fragment (DF)*, and *More Fragments (MF)*

Fragment offset (13 bits) - Specifies the offset of a fragment from the beginning of the original packet (in units of eight bytes)

Time To Live (TTL) (8 bits) - Tracks hop count

Protocol (8 bits) - Identifies the upper-layer protocol

Header checksum (16 bits) - Used for header error detection

Source address (32 bits)

Destination address (32 bits)

Options (variable length) - Optional attributes generated by the originator

Common IP options:

Loose source routing - A list of IP addresses (router interfaces) the packet should traverse

Strict source routing - A routing path which must be followed exactly

Record route - Routers traversed record the address of their outbound interface on the packet

Timestamp - Like record route but also includes a timestamp

Address Resolution Protocol (ARP)

ARP header:

Hardware type (16 bits) - Identifies the type of layer 2 technology (Ethernet, HDLC, etc)

Protocol type (16 bits) - Identifies the network-layer protocol

Hardware address length (8 bits) - Length of the data link address in bytes (e.g. MAC = 6)

Protocol address length (8 bits) - Length of the network address in bytes (e.g. IP = 4)

Operation (16 bits) - Packet type (request/reply type)

Sender's hardware address

Sender's network address

Target hardware address

Target network address

Cisco routers cache ARP entries for four hours by default (this can be modified with `arp timeout <seconds>` at interface configuration).

Proxy ARP

Proxy arp allows a router to issue ARP replies to one subnet on behalf of a host in another subnet, to facilitate inter-subnet communication with a host not configured with a default gateway.

The proxy ARP reply will contain the router's own hardware address for that subnet.

Gratuitous ARP

A gratuitous ARP request is one requesting a reply for the sender's own IP address.

Gratuitous ARP can be used to check for duplicate addresses or to announce the existence of a new host.

Reverse ARP (RARP)

Reverse ARP occurs when the sender requests the network address for a given hardware address.

For purposes of initial device addressing, RARP has been superseded by BOOTP and DHCP.

Internet Control Message Protocol (ICMP)

ICMP header:

Type (8 bits)

Code (8 bits)

Checksum (16 bits)

Other fields... (variable)

Common ICMP types:

- **0** - Echo reply
- **3** - Destination unreachable
- **5** - Redirect
- **6** - Alternate host address
- **8** - Echo
- **9** - Router advertisement
- **10** - Router selection
- **11** - Time exceeded
- **12** - Parameter problem
- **13** - Timestamp
- **14** - Timestamp reply
- **30** - Traceroute

Transmission Control Protocol (TCP)

TCP header:

- **Source port** (16 bits)
- **Destination port** (16 bits)
- **Sequence number** (32 bits) - Identifies the position of a segment within a stream
- **Acknowledgment number** (32 bits) - Identifies the sequence number the source next expects to receive
- **Header length** (4 bits)
- **Reserved** (4 bits)
- **Flags** (8 bits)
- **Window size** (16 bits) - Flow control; specifies the amount of data that may be transmitted from the peer between acknowledgments
- **Checksum** (16 bits) - Error detection for the header and payload
- **Urgent pointer** (16 bits) - Points to the end of urgent data; used only when the URG flag is set
- **Options** (variable)

TCP flags:

CWR - Congestion window reduced

ECE - ECN-Echo

URG - Urgent data

ACK - Acknowledgment

PSH - Push

RST - Reset

SYN - Synchronize

FIN - Final

User Datagram Protocol (UDP)

UDP header:

Source port (16 bits)

Destination port (16 bits)

Length (16 bits)

Checksum (16 bits)

Chapter 2: IPv6 Overview

IPv6 Addressing

IPv6 addresses are presented in 16-bit hexadecimal groups separated by colons. For example, `3ffe:1944:0100:000a:0000:00bc:2500:0d0b`.

Shorthand rules:

One group of all-zero segments can be presented with a double-colon (`::`)

Leading zeros in each segment may be omitted

Subnet identification is performed in CIDR (bit count) notation (`/64`).

`::/0` indicates an all-zeros or wildcard address.

`::/128` represents an unspecified address.

Address Types

IPv6 addresses can be one of three types: unicast, anycast, or multicast.

Broadcast functionality is provided by the "all-nodes" multicast address.

Address types are identified by their leading bits:

Binary	Hex	Type
11111111	FF00::/8	Multicast
11111110 10	FE80::/10	Link-local unicast
11111110 11	FEC0::/10	Site-local unicast (deprecated)
001	2000::/3	Global unicast (currently allocated)

Global Unicast

A global unicast address is broken into three sections:

Global routing prefix (48 bits)

Subnet ID (16 bits)

Interface ID (64 bits)

Local Unicast

Link-local unicasts are unique only to a single layer 2 link.

Site-local unicasts were defined in the original IPv6 standard but have been replaced by *Unique Local Addresses* (`FC00::/7`) in **RFC 4193**.

Anycast

An anycast address is one address configured on multiple end nodes; dynamic routing will ideally forward traffic to the "nearest" or least-cost anycast server.

Any global unicast address applied to more than one device can be considered any anycast address.

Multicast

A multicast address identifies a logical group of devices.

Multicast address structure:

Multicast prefix (8 bits) - Always `0xFF`

Flags (4 bits)

Scope (4 bits)

Group ID (112 bits)

Address Scopes:

`0x0` - Reserved

`0x1` - Node-local

`0x2` - Link-local

`0x5` - Site-local

`0x8` - Org-local

`0xE` - Global

`0xF` - Reserved

Embedded IPv4 Addresses

Different transition technologies have different ways of embedding an IPv4 address in an IPv6 address. Some examples for `10.23.1.5` are:

`FE80::5EFE:10.23.1.5` (ISATAP)

`::FFFF:10.23.1.5` (SIIT)

`FEC0:0:0:1::10.23.1.5` (TRT)

`2002:0A17:0105::/48` (6to4)

IPv6 Header

IPv6 headers have a fixed 40-byte length.

Header format:

Version (4 bits) - Always set to 6

Traffic class (8 bits) - DiffServ Code Point (DSCP)

Flow label (20 bits) - An arbitrary field for differentiating traffic flows

Payload length (16 bits) - Indicates the length of the payload (header length is not included)

Next header (8 bits) - Identifies the extension header or upper-layer protocol that follows

Hop limit (8 bits) - Decrementing hop counter (TTL)

Source address (128 bits)

Destination address (128 bits)

Extension Headers

Extension headers provide for optional extended capabilities such as hop-by-hop options and IPsec encryption.

Next header values:

- **0** - Hop-by-hop options
- **43** - Routing
- **44** - Fragment
- **50** - ESP
- **51** - AH
- **59** - No next header
- **60** - Destination options

If a header is the last in the stack, its next header field will identify the upper-layer protocol that follows (e.g. 6 for TCP or 17 for UDP).

RFC 1883 specifies the order in which extensions headers should appear if they are used.

ICMPv6

IPv6 implements its own version of ICMP, defined in **RFC 2463**.

Like ICMPv4, ICMPv6 uses type/code pairings to identify field types.

Common field types:

- **1** - Destination unreachable
- **2** - Packet too big
- **3** - Time exceeded
- **4** - Parameter problem
- **128** - Echo request
- **129** - Echo reply
- **130** - Group membership query
- **131** - Group membership report
- **132** - Group membership reduction

Neighbor Discovery Protocol (NDP)

NFP is defined in **RFC 4861**

NDP functions:

- **Router discovery**
- **Prefix discovery**
- **Parameter discovery** - Link MTU, etc.
- **Address autoconfiguration** - Replaces DHCP
- **Address resolution** - Replaces ARP
- **Next-hop determination** - Link-layer address for next hop
- **Neighbor unreachability detection**
- **Duplicate address detection**
- **Redirect** - A router can inform a host of a better path out of the link

NDP uses ICMPv6 to exchange messages.

NDP messages types:

- **Router Solicitation (RS)** (Type 133) - Sent by hosts to request an RA
- **Router Advertisement (RA)** (Type 134) - Originated by routers to announce their existence
- **Neighbor Solicitation (NS)** (Type 135) - Facilitates link-layer address resolution and duplicate address detection
- **Neighbor Advertisement (NA)** (Type 136) - Response to an NS
- **Redirect** (Type 137) - Used by a router to inform a host of a better path out of the link

Address Autoconfiguration

On broadcast links, the interface ID (the second half of an IPv6 address) can be automatically generated by converting a 48-bit MAC address to a 64-bit EUI-64 address.

MAC-to-EUI64 conversion:

1. `0xFFFE` is inserted between the two 24-bit halves of the MAC
2. The *Universal/Local bit* (7th bit) is flipped from 0 to 1

For example, `0000:0A0B:1234` becomes `0200:0AFF:FE0B:1234`.

An EUI-64 identifier can be joined with a link-local prefix (`FE80::/10`) to form a complete link-local

address.

A host can receive a global IPv6 address using either *stateful* or *stateles* autoconfiguration.

Stateful autoconfiguration uses DHCPv6 to request an IPv6 address from a server.

In stateless autoconfiguration, a host simply adds its interface ID to a prefix received in a router advertisement (RA).

Duplicate Address Detection (DAD)

DAD is performed on initial configuration of all addresses except anycasts.

New addresses are marked as tentative and cannot be used until they have been verified.

Neighbor Address Resolution

Layer 2 information for IPv6 neighbors is stored in the *neighbor cache* (similar to the ARP cache for IPv4).

Privacy Addresses

RFC 3041 defines IPv6 *privacy addresses* to alleviate privacy concerns over using a static, globally unique identifier (MAC address) as the interface ID.

Privacy addresses use a randomly-generated interface ID, which changes on a regular basis and/or when a new prefix is received.

Chapter 3: Static Routing

A routing table can be populated in three ways:

- Subnets gleaned from directly connected networks
- Manual configuration (static routes)
- Automatically via one or more dynamic routing protocols

A route's next hop must be reachable for the route to take effect.

Configuring Static Routes

IPv4:

```
Router(config)# ip route <destination> <mask> [<interface>] [<next hop>]
```

```
Router(config)# ip route 172.16.0.0 255.255.0.0 192.168.1.1
Router(config)# ip route 10.0.0.0 255.0.0.0 Serial0/0 192.168.2.2
```

Specifying only an outbound interface rather than a next-hop address assumes that the destination network is directly connected to that interface.

IPv6:

```
Router(config)# ipv6 route <destination>/<masklen> [<interface>] [<next hop>]
Router(config)# ipv6 route fec0::8:0:0:0:0/64 fec0::1:204:c1ff:fe50:f1c0
Router(config)# ipv6 route fec0::4:0:0:0:0/64 Serial0/0 fec0::2:2b0:ca4ff:fe73:459
```

IPv6 requires the next-hop address to be specified when configuring a route destined out a broadcast (Ethernet) interface.

Advanced Static Routing

Floating Static Routes

A *floating static route* is one configured with a higher administrative distance. It will only be used if more preferable routes for a destination fail.

Load Sharing

Multiple static routes can be configured to support equal-cost load sharing.

By default, *Cisco Express Forwarding (CEF)* performs load balancing per source-destination pair; all packets from one source to one destination will traverse one interface.

CEF also supports per-packet load balancing for IPv4 traffic.

The CEF load-balancing method can be adjusted:

```
Router(config)# ip load-sharing {per-destination | per-packet}
```

Recursive Lookups

A *recursive lookup* occurs when a route points to a network not directly connected; one or more subsequent lookups are required to determine the next hop.

Troubleshooting Static Routes

Remember to verify both directions of traffic flow when tracing a path.

When a router or interface hardware is replaced, a new EUI-64 identifier will be used; this may require redefining a static route.

Chapter 4: Dynamic Routing Protocols

Distance Vector Routing Protocols

The term *distance vector* is derived from a list (vector) of distances and directions to destinations.

Distance vector protocols include:

- Routing Information Protocol (RIP)
- Xerox Networking System (XNS) RIP
- Novell IPX RIP
- Cisco Interior Gateway Routing Protocol (IGRP)
- Cisco Enhanced IGRP (EIGRP)
- DEC DNA Phase IV
- Appletalk Routing Table Maintenance Protocol (RTMP)

Common distance vector characteristics:

- Periodic updates
- Reliance on neigbhors to propagate advertisements
- Broadcast updates
- Full routing table updates (advertising the entire table every time)

The per-hop nature of distance vector advertisements is known as *routing by rumor*.

Route invalidation timers control how long a route will remain in the routing table without being confirmed by a neighbor.

Split horizon prevents routing loops by preventing the readvertisement of a route to the neighbor from which it was learned. Router A will not advertise routes learned from router B back to router B.

Poison reverse extends the concept of spit horizon by readvertising a learned route back to the neighbor with an infinite metric. Router A will advertise routes learned from router B back to router B with an infinite metric, ensuring router B knows said routes are not reachable via router A.

Holddown timers place a restriction on how often a route may be updated in the table.

Link State Routing Protocols

Link state routers all share the same complete view of the network.

Link state protocol include:

- Open Shortest Path First (OSPF)
- ISO Intermediate System to Intermediate System (IS-IS)
- DEC DNA Phase V
- Novell NetWare Link Services Protocol (NLSP)

Neighbors synchronize their databases upon forming an adjacency. Hello packets are used to form and maintain adjacencies.

Link state protocols converge faster than distance vector protocols because routes can be flooded to neighbors without having to run the routing algorithm.

Sequence numbers are used to identify the revision of an advertisement.

Advertisements are aged and will eventually expire from the database if they are not refreshed periodically.

Networks are commonly divided into link state areas to reduce demand on CPU, memory, and bandwidth required to maintain the database.

Interior and Exterior Gateway Protocols

An *autonomous system (AS)* is a logical network under a common administration.

Interior Gateway Protocols (IGPs) run within an autonomous system, while *Exterior Gateway Protocols (EGPs)* run between autonomous systems.

Chapter 5: Routing Information Protocol (RIP)

There are two versions of RIP:

- **RIPv1** - Classful
- **RIPv2** (or RIPng) - Classless

RIPv1 is defined in **RFC 1058**, and operates on UDP port 520.

RIP uses only hop count as its metric, with a maximum of 15 (a metric of 16 indicates unreachability).

Upon initialization, RIP routers issue *requests* for routes from neighbors. Neighbors issue *responses*

containing their full tables.

Routers broadcast their entire table to the link-local broadcast address of `255.255.255.255` every 30 seconds (on average; a small jitter is included to prevent simultaneous flooding).

Timers

Update (30 seconds) - How often routes are advertised (+/- a small random delay)

Invalid (180 seconds) - How long a received route will stay in the table without being received again, before being marked as invalid.

Flush (240 seconds) - 60 seconds longer than the invalid timer; invalid routes will be flushed from the table when this timer is reached.

Holddown (180 seconds) - Routes will be kept in the table for this time before being replaced by an advertisement with a higher metric.

Timer configuration:

```
Router(config-router)# timers basic <update> <invalid> <holddown> <flush>
```

Header Format

Command (8 bits) - 1 for requests, 2 for responses

Version (8 bits)

The header is followed by 1-25 route entries, each consisting of an address family identifier (set to 2 for IP), network address, and metric.

Configuration

Network configuration:

```
Router(config)# router rip
Router(config-router)# network <IP>
Router(config-router)# network ...
```

Designating passive interfaces:

```
Router(config-router)# passive-interface Serial0/0
```

Specifying neighbors to which advertisements should be sent as unicasts:

```
Router(config-router)# neighbor <IP>
```

An *offset list* can be implemented to artificially increase the metric for certain routes:

```
Router(config-router)# offset-list <ACL> {in | out} <offset> <interface>
```

Triggered extensions (defined in **RFC 2091**) can be enabled per interface to eliminate periodic updates:

```
Router(config-if)# ip rip triggered
```

Chapter 6: RIPv2, RIPng, and Classless Routing

RIPv2

RIPv2 (defined in **RFC 1723**) expands on its predecessor to support:

- Classless routing
- Authentication
- Next hop addresses
- External route tags
- Multicast advertisements (to 224.0.0.9) instead of broadcasts

RIPv2 uses space that was unused in the RIPv1 header to embed a 16-bit route tag in the header, and subnet mask and next hop information in each route entry.

RIPv2 can be run in *compatibility mode*, broadcasting advertisements to ensure backward compatibility with RIPv1.

Authentication can be implemented by inserting a plaintext password or MD5 hash before the first route entry in every advertisement.

RIPng

Next-Generation RIP (RIPng) is an entirely new protocol designed for IPv6; it does not support IPv4.

RIPng uses the same timers, procedures, message types, and metric as RIPv2.

A next hop address is included in a single designated route entry, and all entries which follow it are

assumed reachable by that address.

Configuring RIPv2

Base configuration:

```
Router(config)# router rip
Router(config-router)# version 2
Router(config-router)# network <IP>
Router(config-router)# network ...
```

By default, RIPv2 summarizes on classful boundaries. This can be disabled:

```
Router(config-router)# no auto-summary
```

The version of RIP advertisements sent and listened for can be configured per interface:

```
Router(config-if)# ip rip send version {1 | 2}
Router(config-if)# ip rip receive version {1 | 2}
```

Key chains can be used to implement authentication per interface:

```
Router(config-if)# ip rip authentication key-chain <name>
Router(config-if)# ip rip authentication mode md5
```

Configuring RIPng

IPv6 routing is required to support RIPng:

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

RIPng is enabled per interface rather than in global config:

```
Router(config-if)# ipv6 rip <process> enable
```

Details can be modified per RIPng process:

```
Router(config)# ipv6 router rip <process>
```

```
Router(config-router)# port <UDP port> multicast ff02::9
Router(config-router)# distance <admin distance>
```

RIPng can be configured to artificially increment the metric per interface:

```
Router(config-if)# ipv6 rip <process> metric-offset <count>
```

Enabling summarization:

```
Router(config-if)# ipv6 rip <process> summary-address <network>/<length>
```

Chapter 7: Enhanced Interior Gateway Routing Protocol (EIGRP)

EIGRP is an advanced version of Cisco's proprietary IGRP.

EIGRP runs the *Diffusing Update Algorithm (DUAL)* instead of Bellman-Ford to attain fast convergence while remaining loop-free.

Protocol-dependent modules are used to support protocols other than IP (such as IPX and AppleTalk).

Reliable Transport Protocol (RTP) provides ordered delivery of packets between EIGRP neighbors.

EIGRP can consider bandwidth, delay, reliability, and load in calculating a metric; only bandwidth and delay are considered by default.

EIGRP packet types:

Hello - Peer discovery and maintenance

Acknowledgment - Empty hello packets used to acknowledge messages

Update - Convey route information

Query - Request for a route

Reply - Answer to a query

The default hello interval is 5 seconds on a fast link or 60 seconds on a slow (<= T1) link; this can be adjusted with `ip hello-interval eigrp <seconds>` at interface configuration.

Each hello packet includes a *hold time* (three times the hello interval by deault); this an be adjusted with `ip hold-time eigrp <seconds>` at interface configuration.

DUAL

Of all routes to a destination, the one with the lowest metric will be designated the *successor* route.

The *feasibility condition* states that a route's advertised distance must be less than the router's feasible distance to a destination for it to be considered a *feasible successor*, to avoid creating a routing loop.

Traffic will be load-balanced across multiple paths with the same feasible distance.

DUAL Finite State Machine

Routes remain in the *passive state* while no DUAL calculations are being performed.

An *input event* such as an interface state transition or the reception of an update, query, or reply packet, will cause the router to recalculate the distance for all its feasible successors for the affected route.

If a feasible successor cannot be found, the route enters the *active state*, and queries for a new path are issued to EIGRP neighbors.

If a router does not receive responses from all neighbors to which it issued queries within the active timer (3 minutes by default), the route is considered *stuck in active*, and unresponsive neighbors are removed from the neighbor table.

If all replies are received, the router calculates the feasible distance for all advertised routes, and adds the lowest valid route (if any) as the new successor.

Packet Header

EIGRP is IP protocol 88.

Header format:

- **Version** (8 bits)
- **Opcode** (8 bits) - Packet type (hello, update, etc.)
- **Checksum** (16 bits) - Error detection for the packet
- **Flags** (32 bits)
- **Sequence** (32 bits) - Sequence number used by RTP
- **ACK** (32 bits) - Last sequence number from neighbor
- **Autonomous System (AS) Number** (32 bits)

Type/Length/Value (TLV) triplets follow the header to provide routes and other information.

Configuring EIGRP

Base configuration:

```
Router(config)# router eigrp <AS>
Router(config-router)# network <IP> [<wildcard mask>]
```

EIGRP will automatically load-balance across up to 16 equal-cost links (4 by default). Unequal-cost balancing can be enabled by specifying a multiplier by which metrics may differ:

```
Router(config-router)# variance <variance>
```

The maximum number of paths used in load balacing can be configured:

```
Router(config-router)# maximum-paths <count>
```

To designate a passive interface:

```
Router(config-router)# passive-interface <interface>
```

EIGRP summarizes at classful boundaries by default. To disable automatic summarization:

```
Router(config-router)# no auto-summary
```

Routers can be configured as stubs to limit the types of routes advertised:

```
Router(config-router)# eigrp stub {connected | redistributed | static | summary |
 receive-only}
```

Neighbors do not send queries to stub routers.

Summarization boundaries can be implemented at interfaces:

```
Router(config-if)# ip summary-address eigrp <AS> <network> <mask>
```

EIGRP only supports MD5 authentication, which is enabled per interface:

```
Router(config-if)# ip authentication key-chain eigrp <AS> <keychain>
Router(config-if)# ip authentication mode eigrp <AS> md5
```

Chapter 8: OSPFv2

Open Shortest Path First (OSPF) version 2 is defined in **RFC 2328**.

Link State Advertisements (LSAs) are flooded to all routers within an area, to ensure all routers have a complete database of the network within the area. This database is used by each router to construct an independent shortest path tree.

Adjacencies

Cisco OSPF routers will obtain a router ID from one of three sources, in order:

1. An administratively configured value (the `router-id` command)
2. The IP of the highest loopback interface
3. The IP of the highest other interface

Hello packets are used to establish and maintain neighbor relationships.

Network Types

OSPF supports five network types:

Point-to-point - Connects exactly two routers

Broadcast - Multiaccess network with broadcast capability

Nonbroadcast Multiaccess (NBMA) - Multiaccess network without broadcast capability

Pont-to-multipoint - An NBMA treated as a collection of point-to-point links

Virtual link - A virtual connection designed to bridge areas

Designated Routers

A *Designated Router (DR)* and *Backup Designated Router (BDR)* are elected on multiaccess links to limit the number of adjacencies formed.

Each OSPF interface is assigned a priority between 0 and 255 (1 is default).

The DR and BDR are elected based on highest priority, or by highest router ID in the event of a tie.

Once a DR and BDR are elected, they may not be preempted by a new router with a higher priority.

The DR and BDR multicast LSAs to the *AllSPFRouters* address (224.0.0.5), while all other routers multicast LSAs to *AllDRouters* (224.0.0.6).

OSPF Interfaces

Detailed interface characteristics can be viewed with `show ip ospf interface`.

Interface states:

Down

Point-to-point - Applicable only to point-to-point links

Waiting - DR/BDR election occurring

DR

BDR

DRother

Loopback - Interface is looped (unusable), but still advertised in LSAs

Neighbors

Neighbor states:

Down

Attempt (NBMA only) - Hellos are sent to a neighbor at the hello interval when an interface transitions to active

Init - A hello has been received from the neighbor but did not list the local router's ID

2-Way - A hello containing the local router's ID was received from the neighbor; bidirectional communication has been established

ExStart - A master/slave relationship is formed to facilitate exchange of database descriptions

Exchange - Routers exchange database description packets summarizing the entire contents of their link state databases

Loading - Routers exchange LSRs and LSUs to obtain new LSAs

Full

LSAs exchanged can be acknowledged one of two ways:

Explicitly - Replying with an LSAck containing the header of the LSA received

Implicitly - Sending an update with the same instance of the LSA received

Flooding

Two OSPF packet types are used to flood LSAs throughout an area:

Link State Update (LSU) - Type 4

Link State Acknowledgment (LSAck) - Type 5

All LSAs are acknowledged to ensure reliable flooding.

Each LSA has a 16-bit sequence number and a 16-bit age (in seconds).

Areas

An OSPF domain is segmented into areas to alleviate the demands of maintaining an entire link state database.

Area 0 is the *backbone*; all other areas must connect through it.

There are four nonexclusive router types:

Internal Router - All interfaces belong to a single area

Area Border Router (ABR) - Connects one or more areas to the backbone

Backbone Router - At least one interface is in the backbone area

Autonomous System Boundary Router (ASBR) - Participates in one or more additional routing domains; can be located anywhere in the OSPF domain

A *virtual link* can be configured to provide a virtual path to the backbone. Virtual links must be configured between ABRs and cannot traverse a stub area. At least one end of the virtual link must reside in the backbone area.

Link State Database

All OSPF routers in an area must have the same link state database to construct accurate shortest path trees.

All LSAs are refreshed every *LSRefreshTime* (default 30 minutes). LSAs older than *MaxAge* (default 60 minutes) are deleted.

The LSRefreshTime is staggered for efficiency, so that LSAs are flooded in groups at pseudorandom intervals; this is known as *group pacing*.

LSA Types

1. Router LSA
2. Network LSA
3. Network Summary LSA
4. ASBR Summary LSA
5. AS External LSA
6. Group Membership LSA (unused)
7. NSSA External LSA
8. External Attributes LSA (unused)
9. Opaque LSA (link-local scope)
0. Opaque LSA (area-local scope)
1. Opaque LSA (AS scope)

Router LSAs (type 1) are produced by all OSPF routers, and identify all of a router's links.

Network LSAs (type 2) are produced by the DR on a multiaccess segment to represent the pseudonode.

Network Summary LSAs (type 3) are originated by ABRs to advertise destinations from one are to another.

ASBR Summary LSAs (type 4) are similar to network summary LSAs, but instead advertise the location of an ASBR.

AS External LSAs (type 5) are originated by an ASBR to represent an external route, and flooded throughout the OSPF domain.

Group Membership LSAs (type 6) are used by Multicast OSPF (unsupported by Cisco IOS).

NSSA External LSAs (type 7) are essentially AS External LSAs originated by an ASBR in a not-so-stubby area; type 7 LSAs are only flooded within the area.

External Attributes LSAs (type 8) were proposed as an alternative to iBGP but are not implemented in Cisco IOS.

Opaque LSAs (types 9 through 11) can provide transport for application-specific information (for example, to support MPLS traffic engineering).

Area Types

Stub areas do not receive type 4 or 5 LSAs. Instead, ABRs for a stub area advertise a default route out

of the area.

A *totally stubby area* expands on the stub area concept to also block type 3 LSAs. Routers in a totally stubby area receive only a default route from ABRs.

A *not-so-stubby area (NSSA)* is a stub area containing one or more ASBRs. These ASBRs advertise type 7 LSAs representing external routes.

Type 7 LSAs are converted to type 5 LSAs by the ABR if they are to be advertised out of the area.

Route Table

Each OSPF route is one of four types (listed in order of preference):

Intra-area - Within the local area

Inter-area - To another area

External Type 1 (E1) - A route outside the AS which factors external cost plus internal cost to the ASBR

External Type 2 (E2) - A route outside the AS which only includes external cost

External routes are type 2 by default.

OSPF over Demand Circuits

Periodic LSA refreshes can be disabled for on-demand circuits. This will ensure the link is only brought up during the initial configuration or when it is otherwise needed.

Hellos are not transmitted out interfaces designated as on-demand, except for multiaccess network types (the hellos are needed to facilitate the DR/BDR election process).

LSAs transmitted over a demand circuit have their *DemandCircuit* and *DoNotAge* bits set to 1, disabling the effect of the MaxAge timer.

Demand interfaces are designated with `ip ospf demand-circuit` at interface configuration.

Periodic flooding can also be disabled for regular (non-on-demand) circuits with `ip ospf flood-reduction`.

Configuring OSPF

Base configuration:

```
Router(config)# router ospf <process>
```

```
Router(config-router)# network <network> <mask> area <area>
```

OSPF can be configured to perform reverse DNS lookups on router IDs (assuming at least one DNS server has been configured):

```
Router(config)# ip ospf name-lookup
```

OSPF views secondary addresses as stub networks; the primary address on an interface must participate in the OSPF process for any of its secondary addresses to be included.

All routers in a stub area must be configured to recognize the area as a stub:

```
Router(config-router)# area <area> stub
```

The cost of a default route injected into a stub area by an ABR is 1 by default. This can be adjusted:

```
Router(config-router)# area <area> default-cost <cost>
```

The designation of a totally stubby area only has to be made on its ABRs:

```
Router(config-router)# area <area> stub no-summary
```

All routers in a not-so-(totally)-stubby area (NSSA) must be configured:

```
Router(config-router)# area <area> nssa [no-summary]
```

Redistribution into an NSSA can be enabled by appending `no-redistribute` to the area statement.

Inter-area summarization:

```
Router(config-router)# area <area> range <network> <mask> [not-advertise]
```

Including the `not-advertise` command prevents routes within the specified range from being advertised.

Prefix lists can also be used to prevent certain prefixes from being advertised into or out of an area:

```
Router(config)# ip prefix-list <name> seq 10 deny <network>/<len>
Router(config)# ip prefix-list <name> seq 20 permit 0.0.0.0/0 le 32
Router(config)# router ospf <process>
Router(config-router)# area <area> filter-list prefix <name> {in | out}
```

Implementing MD5 authentication:

```
Router(config-if)# ip ospf message-digest-key <key number> md5 <string>
Router(config-if)# router ospf <process>
Router(config-router)# area <area> authentication message-digest
```

Multiple keys can be configured per interface to enable transitioning passwords without breaking the adjacency.

Configuring a virtual link through area 123 to router 4.4.4.4:

```
Router(config-router)# area 123 virtual-link 4.4.4.4
```

Virtual link operation can be verified with `show ip ospf virtual-link`.

Neighbors must be manually configured for NBMA segments:

```
Router(config-router)# neighbor <IP> [priority <0-255>]
```

The default priority for a manually defined neighbor is 0.

Alternatively, NBMA segments can be configured to act as broadcast segments:

```
Router(config-if)# ip ospf network broadcast
Router(config-if)# ip ospf priority <priority>
```

A third option is to configure the segment as point-to-multipoint, eliminating the need for a DR:

```
Router(config-if)# ip ospf network point-to-multipoint [non-broadcast]
```

The addition of the `non-broadcast` command requires manual neighbor definition.

Enabling demand circuits:

```
Router(config-if)# ip ospf demand-circuit
```

Chapter 9: OSPFv3

A successor to OSPFv2 was developed to address some areas of deprecation as well as provide IPv6 support. OSPFv3 is defined in **RFC 2740**.

Differences From OSPFv2

OSPFv3 is processed per *link* rather than per subnet, as an IPv6 interface typically has several addresses.

IP addresses are removed from router and network LSAs.

Neighbors on all link types are always identified by their router ID; never by interface address.

In addition to domain- and area-local, a link-local flooding scope is introduced, supported by a new *Link* LSA type.

OSPFv3 supports multiple instances per link by adding an *instance ID* to packet headers.

OSPF-specific authentication has been removed. Instead, authentication is provided by IPv6/Authentication Header.

OSPFv3 provides more flexible handling of unknown LSA types, with an option to ignore but flood to neighbors. This can provide greater ease in implementing future additions to the protocol.

OSPFv3 Messages

OSPFv3 uses the same five message types as OSPFv2:

- Hello
- Database Description (DD)
- Link State Request (LSR)
- Link State Update (LSU)
- Link State Acknowledgment (LSAck)

OSPFv3 hellos have a larger options field (24 bits vs. 16), a smaller dead interval field (16 bits vs. 32), and do not include a network mask.

OSPFv3 database description messages also have a larger options field.

The other three message types remain unchanged.

OSPFv3 LSA Types

The OSPFv3 LSA header provides an expanded *Link State Type* field, of which the first three bits are flags:

> **U** - Determines whether to ignore or store and flood unknown the LSA if its type is unknown to the router
>
> **S1** and **S2** - A two-bit value indicating the scope of the LSA (link-local, area, domain, or *reserved*)

LSA types:

> **0x2001** - Router LSA (v2 Router LSA)
>
> **0x2002** - Network LSA (v2 Network LSA)
>
> **0x2003** - Inter-area Prefix LSA (v2 Network Summary LSA)
>
> **0x2004** - Inter-area Router LSA (v2 ASBR LSA)
>
> **0x4005** - AS-External LSA (v2 AS-External LSA)
>
> **0x2006** - Group Membership LSA (v2 Group Membership LSA)
>
> **0x2007** - Type-7 LSA (v2 NSSA External LSA)
>
> **0x0008** - Link LSA
>
> **0x2009** - Intra-area prefix LSA

OSPFv3 Router and Network LSAs do not carry address information. Instead, addressing is decoupled from the SPF tree structure and carried in Intra-Area Prefix LSAs.

OSPFv3 also introduces a Link LSA for advertising information which is only pertinent to directly connected neighbors.

Configuring OSPFv3

Configuration of OSPFv3 is largely similar to that of OSPFv2, merely substituting the `ipv6` command for `ip` and using the relevant address formats.

One major difference is that areas and other interface-specific parameters are defined on the interface for OSPFv3, rather than with the `network` command under the OSPF process:

```
Router(config-if)# ipv6 address 2001:abcd:0:8::1/64
Router(config-if)# ipv6 ospf <process> area <area>
```

All IPv6 addresses configured on an interface are included in OSPFv3.

Because hello packets are sourced from a router's link-local interface address, an adjacency will be formed even if the neighbor is configured in a different prefix.

Multiple instances of OSPFv3 can be configured per-link to limit adjacencies:

```
Router(config-if)# ipv6 ospf <process> area <area ID> instance <instance ID>
```

Chapter 10: Integrated IS-IS

Intermediate System-to-Intermediate System (IS-IS) was developed by ISO as the routing protocol for its *Connectionless Network Protocol (CLNP)*, part of a protocol suite originally intended to replace TCP/IP. IS-IS is defined in ISO 10589.

Integrated IS-IS refers to an IS-IS extension to support IP routing.

Integrated IS-IS Operation

Like OSPF, IS-IS is a link-state protocol which uses SPF to maintain a shortest-path tree.

Some ISO terminology:

Intermediate System (IS) - Router

End System (ES) - Host

Subnetwork Point of Attachment (SNPA) - An interface connected to a subnetwork

Protocol Data Unit (PDU) - A packet pertaining to a specific level of the OSI model (data link PDU, link state PDU, etc.)

OSPF area borders fall on routers; IS-IS area borders fall on links.

An IS-IS network has two levels:

Level 1 contains non-backbone routers

Level 2 contains backbone routers

Routers can be L1, L2, or L1/L2; L1/L2 is the default for Cisco IOS.

An L1 area is similar to an OSPF totally stubby area, as L1/L2 do not advertise L2 routes to L1 neighbors by default.

A router can have up to three area addresses by default (configurable with `max-area-addresses`). An area address is combined with the system ID to form the *Network Entity Title (NET)*.

NET

The area ID is a variable length field which may encompass additional fields for interdomain routing.

The system ID is always 6 bytes, often taken from a MAC address.

The *NSAP Selector (SEL)* is the last byte of the NET and is always set to `0x00`.

IS-IS Functional Organization

The network layer of the OSI model is split into two sublayers:

> **Subnetwork Independent** - Provides consistent network services to the transport layer
>
> **Subnetwork Dependent** - Accesses the services of the data link layer on behalf of the subnetwork independent sublayer

Subnetwork Dependent Functions

Subnetwork dependent functions include:

> Transmission and reception of PDUs
>
> Neighbor discovery
>
> Adjacency formation and maintenance
>
> Link demultiplexing (separating OSI and IP PDUs)

IS-IS has only two network types: broadcast and point-to-point.

L1 adjacencies will only form between routers in the same area; L2 adjacencies will form without regard to area ID.

A router on a broadcast network will form adjacencies with all neighbors, not just the DR. The DR is chosen as the router with the highest interface priority (or in a tie, the highest MAC address).

The IS-IS DR election is simpler than in OSPF, and differs in two ways: no BDR is elected, and a new router with a higher priority than the current DR can preempt it to become the new DR.

Subnetwork Independent Functions

The IS-IS routing function is divided into four processes: update, decision, forwarding, and receive. The forwarding and receive processes deal primarily with CLNS NPDUs.

Update Process

The update process constructs the L1 and L2 link-state databases.

L1 PDU frames on a broadcast network are multicast to `0180.c200.0014` (*AllL1ISs*); L2 PDU frames are multicast to `0180.c200.0015` (*AllL2ISs*).

Partial or *Complete* SNPs are used to request and acknowledge LSPs.

Routers can set the *overload (OL bit)* in LSPs to inform neighbors they are not able to handle transit traffic. This can be convenient for imposing a delay to let iBGP converge, for example. The OL bit can be set for a period of time after startup with `set-overload-bit on-startup`.

Decision Process

The decision process uses the link-state database to construct a shortest-path tree and forwarding database.

Cisco IOS assigns a metric of `10` to every interface by default; this can be configured independently for levels 1 and 2.

IS-IS originally only allowed for a 10-bit metric; `metric-style-wide` is used on IOS to expand this to 32 bits.

Routes can be *internal* or *external*; external routes are typically only found at L2.

L1 paths are preferred over L2 paths.

The `which-route` command can be used to conveniently correlate CLNS and IP addresses.

PDU Formats

IS-IS uses 7 PDU types:

Hello PDUs

- Level 1 LAN Hello (type 15)
- Level 2 LAN Hello (type 16)
- Point-to-point Hello (type 17)

Link State PDUs

- Level 1 LSP (type 18)
- Level 2 LSP (type 20)

Sequence Number PDUs

Level 1 CSNP (type 24)

Level 2 CSNP (type 25)

Level 1 PSNP (type 26)

Level 2 PSNP (type 27)

The PDU header is 8 bytes long, and contains the following fields (8 bits each):

Interdomain Routing Protocol Discriminator - Set to `0x83`

Length Indicator - Header length

Version/Protocol ID Extension - Set to `0x01`

ID Length - Length of the system ID

PDU type - Lower five bits of the octet, identifies the PDU type

Version - Set to `0x01`

Reserved

Maximum Area Addresses - Maximum number of addresses assigned to this area (default 3)

The PDU header is follow by PDU-specific fields and an optional number of *Type/Length/Value (TLV)* fields.

Extensions to IS-IS

Three-Way Handshaking

IS-IS always uses three-way handshaking on broadcast links, but **RFC 3373** was introduced to apply it to point-to-point links as well. A new PDU TLV was introduced to list known neighbors.

Domain-Wide Prefix Distribution

By default, L1 areas are like OSPF totally stubby areas, relying on a default route to reach L2.

Advertising L2 routes into L1 is known as *route leaking*. Route leaking can be dangerous because L1/L2 routers must not re-advertise L2->L1 routes back into L2.

RFC 2666 introduces the *Up/Down* bit in the IP Reachability TLVs, allowing such routes to be marked and preventing advertisement back into L2.

Wide Metrics

RFC 3784 introduces the *Extended IS Reachability* (type 22) and *Extended IP Reachability* (type 135)

TLVs.

These new TLVs expand the original 6-bit metric to 24 bits and 32 bits, respectively.

Wide metrics are enabled in IOS with `metric-style wide`.

Routing IPv6 with IS-IS

The *IPv6 Reachability* (type 236) and *IPv6 Interface Address* (type 232) TLVs introduce support for IPv6 routing.

Dynamic Host Exchange

RFC 2763 introduces the *Hostname TLV* (type 137) to include a human-friendly hostname for the advertising router. This hostname will appear, for example, in the output of `show clns is-neighbors`.

Multiple Topologies

Multi Topology (MT) routing allows for the specification of multiple topologies based on protocol (IPv4, IPv6, multicast).

MT IDs are assigned to an interface to indicate which topologies it belongs to. Separate route tables and SPF trees are kept for each topology.

Mesh Groups

RFC 2973 introduces mesh groups to prevent the unnecessary flooding of LSPs on mesh networks.

A mesh group-capable router has its interfaces in one of three mesh group states:

Inactive - Not participating in a mesh group

Blocked - Does not flood LSPs

Set - Assigned to a subgroup

Subgroups can be defined to divide mesh networks and introduce more redundancy than blocked links provide.

Improving SPF Efficiency

Incremental SPF (iSPF) recognizes changes which do not require the SPF tree to be rebuilt to conserve CPU utilization. iSPF can also confine the recalculation of the SPF tree to a particular area.

A *Partial Route Calculation (PRC)* occurs when a new IP prefix is flooded but does not necessitate rebuilding the SPF tree.

IOS uses an exponential back-off algorithm (configurable with `spf-interval`) to limit how often the SPF tree is rebuilt.

Configuring Integrated IS-IS

Basic configuration:

```
Router(config)# router isis
Router(config-router)# net <NET>
...
Router(config-if)# ip router isis
```

Changing the level of a router:

```
Router(config-router)# is-type {level-1 | level-2-only | level-1-2}
```

Configuring route summarization:

```
Router(config-router)# summary-address <network> <mask>
```

IS-IS supports three levels of authentication: neighbor, area-wide, and domain-wide.

Neighbor authentication:

```
Router(config-if)# isis authentication mode {text | md5} [level-1 | level-2]
Router(config-if)# isis authentication key-chain <key-chain> [level-1 | level-2]
```

Area authentication is configured similarly, but with L1 `authentication` commands configured under the IS-IS process. Domain authentication is configured as L2.

If IPv6 is configured, *all* IS-IS routers in the domain must support IPv6 (unless multiple topologies are configured).

Route leaking can be configured by redistributing L2 routes into L1:

```
Router(config-router)# redistribute isis ip level-2 into level-1 distribute-list <ACL>
```

Leaked routes are marked as `ia` (inter-area) in the routing table instead of L1 or L2.

Multiple L1 areas can be configured on a single router by configuring multiple IS-IS processes:

```
Router(config)# router isis Foo
Router(config-router)# net 49.0001.1111.1111.1111.00  ! Area 1
Router(config-router)# router isis Bar
Router(config-router)# net 49.0002.1111.1111.1111.00  ! Area 2
Router(config-router)# interface f0/0
Router(config-if)# ip routers isis Foo
Router(config-router)# interface f0/1
Router(config-if)# ip routers isis Bar
```

Chapter 11: Route Redistribution

Running multiple routing protocols or processes in conjunction without redistribution between them is often referred to as *ships in the night* routing.

Care must be taken when redistributing between protocols with unalike metrics and unequal administrative distances.

Redistribution from classless to classful routing protocols only works if the classful router has an interface in the subnet(s) being advertised, or if the subnet falls on a classful boundary.

Configuring Redistribution

Redistribution into a routing process is enabled with `redistribute` under the process configuration, optionally followed by a metric specification.

OSPF into EIGRP example:

```
Router(config)# router eigrp 1
Router(config-router)# redistribute ospf 1 metric 10000 100 255 1 2500
```

EIGRP into OSPF example:

```
Router(config)# router ospf 1
Router(config-router)# redistribute eigrp 1 metric 30 subnets
```

The metric for redistributed routes can alternatively be configured with `default-metric`:

```
Router(config-router)# default-metric 30
```

The `metric` keyword of `redistribute` takes precedence over `default-metric` if both are configured.

OSPF summarization is configured with `summary-address` (or `summary-prefix` for OSPFv3) under the OSPF process:

```
Router(config-router)# summary-address 192.168.0.0 255.255.0.0
```

IS-IS also uses the `summary-address` or `summary-prefix` command to summarize redistributed routes.

EIGRP summarization is configured at the interface:

```
Router(config-if)# ip summary-address eigrp 1 192.168.0.0 255.255.0.0
```

A router will point the summary route to interface Null0; this ensures a routing loop isn't created if the summary includes nonexistent subnets.

Chapter 12: Default Routes and On-Demand Routing

Default routes are beneficial in that they reduce resource utilization, however they sacrifice detail and granularity to achieve this.

On-Demand Routing (ODR) provides the ability to automatically discover stub networks while the stub routers rely only on a default route.

ODR is not a true routing protocol; it relies on *Cisco Discovery Protocol (CDP)* to discover stub networks, and is consequently limited to one hop.

ODR-learned routes have an administrative distance of 160, and their metric is always 1.

Configuration

Configuring static default routes for IPv4 and IPv6:

```
Router(config)# ip route 0.0.0.0 0.0.0.0 192.168.0.1
Router(config)# ipv6 rotue ::/0 2001:abcd:0:10::1
```

Static default routes can be redistributed into some routing protocols using `redistribute static`. OSPF and IS-IS use the `default-information originate` command instead.

ODR is enabled with `router odr`; it has no other parameters.

ODR routes can be redistributed into another protocol with `redistribute odr`:

```
Router(config-router)# redistribute odr metric 100
```

Chapter 13: Route Filtering

Route feedback occurs when an advertised route is redistributed back into its protocol of origin.

Route filtering is used to control the propagation of routes between routers.

Route filters can be applied inbound to limit the routes received or outbound to control the routes being advertised.

Configuration

Applying a distribute list to filter inbound routes:

```
Router(config)# access-list 1 permit 192.168.0.0 0.0.255.255
...
Router(config-router)# distribute-list 1 in [<interface>]
```

Distribute lists can reference an ACL or a *prefix list*.

Defining a prefix list:

```
Router(config)# ip prefix-list <number> [permit | deny] <subnet>/<length>
 [{le | ge} <length>]
```

Applying a distribute list outbound for a link-state routing protocol has no effect, as link-state protocols like OSPF and IS-IS advertise link-state information, not routes.

Using a distribute list to control routes redistributed to another protocol:

```
Router(config-router)# distribute-list <number> out <protocol>
```

Altering the administrative distance of a routing protocol:

```
Router(config-router)# distance <distance [...]> [<network>]
```

Chapter 14: Route Maps

Route maps provide a greater level of flexibility than access lists for redistribution and policy routing.

Characteristics that can be matched by a redistribution route map include:

- Outbound interface
- Destination address
- Next hop
- Route source
- Route metric
- Route type
- Route tag

Properties which can be modified by a redistribution route map:

- IS-IS level or OSPF area type
- Route metric
- Route metric type
- Next hop
- Tag

Policy routing matches:

- ACL match
- Packet length

Policy routing sets:

- Outbound interface
- Next hop
- QoS properties

Configuration

Route map configuration:

```
Router(config)# route-map <name> permit 10
Router(config-map)# match <condition>
Router(config-map)# match <condition 2>
Router(config-map)# set <attribute>
```

```
Router(config-map)# set <attribute 2>
Router(config-map)# route-map <name> permit 20
...
```

Configuring policy routing:

```
Router(config-if)# ip policy route-map <name>
```

To apply policy routing to locally-originated traffic:

```
Router(config)# ip local policy route-map <name>
```

Using route maps for redistribution or route tagging:

```
Router(config-router)# redistribute <protocol> [...] route-map <name>
```

© 2008 PacketLife.net