Exam Manual

LearnSmart

CISCO (642-901) BSCI

Smarter Training

This LearnSmart exam manual covers the most important topics you will encounter on the Scalable Internetworks exam (BSCI). By studying this manual, you will become familiar with an array of exam-related topics, including:

- Implementing EIGRP operations
- Implementing multi-area OSPF operations
- Describing Integrated IS-IS
- And more!

Give yourself the competitive edge necessary to further your career as an IT professional and purchase this exam manual today!

Building Scalable Cisco Internetworks (BSCI) LearnSmart Exam Manual

Product ID: 011237
Production Date: July 19, 2011

Warning and Disclaimer

Every effort has been made to make this document as complete and as accurate as possible, but no warranty or fitness is implied. The publisher and authors assume no responsibility for errors or omissions. The information provided is on an "as is" basis. The authors and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this document.

Volume, Corporate, and Educational Sales

Favorable discounts are offered on all products when ordered in quantity. For more information, please contact us directly:

1-800-418-6789
solutions@learnsmartsystems.com

International Contact Information

International: +1 (813) 769-0920

United Kingdom: (0) 20 8816 8036

Table of Contents

Abstract

The Cisco Certified Network Professional is one of the most well respected certifications in the world. By attaining it, students and candidates signify themselves as extremely accomplished and capable Network Professionals. These four exams, created by Cisco Systems, are extremely difficult and not to be taken lightly. They cover a myriad of topics, in particular - the BSCI (Building Scalable Cisco Internetworks) exam covers all the way to the most detailed analysis of routing packets across multiple subnetted networks, including VLANs and other complex network concepts. BSCI is multiple choice, simulative, and incorporates test strategies such as "drag and drop" and "hot area" questions to verify a candidate's knowledge.

Before taking this exam, you should be very familiar with both Cisco technology and networking. You must have also attained the Cisco CCNA certification by passing either the one or two part path.

Your Product

This BSCI Exam Manual has been designed from the ground up with you, the student, in mind. It is lean, strong, and specifically targeted toward the candidate. Unlike many other BSCI products, the LearnSmart BSCI Exam Manual does not waste time with excessive explanations. Instead, it is packed full of valuable techniques, priceless information, and brief, but precisely worded, explanations. While we do not recommend using only this product to pass the exam, but rather a combination of LearnSmart Audio Training, Practice Exams, and Video Training, we have designed the product so that it and it alone can be used to pass the exam.

About the Author

Sean Wilkins is an accomplished networking consultant and has been in the field of IT since the mid 1990's working with companies like Cisco, Lucent, Verizon and AT&T. In addition to being a CCNP and CCDP, Sean is also a MCSE and an overall "IT expert." In addition to working as a consultant, Sean spends a lot of his time as a technical writer and editor.

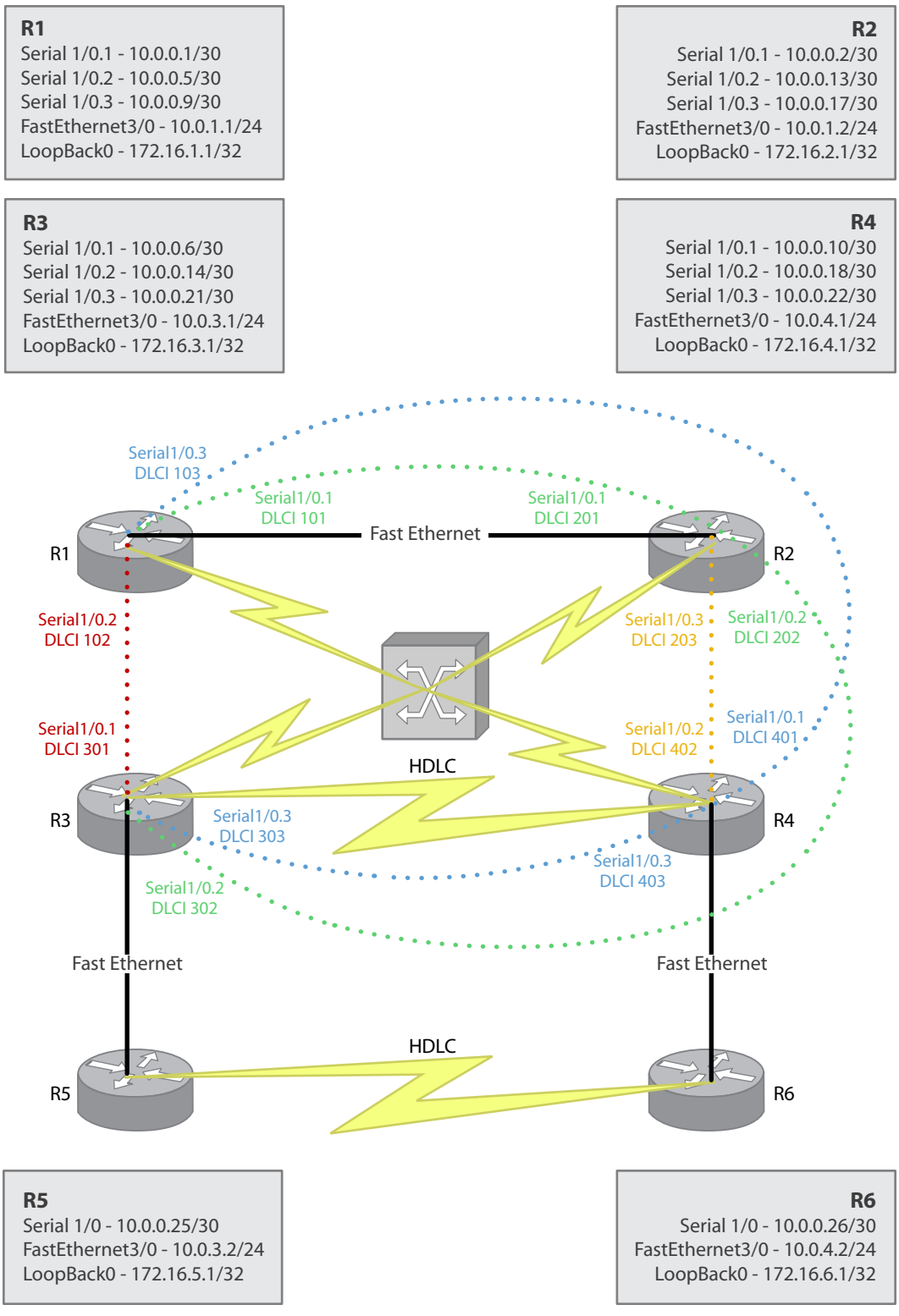


Figure 1 - Samples Lab Layout (Single Routing Protocol)

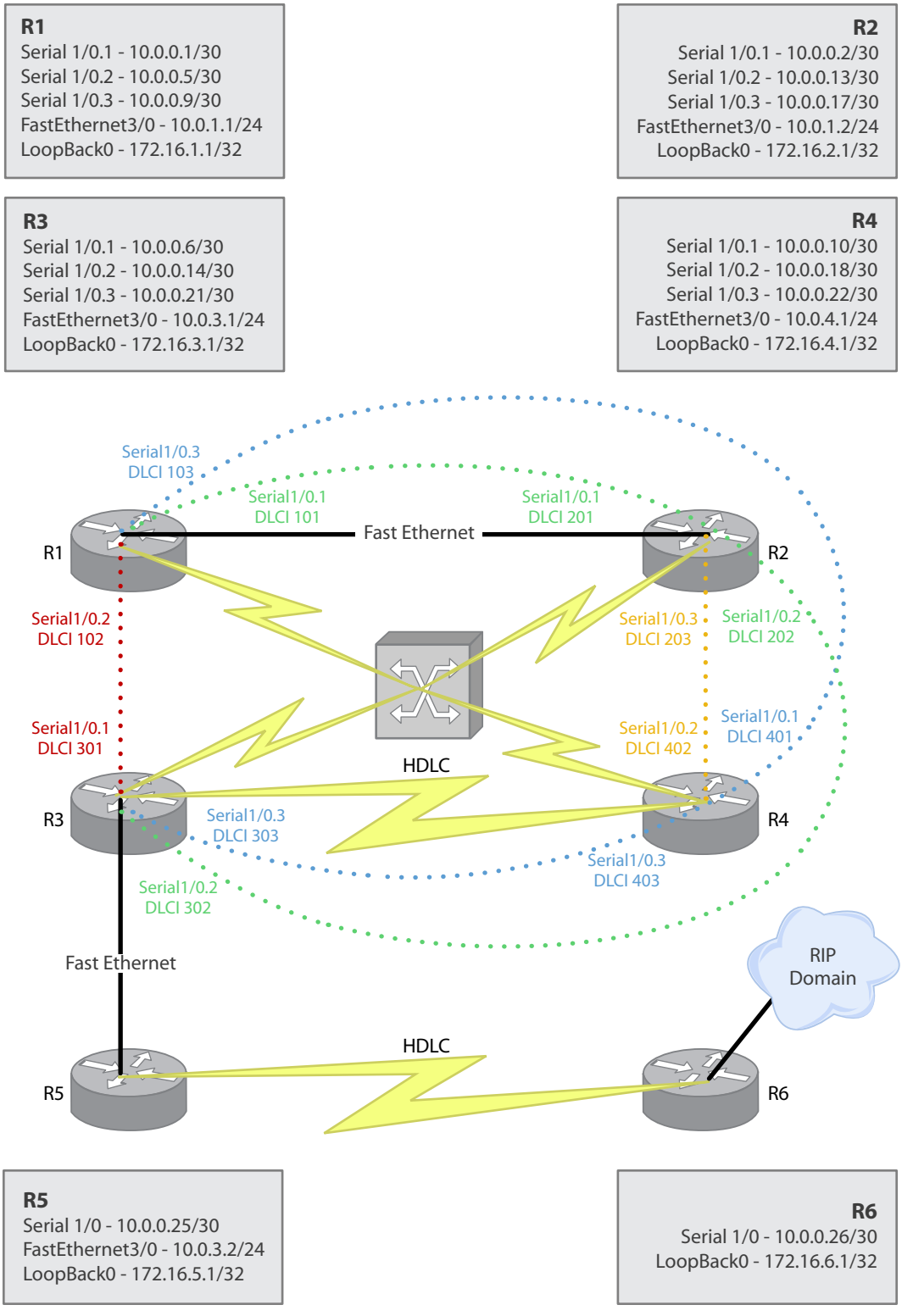


Figure 2 - Lab Layout (Multiple Routing Protocol)

Domain 1 - Implement EIGRP Operations

EIGRP Overview

EIGRP (Enhanced Interior Gateway Routing Protocol) is a Cisco proprietary routing protocol. EIGRP has advantages of both distance vector and link state routing protocols. It does this though the use of the Diffusing Update Algorithm (DUAL). EIGRP has very low network overhead. On startup EIGRP uses *hello* packets to communicate with neighbors and to exchange full topological information. This is the only time EIGRP exchanges full topology tables. All EIGRP updates are only sent when there is a network change and only the changes are sent over the network, which additionally reduces the overhead. EIGRP keeps two different routing tables. One, the *routing table* holds the current primary routing information to all known networks, and two, the *topology table* holds the information in the routing table and information related to all potential backup routes. This enables almost instantaneous switchover upon failure should multiple paths exist. EIGRP also provides classful boundary summarization by default.

Another main advantage of EIGRP is that its design works around protocol dependent modules, which enables it to be used with a number of different protocols. In the past this was good because it could be used for not only IPv4 but also IPX (Novell) and AppleTalk (Apple). In today's world these last two protocols are rather obsolete. Another protocol that can utilize EIGRP isIPv6. EIGRP also utilizes Reliable Transport Protocol (RTP) for communications to make sure that all communications are reliable. This traffic is transported via IP protocol 88. Protocol 88 is reserved exclusively for EIGRP traffic.

Within the Cisco world, internal EIGRP routes have an administrative distance of 90, while external EIGRP routes have an administrative distance of 170 and summary EIGRP routes have an administrative distance of 5.

EIGRP Routing and DUAL Basics

Within EIGRP there are a number of terms that must be explained:

Advertised Distance (reported distance)	The Advertised Distance (AD) is the distance from a given neighbor to the destination router.
Feasible Distance	The Feasible Distance (FD) is the distance from the current router to a destination network.
Feasibility Requirement	Within EIGRP there is a requirement that must be met for a route to be considered feasible and loop-free. This requirement states that in order for a route to be feasible the Advertised Distance of the alternate route must be lower than that of the Feasible distance of the current route. (See Example)
Feasible Successor	If an alternate route exists and it meets the requirements of the Feasibility Requirement then it is considered a Feasible Successor.
Neighbor Table	The neighbor table contains all information related to the neighbors within EIGRP. An independent copy of the neighbor table is kept per protocol (IPv4 and/or IPv6).
Topology Table	The topology table within EIGRP differs from the routing table. The topology table keeps track of all routes that have been learned and meet requirements. An independent copy of the topology table is kept per protocol.

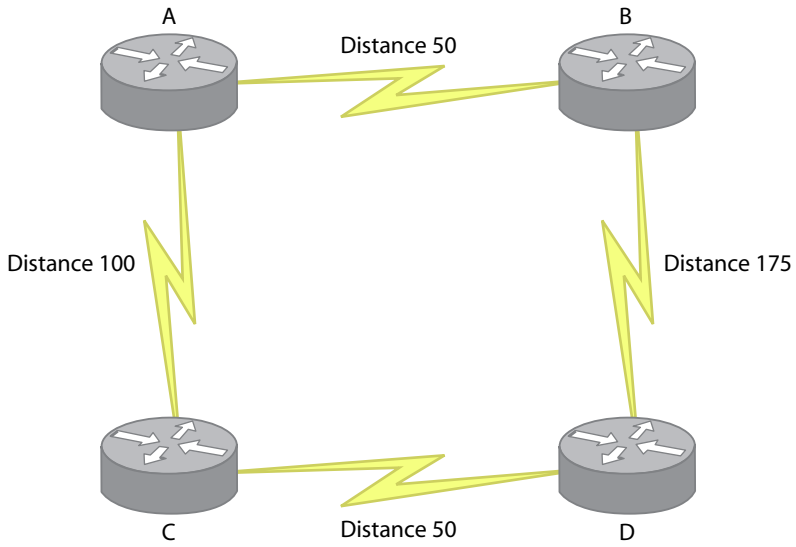


Figure 3 - EIGRP Distance Example

In the above figure a route from A to D can be accomplished in two ways: from A to B to D or from A to C to D. Now if we do the math, a route from A to B equals 50 and the route from B to D equals 175, so this route's total feasible distance is 225. The route from A to C equals 100 and the route from C to D equals 50, so this route's feasible distance is 150. So here is what we have in the tables:

Router	FD	AD
A to B to D	225	175
A to C to D	150	50

So, according to these tables, the main route would be from A to C to D with an FD of 150. At this point, we must verify that the potential alternate route (A to B to D) meets the feasibility requirement. The FD of the current route is 150 and the advertised distance of the alternate route is 175. In this case, FD < AD, so the route from A to B to D would not be considered feasible. Because the only alternate route is not considered feasible, there is only one path considered by router A to router D (A to C to D). Only if this path goes down would router A query router B to check for a new route to Router D. If the alternate route would have passed the feasibility requirement then both routes would have existed in the topology table on router A and any failure would have an almost instantaneous failover to the alternate route.

Packet Types

Hello	*Hello* packets are used for neighbor discovery and communications. *Hello* packets do not require acknowledgement. *Hello* packets are sent via multicast on 224.0.0.10.
Update	*Update* packets are used to exchange the reachability information of destinations. When new neighbors are discovered, packets are sent via unicast during initial exchanges. All updates other than this are sent via multicast on 224.0.0.10. All updates are transmitted reliably and acknowledged.
Acknowledgement	*Acknowledgement* packets are used to verify reception of *update* packets. *Acknowledgement* packets are always sent via unicast.
Query	*Query* packets are sent out when a router has topology change and has no feasible successors. *Query* packet are sent via multicast on 224.0.0.10.
Reply	*Reply* packets are sent out in response to *query* packets. *Reply* packets are sent via unicast.

Neighbor Discovery and Communications

EIGRP uses a *hello* packet mechanism in order to communicate with its neighbors and establish adjacencies. This is done via multicast on 224.0.0.10. When each router sets up adjacency, the neighbors are added to a neighbor table. Unlike some other routing protocols, EIGRP keeps track of all routes advertised to it by neighbors. This is used so that faster convergence can be achieved when alternate routes (feasible routes) exist. All of these routes are added to the topology table. All routes that are calculated to have the smallest cost are installed into the routing table. Adjacencies stay active through the use of *hello* packets which are exchanged between each EIGRP router. The interval at which *hello* packets are sent is determined by two things: the type of interface and the speed of that interface.

A 5-second interval is used for the following:

Ethernet
Token Ring
FDDI
Point-to-Point Serial interfaces and subinterfaces
Point-to-Multipoint interfaces and subinterfaces above T1 speeds (1.544 Mbps)

A 60-second interval is used for the following:

Point-to-Multipoint interfaces and subinterfaces below T1 speeds (1.544 Mbps)
ATM switched Virtual Circuits (VC)
ISDN BRI

All routes are considered active and valid unless 3 consecutive *hello* packets are missed. This is considered the *hold time*. The *hello* interval and the *hold time* can be changed with the **ip hello-interval eigrp** and **ip hold-time eigrp** interface commands. When changing the *hello* interval, the *hold time* needs also to be changed.

If you are setting up EIGRP on a network which uses secondary IP addresses on the interfaces EIGRP will only setup adjacencies on the primary addresses. Another thing that must be configured to establish adjacencies when using frame-relay map command is the optional broadcast option. If this option is not used in this configuration then adjacencies will not be established.

EIGRP, in recent releases, has also implemented a "Goodbye" message which is sent to all neighbors when an EIGRP router process is shutting down. This informs the neighbors of the impending topology change, thus allowing the neighbors to more efficiently recalculate topology change.

Split Horizon and Poison Reverse

In addition to the feasibility requirement, EIGRP uses two other techniques for preventing loops: Split Horizon and Poison Reverse. These two techniques are used under different circumstances. Split Horizon is a rule which states that once a route is learned from one interface it should not be advertised back to the same interface on which it was learned on. Poison Reverse is a rule which states that once a route is learned through an interface then that route should be advertised out to that interface as unreachable. Under normal operating conditions, Split Horizon is turned on and Poison Reverse is turned off.

Within EIGRP these two rules are used in two different situations: on startup, when there is a topology change, and when queries are sent. On startup, an EIGRP router sends out *hello* packets to discover the network topology, while in this state EIGRP utilizes Poison Reverse by sending out an unreachable advertisement for every table entry received. On topology changes, it is typical for all routers to utilizeSplit Horizon to prevent routing loops. However, when there is a topology change, which changes the interface through which the router reaches a network, then the router will turn off Split Horizon and poisons the old routes out all interfaces.

EIGRP Route States

Within EIGRP routes can be in one of two states, active and passive. Routes considered active are currently being computed; routes in a passive state have been computed and are in the topology and routing tables.

Stuck-In-Active (SIA) Routes

Within EIGRP there is an error message that shows a route as Stuck-In-Active (SIA). What this means is that a route has gone down on the current router and there is no feasible successor. Because of this it queries its neighbors. If the router does not get a response back from this query within the time allotted (around three minutes), then the neighbor will be cleared and the route will be considered SIA.

EIGRP Pacing

EIGRP pacing is the act of limiting the amount of routing bandwidth allowed over a connection. By default, EIGRP limits the amount of available bandwidth to 50%. Now, for higher bandwidth links this is not a problem, but for routers that are connected through low bandwidth links this could be service affecting. The amount of the available bandwidth EIGRP is allowed can be controlled through the **ip bandwidth-percent eigrp** *as-number percent* command.

EIGRP Route Types

EIGRP has three different types of route types: Internal, External and Summary. Internal routes are those routes which are calculated inside an EIGRP AS. External routes are routes that are injected from other routing protocols or from another EIGRP AS. Summary routes are internal routes that have been summarized at a network boundary. Keep in mind that each of these has a different administrative distance inside a Cisco router and are considered most relevant in order of summary, internal and external routes.

EIGRP Stub Routing

EIGRP has a stub routing feature that sets up a router to be at the edge of a network and not be responsible to respond to queries. All queries that are directed to a stub router will be responded to with an "inaccessible" message. Stub routers send out special neighbor packets that notify the neighbors about the routers stub status. Having a stub status does not disable neighbors from advertising routes to the stub router. If a "true" stub router is required or sought, the distribution router should be configured to only send out default routes to the stub router.

Load Balancing

EIGRP has the ability to support multiple unequal cost routes through the use of the **variance** command. There are two main things that EIGRP uses before a route is installed into the routing table. The route must meet the feasibility requirements and the metric of the route must be better than the "best" route times the variance multiplier. Now by default this variance multiplier is 1. This enables all routes with the same "best" metric to be installed in the routing table. The variance multiplier is able to be set through the **variance** command. So for example, if the variance multiplier was set to 2, then all routes that have a metric lower than 2 times the "best" route metric would be installed into the routing table.

EIGRP Metric Calculation

Metric calculation with EIGRP is quite complex and easy at the same time. By default, EIGRP uses two main metric variables that it uses for calculation, bandwidth and delay. However, EIGRP has the ability to use other metrics including load, reliability and MTU. It is vital with EIGRP that the metric variables used for calculation be used network-wide. Having metrics calculated differently across the network will result in routing problems as the metrics would be calculated differently on each router. Figure 4 shows the formula that EIGRP used to calculate the metric, through the use of K values, the metric changes which variables are used. By default the K values are the following:

K_1	1
K_2	0
K_3	1
K_4	0
K_5	0

$$EIGRPMetric = 256 \times \left(K_1 \times bandwidth + \frac{K_2 \times bandwidth}{256 - load} + K_3 \times delay \right) \frac{K_5}{reliability + K_4}$$

Default K Values: $K_1 = 1, K_2 = 0, K_3 = 1, K_4 = 0, K_5 = 0$

$$EIGRP\ DefaultMetric = 256 \times \left(1 \times bandwidth + \frac{0 \times bandwidth}{256 - load} + 1 \times delay \right)$$

(For the math people $\frac{K_5}{reliability + K_4}$ becomes 1 by deafult)

$$EIGRP\ DefaultMetric = 256 \times (bandwidth + delay)$$

$$bandwidth = 10{,}000{,}000(10\text{^}7) \div MinimumPathBandwidth(Kbps)$$

$$delay = sumofpathdelay \left(sumofpathdelay \text{ is in 10's of microseconds } \frac{\text{microseconds}}{10} \right)$$

load = Effective Load (0 to 255 = 0% to 100%)
The load is a five-minute exponentially weighted average that is updated every five seconds

reliability = Effective Reliability (255 to 0 = 100% to 0%)

Figure 4 - EIGRP Metric Formula

In order to give a good idea of how EIGRP handles route calculation we will go over a couple of examples. We will go over specific calculations from three different routing examples, networks 172.16.5.1, 172.16.2.1 and 10.0.0.21.

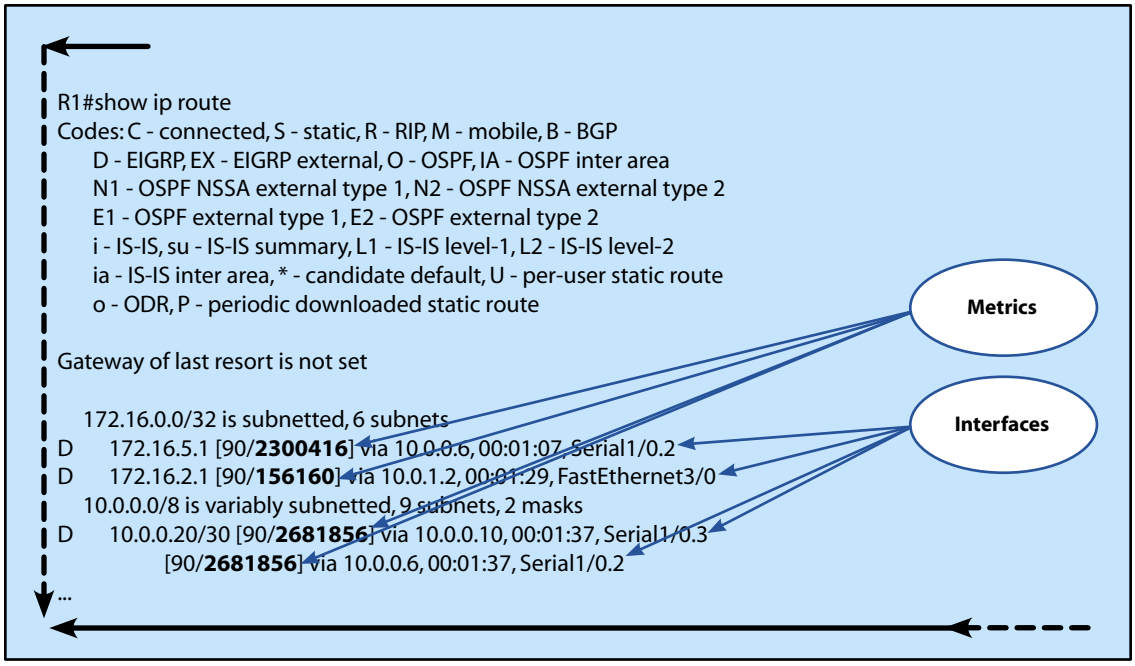


Figure 5 - show ip route (costs bolded)

Based on EIGRP metric calculations, the route to 172.16.5.1 is calculated by adding up the metrics for the path from source to destination. In this case, from Router **R1** to Router **R5**, according to the route table, this has a metric of **2300416**. This is calculated as follows:

EIGRPDefaultMetric = 256 x (*bandwidth* + *delay*)

$metric_{bandwidth}$ = 10,000,000 ÷ 1544 (minimum bandwidth = 1,544Kbps)

$metric_{bandwidth}$ = 6476 (Cisco routers use integers only, no rounding)

$$metric_{delay} = \frac{25100usec}{10} = 2510$$

metric = 256 x (6476 + 2510)

metric = 2,300,416

Figure 6 – R1 to 172.16.5.1 EIGRP Calculation

The second example is a route from Router **R1** to 172.16.2.1. In this case from Router **R1** to Router **R2**, according to the route table, this has a metric of **156,160**. This is calculated as follows:

EIGRPDefaultMetric = 256 x (*bandwidth* + *delay*)

$metric_{bandwidth}$ = 10,000,000 ÷ 100000 (minimum bandwidth = 100,000Kbps)

$metric_{bandwidth}$ = 100

$$metric_{delay} = \frac{5100usec}{10} = 510$$

metric = 256 x (100 + 510)

metric = 156,160

Figure 7 – R1 to 172.16.2.1 EIGRP Calculation

And finally, the third example is a route from Router **R1** to 10.0.0.21. In this case, from Router **R1** to Router **R3** or Router **R1** to Router **R4** to Router **R3**, according to the route table, these both have a metric of **2681856**. This is calculated as follows:

EIGRPDefaultMetric = 256 x (*bandwidth* + *delay*)

$metric_{bandwidth}$ = 10,000,000 ÷ 1544 (minimum bandwidth = 1,544Kbps)

$metric_{bandwidth}$ = 6476

$$metric_{delay} = \frac{40000usec}{10} = 4000$$

metric = 256 x (6476 + 4000)

metric = 2,681,856

Figure 8 – R1 to 10.0.0.21 EIGRP Calculation

EIGRP Route Summarization

EIGRP is unique because by default it performs classful summarization. This can be an advantage and disadvantage at the same time. If a network is only using IP networks at classful boundaries and has laid out their network hieratically, then automatic summarization will work fine. However, if the network is not setup like this, as most aren't, at least not completely, then automatic summary must be disabled. In order to disable automatic summary, the **no auto-summary** command must be issued in eigrp configuration mode.

EIGRP also gives the option for the engineer to configure manual summarization. This is done through the ip summary-address eigrp command. For specifics of the command, refer to the EIGRP configuration section. This command enables interface level configuration of summarization, thus limiting the amount of networks that are advertised out a specific interface. Manual summary on interfaces also limits EIGRP queries. This is useful because EIGRP queries must be answered, and by having a summary point at an interface the summarizing router limits queries at the summary point. This command should not however be used to generate default routes (0.0.0.0 0.0.0.0) from an interface. This causes the creation of a summary route going to the null0 interface with an administrative distance of 5. This can cause many routing problems inside the network. If a default route needs to be sent into a network, utilize the distribute-list command. For more information on this command refer to domain 4 in this manual.

EIGRP Security

EIGRP has the ability to authenticate adjacencies on each of its routed interfaces. This is done with a key that is used to create an MD5 hash that is then transmitted with routing traffic. This provides for a secure method of authentication by not transmitting the key over the wire.

EIGRP uses Message Digest Authentication (MD5) with key chains and variable numbers of keys that are used to create the MD5 hash, which is also called the "message digest". EIGRP uses key chains because they give the ability to utilize multiple keys for send and receives as well as allowing the use of send and receive lifetimes that can be configured to be used and then become inactive after a given amount of time. For configuration information, please refer to the EIGRP configuration section in this manual.

EIGRP Configuration

This section includes a number of basic commands that are required to setup EIGRP.

router

The first most important command in configuring EIGRP is **router eigrp *as-number***, which tells the router that it will be using EIGRP as a routing protocol. The *as-number* is a number that identifies the autonomous system which EIGRP is running. An autonomous system is an independent grouping. EIGRP can run several AS's on Cisco routers if needed.

Syntax:

router(config-router)#**router eigrp** *as-number*

network

The second most important command is **network** *ip-address* [**wildcard-mask**] that tells the router which networks are to be included within EIGRP. The *ip-address* is used to denote the network that is included; typically this is the subnet address of the network (192.168.1.0 for the 192.168.1.0/24 network). The *wildcard-mask* can be one of the parts of EIGRP commands that tends to confuse many engineers; specifically it is the inverse subnet mask of the network you want inside OSPF. In order to try to clear up any problems about this we will go over two examples, one with a common network and another with a less common network outside routing circles. Wildcard-masks are more commonly used in OSPF networks but can also be used with EIGRP as the same concepts apply.

Syntax:

router(config-router)#**network** *ip-address* [**wildcard-mask**]

Example:

For the first example we will show an example of including the 192.168.1.0/24 network:

192.168.1.0/24 is also written out to be 192.168.1.0 with a subnet mask of 255.255.255.0 and this in binary is:

11111111.11111111.11111111.00000000

Since we are looking for the inverse mask we will take the inverse of this subnet mask:

00000000.00000000.00000000.11111111

Once you turn this back into decimal you get 0.0.0.255 and this is the inverse mask you would use in the command. So now that we have this we will write out the command with this information;

router(config-router)#**network** 192.168.1.0 0.0.0.255

The former example was rather simple, using a well known network and subnet mask. Iin our second example, we will make it a bit harder. We will show an example including the 1.2.3.0/22 network:

1.2.3.0/22 is CIDR notation and is also written out to be 1.2.3.0 with subnet mask of 255.255.252.0 and this in binary is:

11111111.11111111.11111100.00000000

Since we are looking for inverse mask again we will take the inverse of this subnet mask:

00000000.00000000.00000011.11111111

Now once you turn this back into decimal you get 0.0.3.255 and again this is the inverse mask you would use in the command;

router(config-router)#**network** 1.2.3.0 0.0.3.255

Another thing to note is that you can also specify a single interface address. In order to do this you use the inverse mask 0.0.0.0, so if you wanted to use the address 1.2.3.4 you would use the following command:

router(config-router)#**network** 1.2.3.4 0.0.0.0

ip bandwidth-percent eigrp

This command is used by EIGRP to limit the amount of bandwidth used specifically for EIGRP on a specified interface. The *as-number* specifies which AS you are specifying for the command. The *percent* is a percentage of available bandwidth that EIGRP is allowed to use for the interface. The default is 50%.

Syntax:

router(config-if)# **ip bandwidth-percent eigrp** *as-number percent*

variance

The **variance** *multiplier* command is used when configuring unequal load balancing. The *multiplier* is used in order to configure how a route is chosen for insertion into the routing table. By default, the variance is set to 1. This command enables the use of a multiplier other then 1.

Syntax:

router(config-router)# **variance** *multiplier*

metric weights

The **metric weights** *tos k1 k2 k3 k4 k5* command is used in order to change the K-values. By default, K_1 and K_3 are equal to 1 and K_2, K_4 and K_5 are equal to 0. *tos* should always be set to 0 and the K-values can be set accordingly. The K-values must be the same across an EIGRP AS or the potential for problems arises because different routers will calculate metrics for routes differently.

Syntax:

router(config-router)#**metric weights** *tos k1 k2 k3 k4 k5*

auto-summary

The **auto-summary** command is used to turn on or off automatic summarization.

Syntax:

router(config-router)#**auto-summary (on by default)**

router(config-router)#**no auto-summary**

ip summary-address eigrp

The **ip summary-address eigrp** *as-number ip-address mask* [*admin-distance*] command is used in order to setup a summary route on a specified interface. The *as-number* specifies the AS to which the summary is referring. The *ip-address* and *mask* are used to set the summary address and the subnet of the mask to specify the boundary of the summary. The *admin-distance* can be used to change the administrative distance of the summary. By default, the administrative distance for EIGRP summary routes is 5.

Syntax:

router(config-if)#**ip summary-address eigrp** *as-number ip-address mask* [*admin-distance*]

ip authentication mode eigrp

The **ip authentication mode eigrp** *as-number* **md5** command is the first command used to setup EIGRP authentication between AS routers on a specified interface. The *as-number* is the AS of EIGRP being setup for authentication. **md5** is the type of authentication used.

Syntax:

router(config-if)#**ip authentication mode eigrp** *as-number* **md5**

ip authentication key-chain eigrp

The **ip authentication key-chain eigrp** *as-number key-chain* command is the second command used to setup EIGRP authentication between AS routers on a specified interface. The *as-number* is the AS of EIGRP being setup for authentication. The *key-chain* is the name of the key chain used to authenticate adjacencies.

Syntax:

router(config-if)#**ip authentication key-chain eigrp** *as-number key-chain*

key chain

The **key chain** *name-of-chain* command is used to setup a key chain for use with EIGRP authentication. The *name-of-*chain is the name of the key chain and must match the name used in the **ip authentication key-chain eigrp** command above.

Syntax:

router(config)#**key chain** *name-of-chain*

key

The key *key*-id command is used to setup the keys for use with authentication. A number of keys can be used for authentication and used with different key-strings and use timers. The *key*-id is the number given to the key.

Syntax:

router(config-keychain)#**key** *key-id*

key-string

The **key-string** command is used in order to set a passphrase for authentication of other EIGRP routers within the AS. The *text* is the passphrase that will be used on the interface on both sides of the connection.

Syntax:

router(config-keychain-key)#**key-string** *text*

accept-lifetime

The **accept-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*} command is used to control how long a specific key will be considered valid for incoming communications. By default, keys will be accepted for an infinite amount of time. *Start-time* and *end-time* will be entered in the format *hh:mm:ss Month date year*.

Syntax:

router(config-keychain-key)#**accept-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*}

send-lifetime

The **send-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*} command is used to control how long a specific key will be considered valid for outgoing communications. By default, keys will be accepted for an infinite amount of time. *Start-time* and *end-time* will be entered in the format *hh:mm:ss Month date year*.

Syntax:

router(config-keychain-key)#**send-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*}

ip hello-interval eigrp

The **ip hello-interval eigrp** *as-number seconds* command is used to change the hello interval. By default, the hello interval is either 5 or 60 depending on the type and speed of the interface. The *as-number* specifies the AS which hello interval is being modified.

Syntax:

router(config-if)#**ip hello-interval eigrp** *as-number seconds*

ip hold-time eigrp

The **ip hold-time eigrp** *as-number seconds* command is used to change the hold-time interval. By default, the hold-time is three times the hello interval. However, if the hello interval is changed manually by the ip hello-interval eigrp command, the hold-time is not automatically modified and must be changed when the hello interval is changed. The *as-number* specifies the AS which hold-time is being modified.

Syntax:

router(config-if)#**ip hold-time eigrp** *as-number seconds*

ip split-horizon eigrp

The **ip split-horizon eigrp** *as-number command* is used to enable or disable split-horizon.

Syntax:

router(config-if)#**ip split-horizon eigrp** *as-number*

router(config-if)#no **ip split-horizon eigrp** *as-number*

eigrp stub

The **eigrp stub** [**receive-only** | **connected** | **static** | **summary** | **redistributed**] command is used to configure a router as a stub router. **receive-only** is used to configure the router as a receive-only stub router. **connected** is used to advertise connected routes. **static** is used to advertise static routes. **summary** is used to advertise summary routes. **redistributed** is used to advertise redistributed routes.

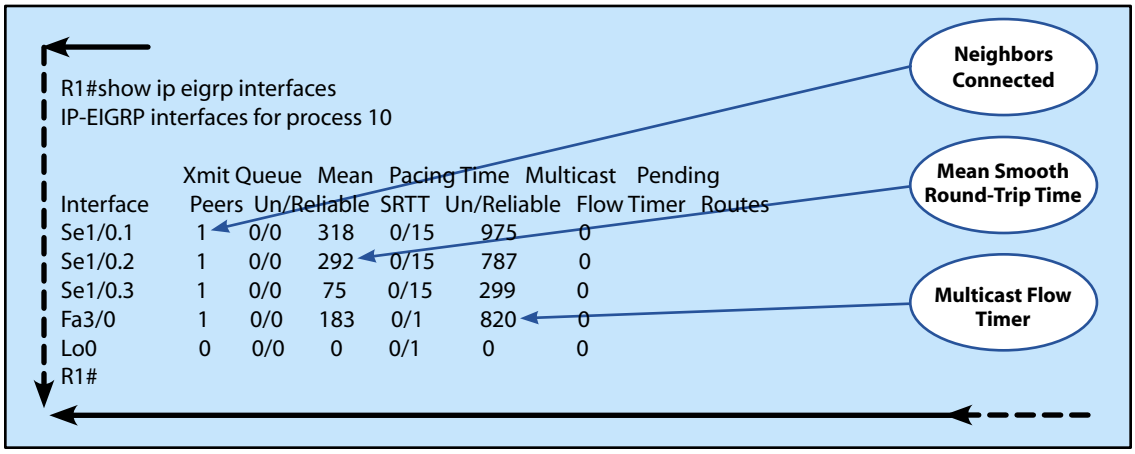
Sytnax:

router(config-router)#**eigrp stub** [**receive-only** | **connected** | **static** | **summary** | **redistributed**]

EIGRP Troubleshooting

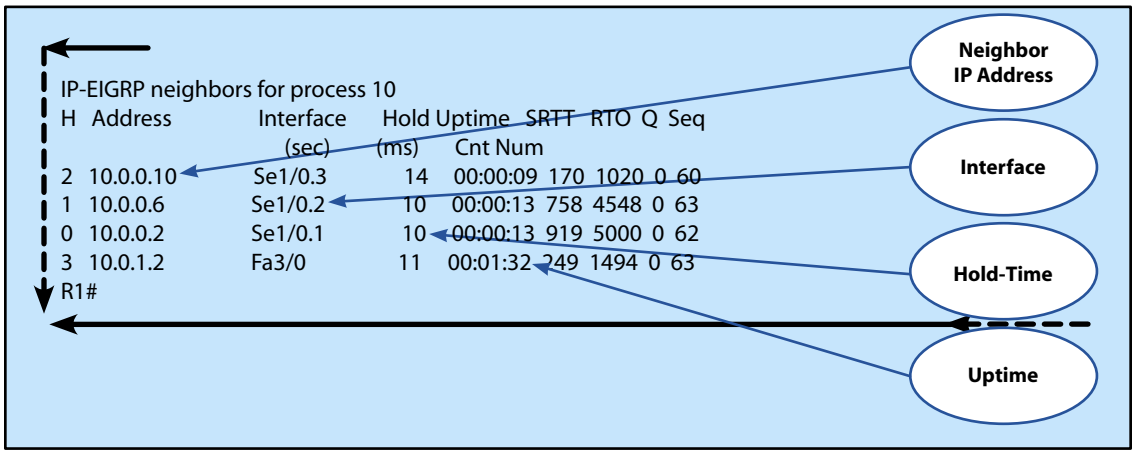
show ip eigrp interfaces

This command is used to display information about EIGRP configured interfaces.The following highlights the most important parts.



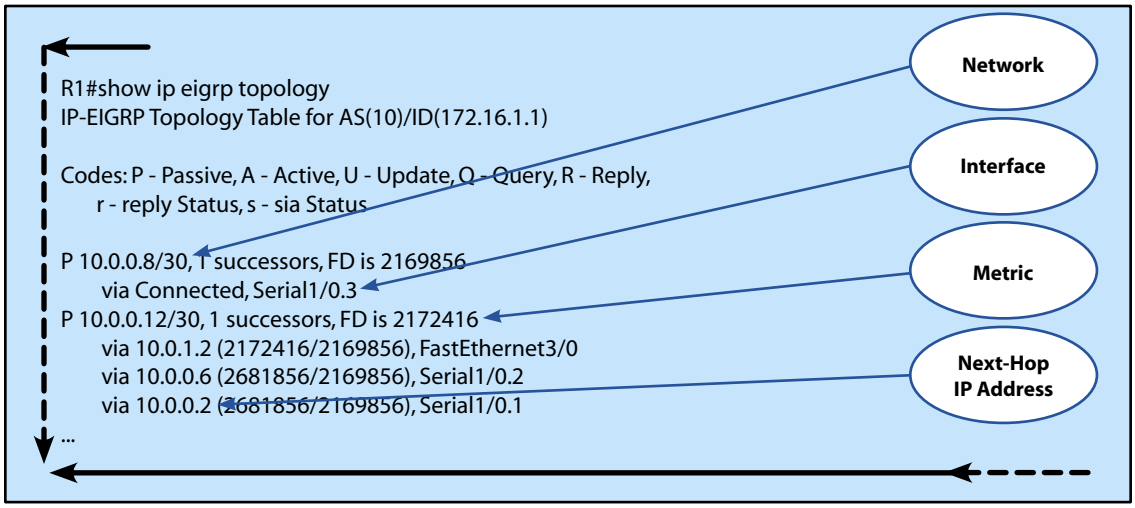
show ip eigrp neighbors

This command is used to display information about EIGRP neighbors. The following highlights the most important parts.



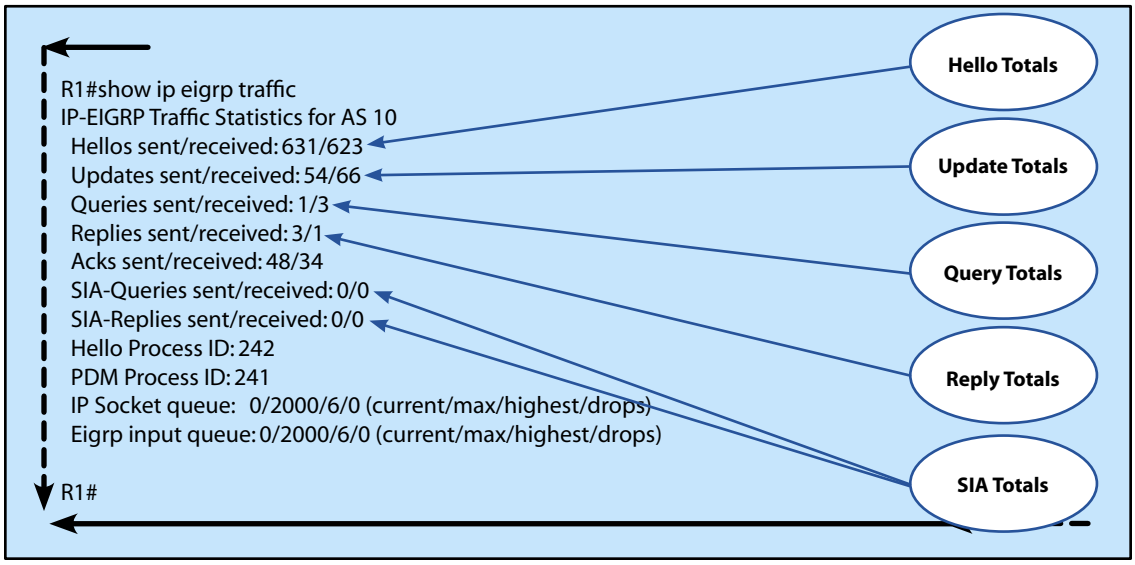
show ip eigrp topology

This command is used to display information about EIGRP topology. The following highlights the most important parts.



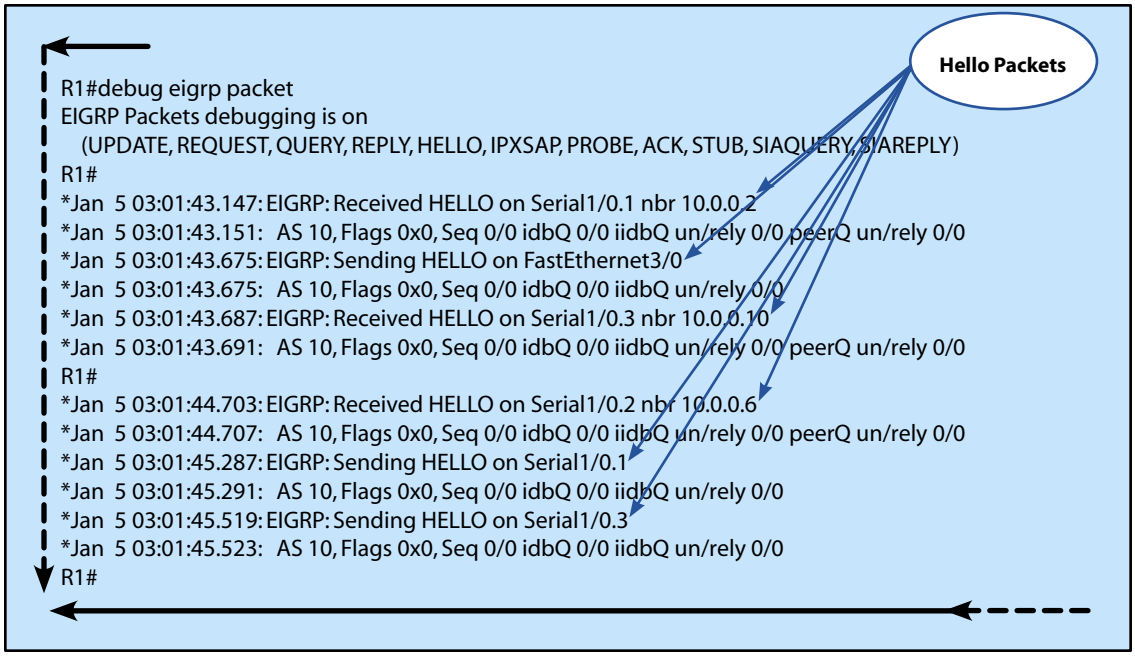
show ip eigrp traffic

This command is used to display information about EIGRP traffic. The following highlights the most important parts.



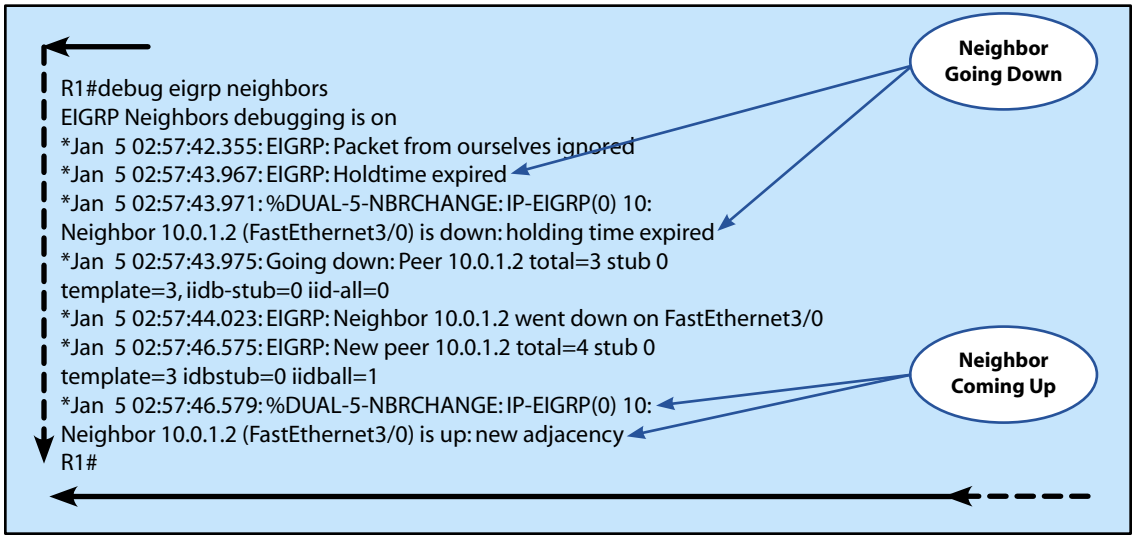
debug eigrp packet

This command is used to display EIGRP packet debugging information.



debug eigrp neighbors

This command is used to display EIGRP neighbor debugging information.



Domain 2 - Implement Multiarea OSPF Operations

OSPF Overview

OSPF stands for Open Shortest Path First and is a link-state routing protocol that makes use of the Dijkstra Shortest Path First (SPF) algorithm. It is based on RFC 2328, which is an industry standard and unlike EIGRP and IGRP which are both Cisco proprietary. OSPF has many advantages over distance vector protocols (RIP and IGRP), which includes:

Quick Convergence
Bandwidth Conservation
Use of Multicast over Broadcasts
Enables Incremental Updates
Classless IP Support
Supports Summarization
Encourages Hierarchical Design

Within the Cisco world OSPF has an administrative distance of 110.

OSPF Assumptions and Network Types

OSPF makes some assumptions that are not always possible. One of these assumptions is that all routers can be contacted via multicast. Now on a broadcast network type like Ethernet this is not a problem, but it is a problem on NonBroadcast MultiAccess (NBMA) networks that don't allow multicasts or broadcasts.

There are solutions to this problem however in order to account for these types of NBMA networks. Cisco provides four different network types that can be configured to the NBMA interface:

Broadcast Multiaccess	
Point-to-Point	
Point-to-Multipoint	Point-to-Multipoint Broadcast (default) Point-to-Multipoint Nonbroadcast
Nonbroadcast Multiaccess (NBMA) (Default for Serial Interfaces)	

Of the above interfaces, the following are Cisco specific:

Broadcast Multiaccess
Point-to-Multipoint Nonbroadcast
Point-to-Point

Within OSPF, most communications are controlled through one of the routers, which are considered the Designated Router (DR). The DR is used to control communication within a given segment, typically a broadcast segment (i.e. Ethernet). Other routers within the segment are capable of becoming Backup Designated Routers (BDR) in case something happens to the DR. The process of becoming a DR or BDR is done through an election process. The election process uses OSPF priority to choose the DR then the BDR's. In the case of a tie the Router ID (RID) is used (typically, it is easiest to configure a loopback address on the router that will become the RID). Broadcast and NBMA network types make use of DR's and expect a full mesh topology. Cisco designates routers with a priority of 0 are given the designation DROTHER. These routers cannot become the DR or a BDR and are user configurable. These routers are typically the routers that don't have the hardware or software requirements the DR requires.

Point-to-Multipoint links do not perform a DR election and all neighbors by default are discovered automatically. Cisco also provides a Nonbroadcast option for Point-to-Multipoint links. If this option is used, neighbors must be manually configured. This option is typically used in a situation when the media does not have an ability to perform multicast discovery and does not have the ability to emulate multicast discovery as used in NBMA.

The main difference between broadcast and non-broadcast network types is the network cannot utilize multicast to discover neighbors. Because of this, all neighbors must be manually configured on each router. But within a broadcast router type, all neighbors are automatically discovered.

Point-to-Point links know their neighbors because they are only linked to each other.

> Cisco serial interfaces default to using NBMA for the physical interface; this however changes if subinterfaces are used. Cisco gives two options for subinterfaces: Point-to-Point and Multipoint. Point-to-Point subinterfaces by default are Point-to-Point OSPF links. Multipoint subinterfaces by default are NBMA like the physical but can be configured to be any OSPF link type except Point-to-Point.

In order to determine the status of its neighbors, each router sends out periodic *hello* packets to all its neighbors, and to each *hello* it expects an acknowledgement. If the router has not received an acknowledgement from its neighbor by the time the *dead timer* expires, it assumes the neighbor is down and takes the appropriate action in changing its routing tables. Specific OSPF network types have different timers. The following table shows the timer differences and summarizes the different router types:

	Nonbroadcast (NBMA)	**Point-to-Multi-point (Broadcast)**	**Point-to-Multipoint nonbroadcast**	**Broadcast**	**Point-to-Point**
DR/BDR	Yes	No	No	Yes	No
Identify Neighbor?	Yes	No	Yes	No	No
Timer Intervals (Hello/Dead)	30/120	30/120	30/120	10/40	10/40
RFC or Cisco	RFC	RFC	Cisco	Cisco	Cisco
Network Supported	Full Mesh	Any	Any	Full Mesh	Point-to-Point

Figure 9 - Router Type Difference

Neighbor Discovery and Communications

In order for OSPF to discover its neighbors a mechanism must be in place to perform this function and continue to communicate throughout OSPF. This discovery process is done via a *hello* packet that is sent out via multicast. All potential neighbors receive this packet and respond with their corresponding information via unicast. Once initial communications are established, Database Descriptor (DBD) packets are exchanged notifying each other of their interfaces and area memberships. All LSA updates are sent out via multicast.

OSPF details a number of neighbor states that show what part of this conversation the router is in. These are:

Down	This is the first OSPF state, a router is in this state when it has not received any *hello* packets.
Attempt	This OSPF state is only valid for manually configured neighbors; a router is in this state when it has not received any response to unicast packets to its neighbors.
Init	This OSPF state is when a router has received a *hello* packet but it *did not* contain its own router id. The router id is inserted into the *hello* packet for acknowledgement of receipt.
2-Way	This OSPF state is achieved when a router has received a *hello* packet and it *did* contain its own router id. At this stage DR and BDR election occurs, if needed.
Exstart	In this OSPF state, routers and their DR's and BDR's establish a master-slave relationship.
Exchange	In this OSPF state, routers exchange DBD packets that contain link-state header information and the entire link-state database. This information is compared and the most current information is obtained from the neighbors.
Loading	In this OSPF state, routers exchange actual link state information. This is done through the exchange of link state request packets and link state update packets. All link state packets are acknowledged.
Full	In this OSPF state, routers are fully adjacent with each other and completely synchronized.

Areas

OSPF's way of providing division within a routing domain is to define *areas*. *Areas* provide a method for network designers to separate routes to specific parts of the network. These routes from *area* to *area* are summarized and inserted into the adjacent *areas*. *Areas* also run separate SPF processes. This allows constantly changing networks to be isolated away from the stable networks thus speeding up routing and optimizing router time. OSPF has a few requirements, which are required for design. The requirements affect OSPF design in two ways. OSPF requires an area 0 (or 0.0.0.0), which is considered the *backbone* area,. In single area OSPF networks, this would be the only area. The second requirement is that all areas must be directly connected to area 0. The only exception to this is virtual-links, which will be detailed later in this document.

OSPF has a number of different types of areas; these areas are described as follows:

Standard Area	Within a standard area all routers are aware of all networks with in the area. All routers within the area have a complete copy of the topology database.
Stub Area	Stub areas are the same as standard areas except Type 5 LSA's (External Traffic) are not allowed. The information that is contained within the Type 5 LSA's is distributed via a default route which is injected into the area by the ABR's. All egress external traffic is routed via the areas closest ABR. Stub areas are used when routers within an area have limited processing and memory capabilities, limiting the number of routes within the routing tables and database.
Totally Stubby Area (Cisco Proprietary)	Totally stubby areas are a creation specific to Cisco, this type of area does not allow Type 3, 4 or 5 LSA's. The information that is contained within the Type 3, 4, and 5 LSA's is distributed via a default route, which is injected into the area by the ABR's. All egress external traffic is routed via the area's closest ABR. Like stub areas, totally stubby areas also limit the requirements of the routers within the area. Totally stubby areas limit not only external routes from getting into the area's routing tables and databases but also Inter-area routes.
Not So Stubby Area (NSSA)	NSSA's are considered stub areas that contain an ASBR. Since stub areas don't allow Type 5 LSA's, NSSA's get around this with Type 7 LSA's by disguising Type 5 LSA's through the area. The ABR's convert Type 7 LSA's into Type 5 LSA's at the area boundary.

Router Types

Inside OSPF there are a number of different types of routers; each type has its own specific duties within the network and within each specific area. The following is a description of these router types and their duties:

Internal Router	An internal router is any router that resides inside only one area. Its main duties are to compute the SPF calculations within the area (from its point of view) and to forward packets in the way calculated by SPF.
Backbone Router	A backbone router has an interface inside the Backbone area (Area 0 or Area 0.0.0.0).
Area Border Router (ABR)	An area border router has interfaces in multiple areas and thus borders multiple areas. Because this router has interfaces in multiple areas it also has a full copy of topology of all areas of which it is a member. The ABR's are responsible for routing between areas AND routing inside each area it is connected to.
Autonomous System Boundary Router (ASBR)	An autonomous system boundary router has an interface that connects into an OSPF network and an interface that connects into some other type of routing protocol domain (i.e. RIP). The ASBR's are responsible for advertising external route information into OSPF and from OSPF into the other routing protocol domain. Although ASBR's are typically also a backbone router they can also exist in other areas.

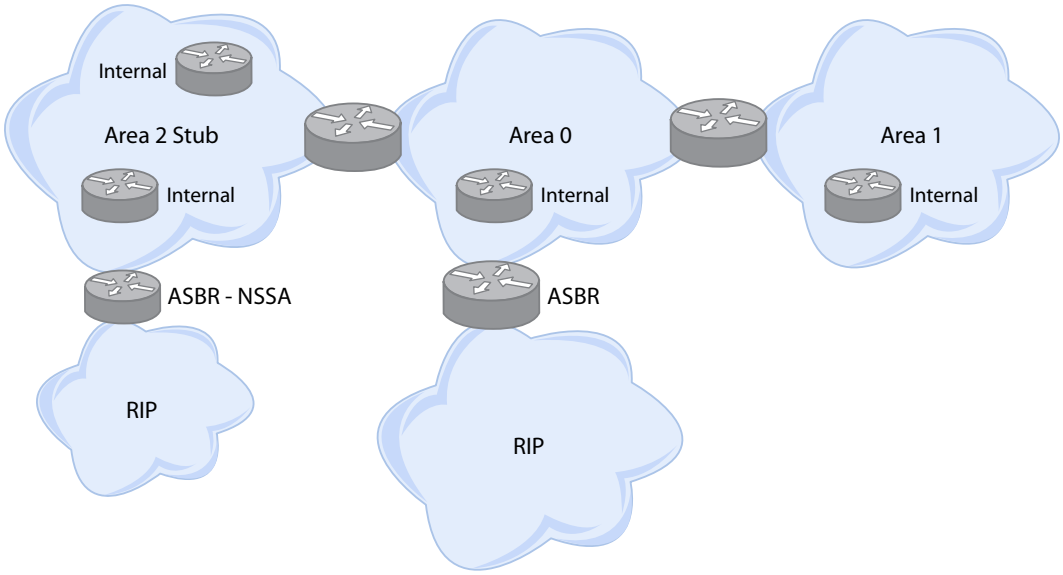


Figure 10 - Multiple Area Example

Router Communications

OSPF, like other link-state protocols, requires that all routers within an *area* must have a complete copy of the topology of the network. This enables the router to compute the most efficient path to all destinations. Multicast communications between routers is on 224.0.0.5. All communications between DR's is on 224.0.0.6.

Packet Types

Hello	*Hello* packets are used for neighbor discovery and communications. *Hello* packets are sent via multicast on 224.0.0.5
Database Description	Database Description packets are used to exchange topological information between neighbors. Database Description packets are sent via unicast.
Link-State Request	Link-State request packets are sent when a router thinks that part of its database are out of date. The request is sent to a neighbor requesting there database so the router can compare it against its database.
Link-State Update	Link-State Update packets contain one or more Link-State Advertisements. Updates are sent via multicast on 224.0.0.5. All updates are acknowledged. If retransmission is needed an update is then sent via unicast.
Link-State Acknowledgment	Link-State Acknowledgements are used to acknowledge update packets from neighbors.

In order for this type of communication to be possible, OSPF uses LSA's (Link State Advertisements). Within OSPF there are six types of LSA's:

Router Link LSA (Type 1)	Generated by every router, lists their neighbors and the cost to each. These are flooded via multicast 224.0.0.5.
Network Link LSA (Type 2)	Generated by the designated routers, lists all of the routers that are on the segment they are adjacent to. These are flooded via multicast 224.0.0.5.
Network Summary Link LSA (Type 3)	Generated by the ABR's between areas, list the prefixes available in each corresponding area. Summarization is propagated via Type 3 LSA's. Inter-area traffic is advertised inside OSPF with the designation IA.
AS External ASBR Summary Link LSA (Type 4)	Generated by the ASBR's between areas, this is used to notify the network of the presence of the ASBR's.

Link State Advertisements cont'd on next page.

Link State Advertisements cont'd.

External Link LSA (Type 5)	Generated by the ASBR's, this is used to advertise external routes (from other routing processes) into OSPF and to advertise default routes out of OSPF. These are flooded to all areas except stubs. External traffic is advertised inside OSPF with two designations, External-Type 1(E1) and External-Type 2(E2-Default). E2 routes costs use the advertised cost only. E1 routes use the advertised cost as a base cost and add all internal cost to compute the final cost.
NSSA External LSA (Type 7)	Generated by the ASBR's that are present in a NSSA. These are used to tunnel Type 5 LSA's through a NSSA that does not allow Type 5 LSA's. The Type 7 LSA's are converted to Type 5 LSA's at the ABR.

OSPF Route Calculation

In order to calculate the best route for a given destination OSPF needs to calculate a *cost* for each destination network it can reach. This process can be quite complex. The following is the formula that OSPF uses:

$$Cost = \frac{100{,}000{,}000bps}{LinkSpeed}$$

Figure 11 - OSPF Cost Formula

This formula is used on every link between the source and the destination. In order to give everyone a good idea of how OSPF handles route calculation we will go over a couple examples. The following is a ***show ip route*** command that was issued on Router **R1**. We will go over the specific calculations for three different routing examples, networks: 172.16.5.1, 172.16.2.1 and 10.0.0.21.

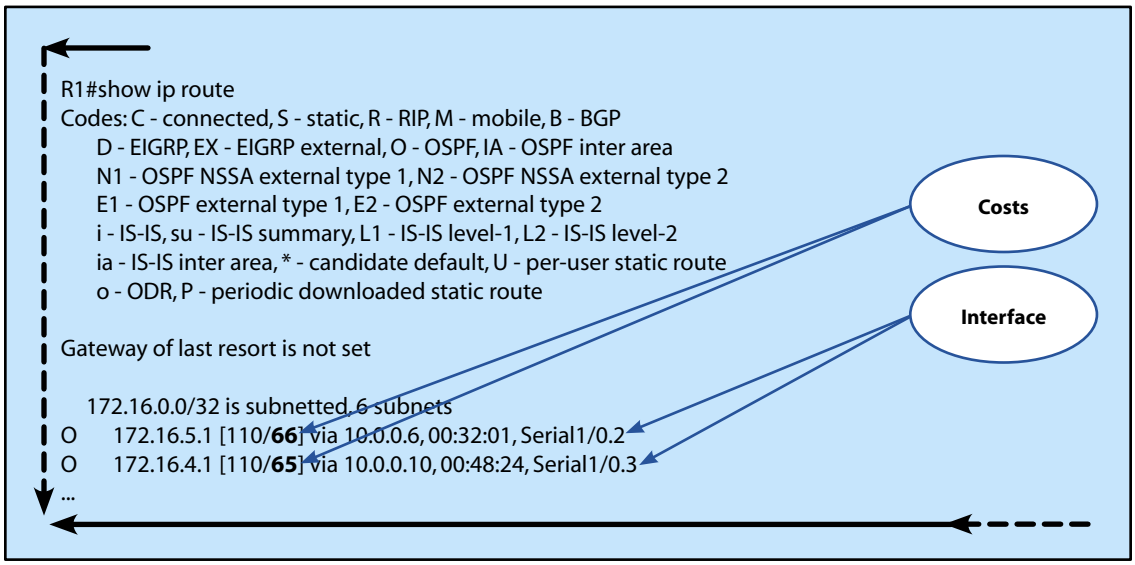


Figure 12 - show ip route (costs bolded)

Based on OSPF calculating ***cost*** by the formula in Figure 11, the route to 172.16.5.1 is calculated by adding up the cost of each link between the source and the destination router. In this case from Router **R1** to Router **R5**, according to the route table, this has a cost of **66**. This is calculated as follows:

$$\text{cost}_{R1\text{-}R3} = \frac{100{,}000{,}000 bits}{1{,}544{,}000 bits} = 64.77 = 64 \text{ (cisco routers use integers only, no rounding)}$$

$$\text{cost}_{R3\text{-}R5_{FastEthernet}} = \frac{100{,}000{,}000 bits}{100{,}000{,}000 bits} = 1$$

$$\text{cost}_{R5_{FastEthernet}\text{-}R5_{Loopback0}} = \frac{100{,}000{,}000 bits}{8{,}000{,}000{,}000 bits} = 1$$

$$\text{cost}_{R1\text{-}R3} + \text{cost}_{R3\text{-}R5_{FastEthernet}} + \text{cost}_{R5_{FastEthernet}\text{-}R5_{LoopBack0}} = \text{cost}$$

$$64 + 1 + 1 = 66$$

$$cost = 66$$

Figure 13 – R1 to 172.16.5.1 OSPF Calculation

The second example is a route from Router **R1** to 172.16.2.1. In this case from Router **R1** to Router **R2**, according to the route table, this has a cost of **2**. This is calculated as follows:

$$\text{cost}_{R1\text{-}R2_{FastEthernet}} = \frac{100{,}000{,}000 bits}{100{,}000{,}000 bits} = 1$$

$$\text{cost}_{R2_{FastEthernet}\text{-}R2_{Loopback0}} = \frac{100{,}000{,}000 bits}{8{,}000{,}000{,}000 bits} = 1$$

$$\text{cost}_{R1\text{-}R2_{FastEthernet}} + \text{cost}_{R2_{FastEthernet}\text{-}R2_{LoopBack0}} = \text{cost}$$

$$1 + 1 = 2$$

$$cost = 2$$

Figure 14 – R1 to 172.16.2.1 OSPF Calculation

And finally the third example is a route from Router **R1** to 10.0.0.21. In this case from Router **R1** to Router **R3** or Router **R1** to Router **R4** to Router **R3**, according to the route table, these both have a cost of **128**. This is calculated as follows:

$$\text{cost}_{R1\text{-}R4} = \frac{100{,}000{,}000 bits}{1{,}544{,}000 bits} = 64$$

$$\text{cost}_{R4\text{-}R3} = \frac{100{,}000{,}000 bits}{1{,}544{,}000 bits} = 64$$

$$\text{cost}_{R1\text{-}R4} + \text{cost}_{R4\text{-}R3} = \text{cost}$$

$$64 + 64 = 128$$

$$cost = 128$$

or

$$\text{cost}_{R1\text{-}R3_{Serial1/0.1}} = \frac{100{,}000{,}000 bits}{1{,}544{,}000 bits} = 64$$

$$\text{cost}_{R3_{Serial1/0.1}\text{-}R3_{Serial1/0.3}} = \frac{100{,}000{,}000 bits}{1{,}544{,}000 bits} = 64$$

$$\text{cost}_{R1\text{-}R3_{Serial1/0.1}} + \text{cost}_{R3_{Serial1/0.1}\text{-}R3_{Serial1/0.3}} = \text{cost}$$

$$64 + 64 = 128$$

$$cost = 128$$

Figure 15 – R1 to 10.0.0.21 OSPF Calculation

Route Summarization

OSPF is capable of doing two different types of route summarization, Internal and External. Internal summarization is the act of summarizing a number of internally (within OSPF) controlled networks at an area boundary (ABR). The purpose of this is to limit the exposure of all routers within the OSPF process from needing to store and process topology information for a different area. By providing a summary of this information at an area boundary, routers in a separate area can have only a summary route entered into their routing tables instead of each separate network. External summarization has the same purpose, but instead of being a summary of internally controlled networks, it is a summary of externally controlled networks. External summarization is controlled at the exterior edge of the OSPF network to another routing protocol. Within OSPF this router is the ASBR.

Route Redistribution

Route Redistribution is the act of inserting external routes (other routing protocols) into OSPF; this is done at the ASBR. Remember, earlier in this document it was stated that there were two different types of external routes, External Type-1 and External Type-2 (Default). This is where the type of external route is configured. External Type-2 routes take the cost you configure at the ASBR and advertise that cost as fixed into the OSPF network. An External Type-1 route takes this configured cost and adds normal OSPF cost across the paths to the destination as normal. This redistribution is done through the **redistribute** command that will be detailed later in this manual.

It is also possible to redistribute static routes and connected interfaces into OSPF through the use of the **redistribute connected subnets** and **redistribute static subnets, respectively.**

OSPF Security

OSPF has the ability to authenticate routing traffic through the use of a password. By default, OSPF uses Null authentication, which enables any router to exchange routing information. OSPF provides two mechanisms to perform this: simple password authentication and message digest authentication.

The simple password authentication mechanism exchanges a password between the routers; if this password is verified then the routers continue to communicate. The password is *not* encrypted when transmitted across the network media, thus enabling the possibility of it being captured and used with a rogue router.

Message digest authentication uses a key-id and a password to compute a MD5 "message digest" that is exchanged over the network media with the other routers. If this digest matches then communications continue. Nothing in the "message digest" can be reversed to find the password, which is why this mechanism is preferred and considered more secure.

Virtual Links

Virtual links provide a workaround to one of OSPF's main rules, which states that all areas must directly connect to area 0. Virtual links should be used sparingly, mainly in two situations. You can use virtual links when two or more networks are joining and virtual links can be used as a stop-gap until the network can be redesigned with a central area 0. The second main use is in failure situations when a link to area 0 has been disconnected or has failed. In this situation a virtual link can be used if there is another path to area 0 through another transit area.

Virtual links have two main requirements. It must be established between two routers that share a common area and one of these routers must be also connected to the backbone (Area 0).

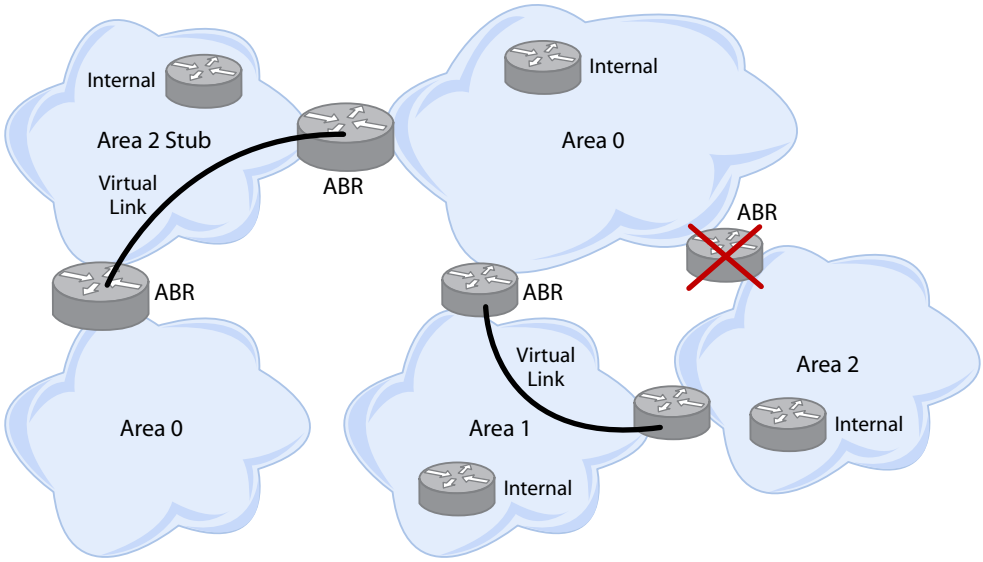


Figure 16 - Virtual Links Example

OSPF Configuration

This section includes a number of the basic commands that are required to setup OSPF in both single and multiple area networks.

router

The **router ospf *process-id*** command tells the router that it will be using OSPF as a routing protocol. The *process-id* is a number that internally identifies this instance of OSPF on this router. It is only relevant to the local router and does not have to be the same across your network. It is possible, although not common, to run more than one instance of OSPF on the same router.

Syntax:

router(config)#**router ospf** *process-id*

router-id

The **router-id** *ip-address* command is used to hard set the router-id of the OSPF process on the router. The *ip-address* is the IP address given as the router-id.

Syntax:

router(config-router)#**router-id** *ip-address*

network

The **network *address wildcard-mask*** **area *area-id*** command instructs the router as to which networks are to be included within OSPF. The *address* is used to denote the network that is included; typically this is the subnet address of the network (192.168.1.0 for the 192.168.1.0/24 network). The *wildcard-mask* can be one of the parts of OSPF commands that tend to confuse many engineers. The application of wildcard-masks is the same as described in the **network** section of EIGRP. The *area-id* is simply the area number you want to include this network into.

Syntax:

router(config-router)#**network** *address wildcard-mask* **area** *area-id*

neighbor

Within OSPF there are times when you are unable to utilize multicast. In this situation you must manually configure your neighbors. In this command, *ip-address* is the only required variable; all the other options are optional. The *priority* and *poll-interval* options are valid on all network types except point-to-multipoint. The *cost* option is valid on all network types except NBMA. The following command is used for this purpose:

Syntax:

router(config-router)#**neighbor** *ip-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*] [**database-filter all**]

- The *ip-address* option is the IP address of the neighbor.
- The **priority** *number* is used to change the priority of the neighbor.
- The **poll-interval** *seconds* is used to change the polling interval between the neighbors.
- The **cost** *number* assigns a cost from the local router to the neighbor. If not specified the interface cost is used.
- The **database-filter all** allows you to filter the outbound LSA to this neighbor.

ip ospf priority

In order to control the priority of the router, the command **ip ospf priority** *number* exists. The priority controls the election of the DR/BDR. The higher the number the more likely the router will be selected. If you set the router to a priority of 0, the router will never participate in the election.

Syntax:

router(config-if)#**ip ospf priority** *number*

- The *number* can be from 0 to 255.

ip ospf cost

In order to override the default cost assigned to an OSPF interface, you use the **ip ospf cost** *cost* command. If you are only connecting to other cisco routers it is recommended that you keep the defaults. Some other router vendors use different cost metric calculations. In these situations, it might be required to change the cost on specific interfaces.

Syntax:

router(config-if)#**ip ospf cost** *cost*

ip ospf network

In order to control what type of network type is assigned to an interface you must use the **ip ospf network** {**broadcast** | **non-broadcast** | {**point-to-multipoint** [**non-broadcast**] | **point-to-point**}} command.

Syntax:

router(config-if)# **ip ospf network** {**broadcast** | **non-broadcast** | {**point-to-multipoint** [**non-broadcast**] | **point-to-point**}}

auto-cost

The default auto-cost used by OSPF for cost calculations is 100Mbps. However, with today's higher speed interfaces it may be necessary to change this in order to differentiate between high-speed interface costs. The **auto-cost reference-bandwidth** *Mbps* command is used to change this.

Syntax:

router(config-router)#**auto-cost reference-bandwidth** *Mbps*

area stub

In order to define an area as either stubby or totally stubby you must use this command **area** *area-id* **stub [no-summary]**. The stub command must be used on all routers within the stub area. For totally stubby areas you must add the **no-summary** option on the ABR into the area.

Syntax:

router(config-router)# **area** *area-id* **stub [no-summary]**

- *area-id* is the area that will be stubby (or totally stubby).

area default-cost

Another important stub command is **area** *area-id* **default-cost** *cost*. This command is used to define the cost of the summary route that is injected into the stub area.

Syntax:

router(config-router)# **area** *area-id* **default-cost** *cost*

- *area-id* is the area that is stubby.
- *cost* is the cost that will be associated with the summary route.

area range

In order to configure internal summarization you must utilize the **area** *area-id* **range** *ip-address ip-address-mask* [**advertise** | **not-advertise**] [**cost** *cost*] command. The command it only to be used on the ABR. Specifically, internal route summarization limits the routes advertised from one area into the other areas. By using this command summary routes can be used to consolidate all individual routes into a limited amount of summary routes.

Syntax:

router(config-router)# **area** *area-id* **range** *ip-address ip-address-mask* [**advertise** | **not-advertise**] [**cost** *cost*]

- *area-id* is the area that is to be summarized.
- *ip-address* is the IP network that is going to be summarized.
- *ip-address-mask* is the subnet mask of the boundary you want to summarize at.
- **advertise** is an optional parameter that enables the advertisement of the route via Type-3 LSA's.
- **not-advertise** is an optional parameter that disables the advertisements of the route summary and all component routes are hidden from other areas.
- **cost** *cost* is the cost that is associated with the summary route into the other areas for SPF calculations.

summary-address

In order to configure external summarization you must utilize the **summary-address** {*ip-address mask* | *prefix mask*} [**not-advertise**] [**tag** *tag*] command. This command will only be used on the ASBR. Specifically external summarization affects how the OSPF network is advertised to an external routing protocol. By using this command, OSPF is summarized before its networks are advertised externally, limiting the overall amount of information sent to the external neighbor.

Syntax:

router(config-router)#**summary-address** {*ip-address mask* | *prefix mask*} [**not-advertise**] [**tag** *tag*]

- *ip-address* is the IP network that is going to be summarized.
- *mask* is the subnet mask of the boundary you want to summarize at.
- *prefix* is the IP route prefix of the Destination.
- *mask* is the subnet mask of the boundary you want to summarize at.
- **not-advertise** is an optional parameter that suppresses the routes matched by the prefix/mask pair.
- **tag** *tag* is an optional parameter that allows you to set a tag for use with route maps to control redistribution.

area default-cost

In order to set the default cost of the summary route advertised to other areas, the area *area-id* **default-cost** *cost* command is used.

Syntax:

area *area-id* **default-cost** *cost*

- *area-id* is the area that you wish to set the default cost of.
- *cost* is the cost of the summary route.

ip ospf authentication

The **ip ospf authentication** [**message-digest** | **null**] command is used to enable authentication on an interface. The **message-digest** parameter is used to enable the use of MD5 digests. The **null** parameter is the default and specifies that the authentication will be done with clear text.

Syntax:

router(config-if)#**ip ospf authentication [message-digest | null]**

ip ospf authentication-key

The **ip ospf authentication-key** *password* command is used to enable clear text OSPF authentication per interface. The *password* specifies the password used for authentication. Using this command specifies that the password will be sent in the clear; if security is important use the **ip ospf message-digest-key md5** command.

Syntax:

router(router-if)#**ip ospf authentication-key** *password*

ip ospf message-digest-key md5

The **ip ospf message-digest-key** *key-id* **md5** *key* command is used to enable OSPF MD5 authentication per interface. The *key-id* is a number from 1 to 255, which is used to identify the key. There is usually one key id per interface. The *key* is the password that is used to create the message digest for exchange with another router on the interface.

Syntax:

router(config-if)#**ip ospf message-digest-key** *key-id* **md5** *key*

area virtual-link

In order to configure a virtual link from one area to another, the **area** *transit-area-id* **virtual-link** *transit-router-id* [**hello-interval** *seconds*] [**retransmit-interval** *seconds*] [**transmit-delay** *seconds*] [**dead-interval** *seconds*] command must be used.

Syntax:

router(config-router)#**area** *transit-area-id* **virtual-link** *transit-router-id* [**hello-interval** *seconds*] [**retransmit-interval** *seconds*] [**transmit-delay** *seconds*] [**dead-interval** *seconds*]

- *transit-area-id* is the transit area between the two connecting areas.
- *transit-router-id* is the IP associated with the virtual link neighbor.

hello-interval *seconds* is an optional parameter that specifies the interval between hello packets.

retransmit-interval *seconds* is an optional parameter that specifies the interval between retransmissions for adjacencies.

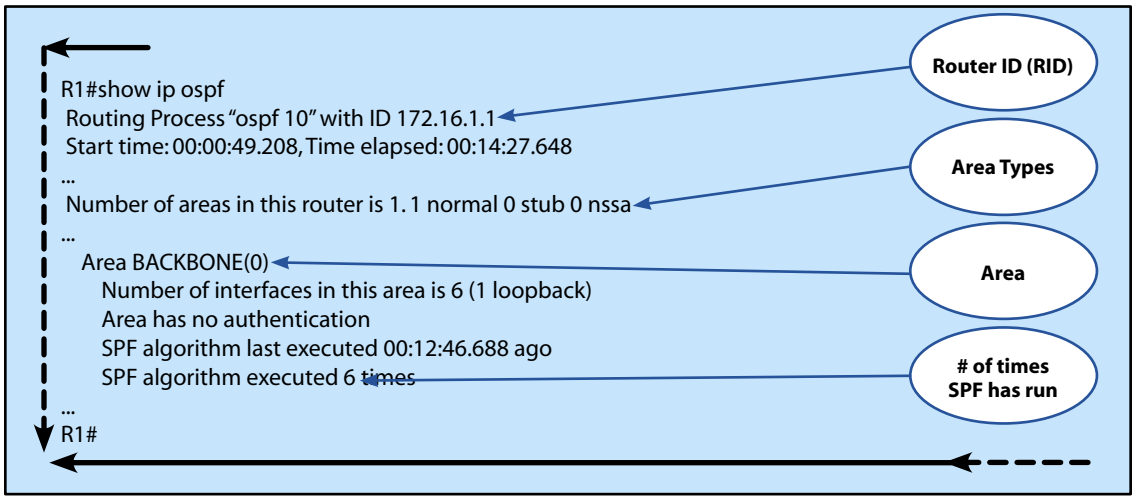
transmit-delay *seconds* is an optional parameter that specifies the estimated time it takes to transmit a link-state update packet on an interface.

dead-interval *seconds* is an optional parameter that specifies the interval between the time a router receives a hello packet and when it considers the adjacent router down.

OSPF Troubleshooting

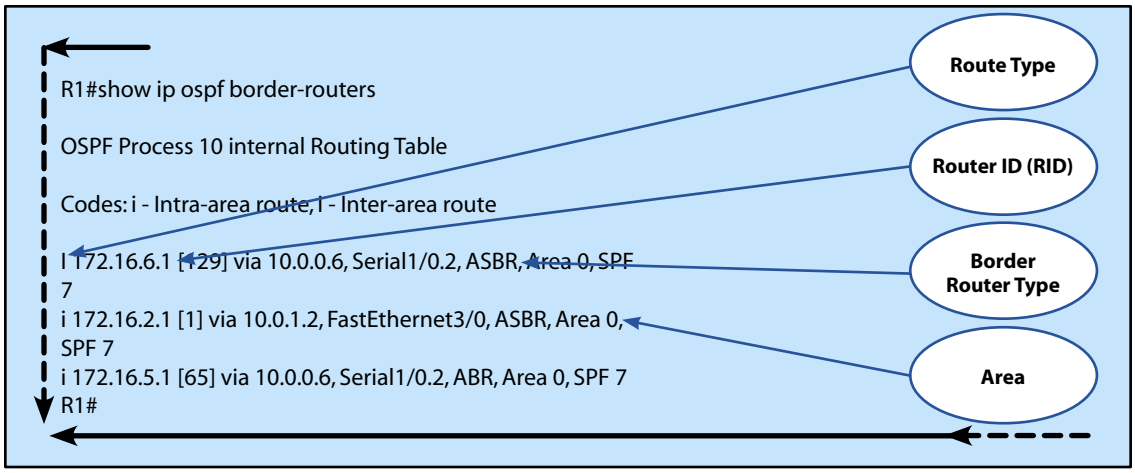
show ip ospf

This command is used to display general information about OSPF. The following highlights the most important parts.



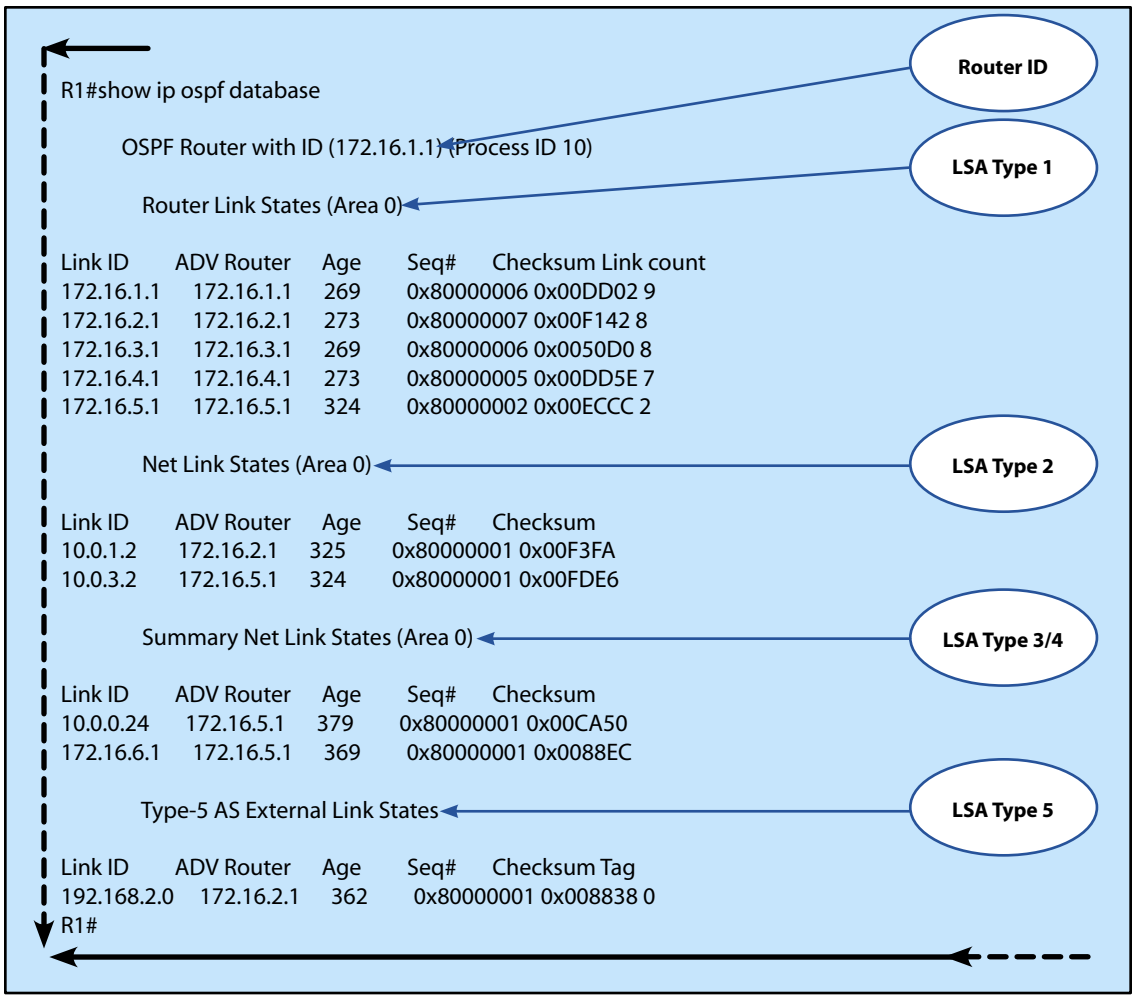
show ip ospf border-routers

This command is used to display the table entries to the ABR and ASBR's.



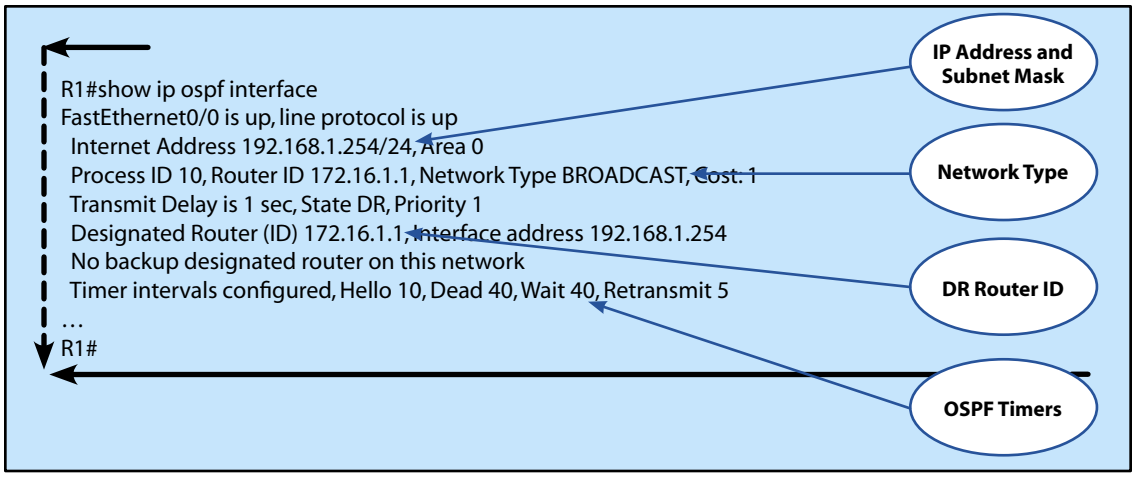
show ip ospf database

This command is used to display information related to the OSPF database.



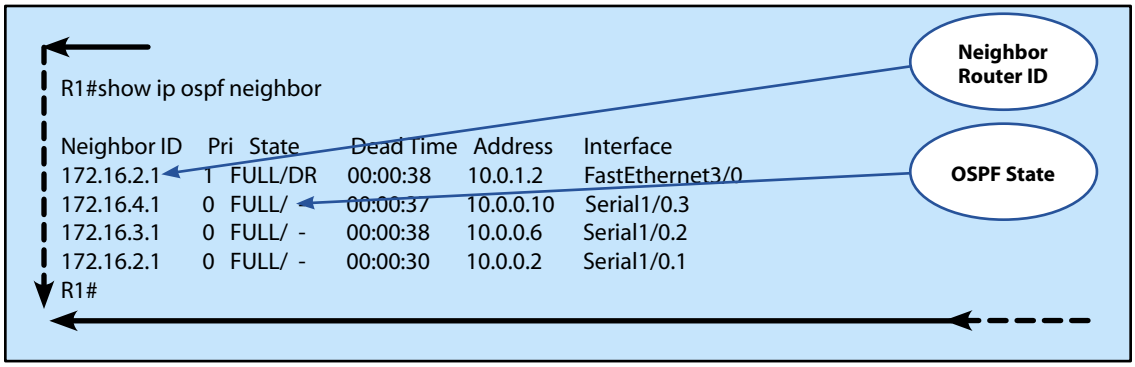
show ip ospf interface

This command is used to display OSPF related interface information.



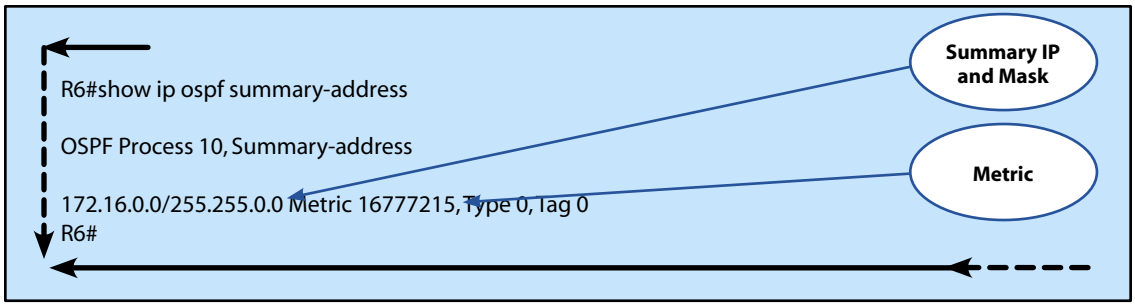
show ip ospf neighbor

This command is used to display OSPF neighbor information on a per interface basis.



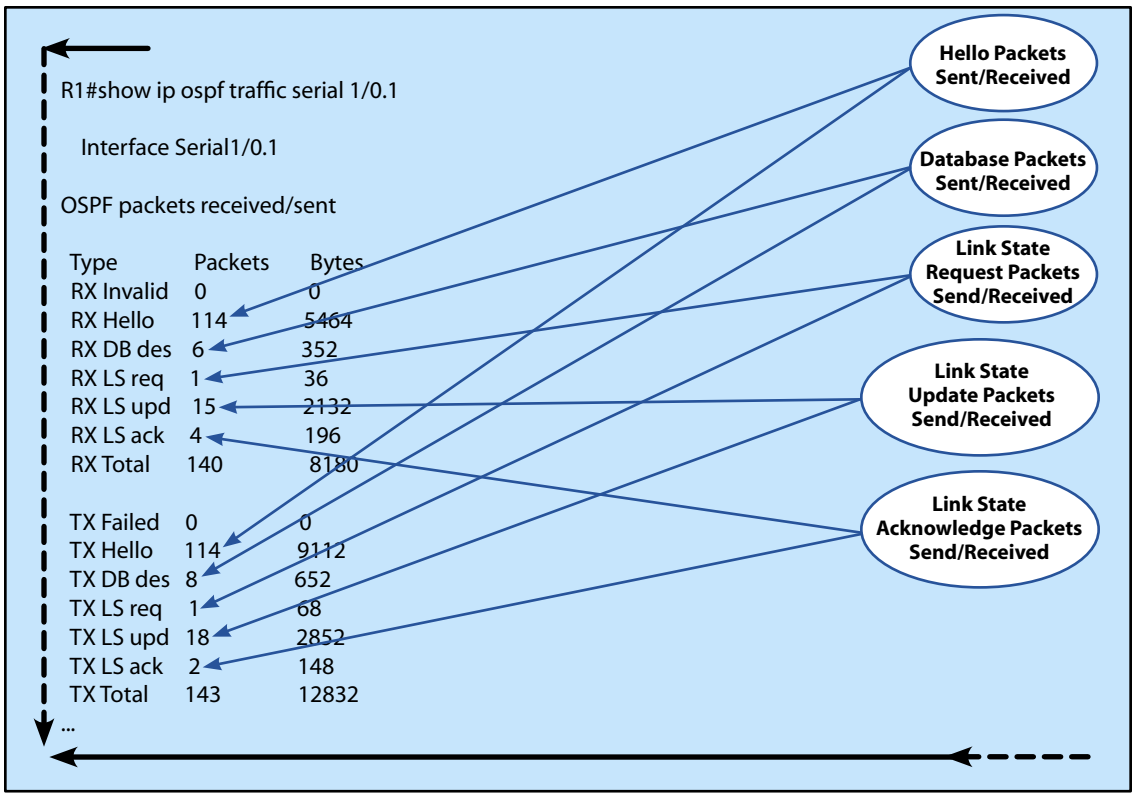
show ip ospf summary-address

This command is used to display all summary address redistribution information.



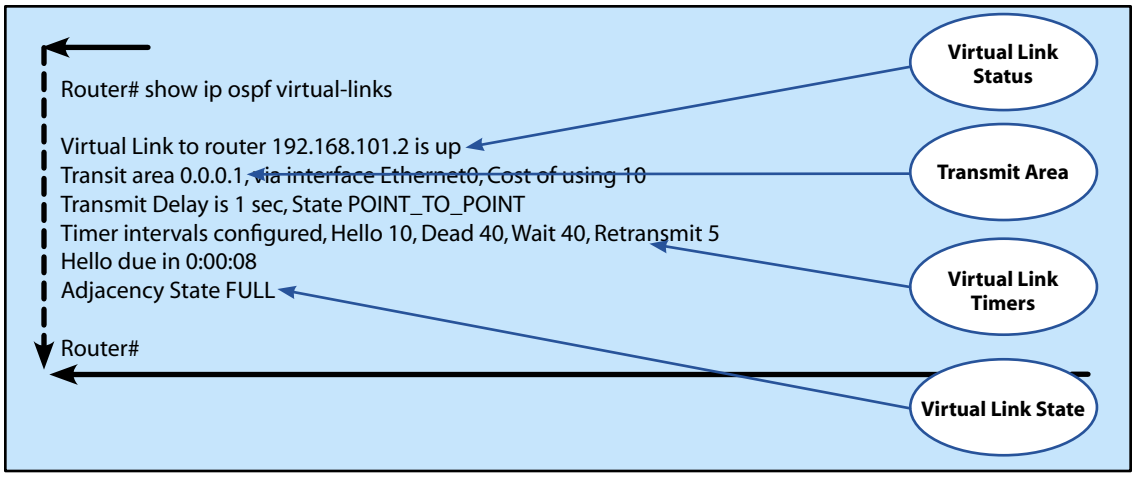
show ip ospf traffic

This command is used to display OSPF traffic statistics.



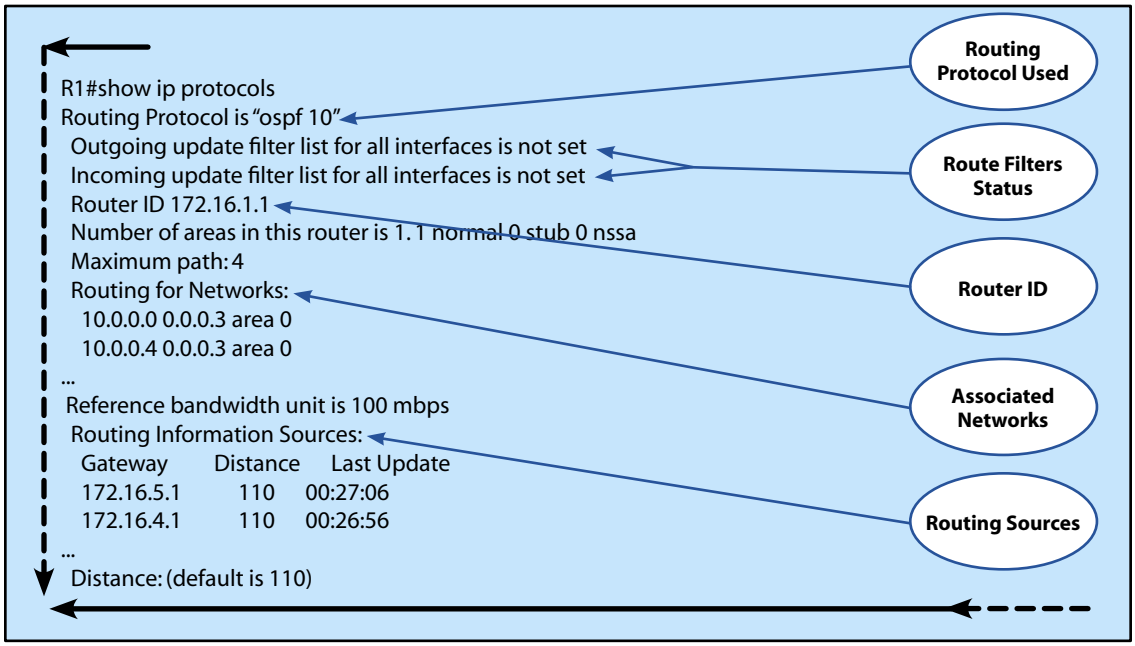
show ip ospf virtual-links

This command is used to display the parameters and the current state of OSPF virtual links.



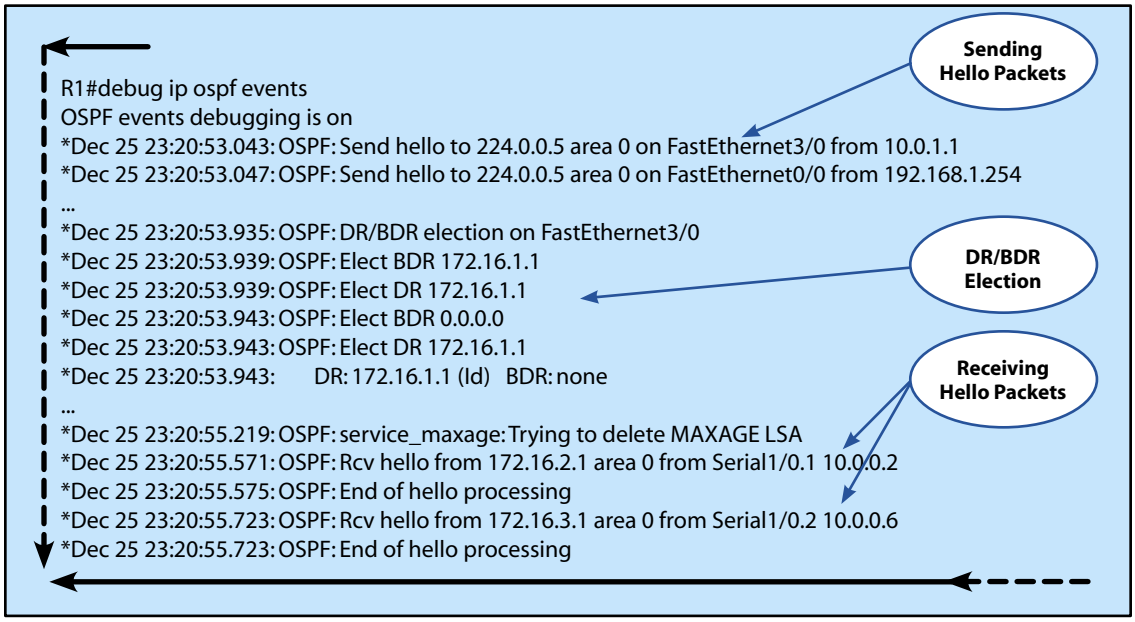
show ip protocols

This command is used to display the parameters and current state of the active routing process.



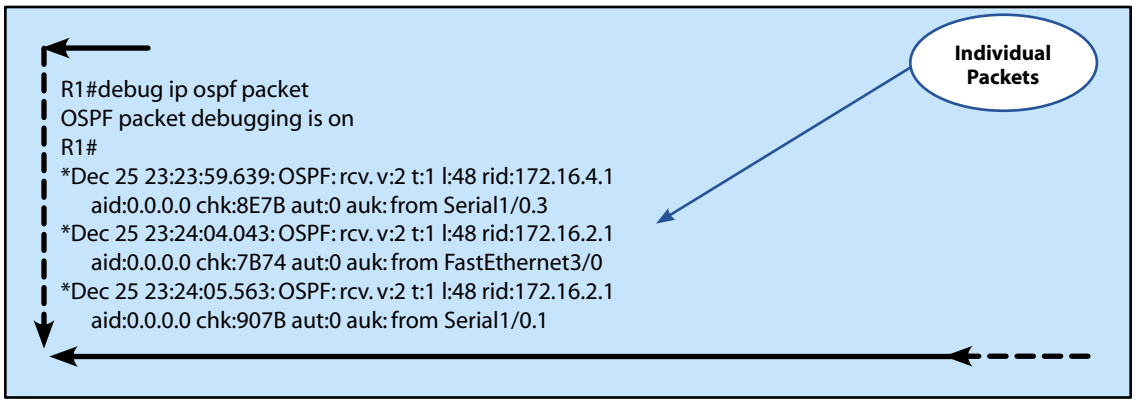
debug ip ospf events

This command is used to display information related to ongoing OSPF events.



debug ip ospf packet

This command is used to display information related to each packet received.



Domain 3 – Describe Integrated IS-IS

IS-IS Overview

The IS-IS routing protocol was originally developed as a routing protocol to compete with OSPF. Because of this, OSPF and IS-IS are very similar. IS-IS was developed as a network layer protocol and uses a Type-Length-Value (TLV) structure. These TLV's are defined to carry different types of information. Originally, IS-IS was built to route only the Connectionless Network Protocol (CLNP), which is the OSI's version of IP. This was extended through RFC1195, which provided support for routing IP over IS-IS and which is also called Integrated IS-IS. TLV 128 defines a structure to support IPv4 information including the Link State Packets (LSP) that are used by IS-IS. There are several different TLV's that are defined to support a number of different types of traffic, but primarily TLV 128 is used for IPv4 traffic. There are a couple of drafts in process that define IPv6 support, specifically TLV 232 and 236.

IS-IS Basics

While IS-IS and OSPF have very similar architectures they do have several differences that must be understood. Because IS-IS was originally built to support only CLNP and not IP, a different type of address structure is required for router identification and for interarea communications. IS-IS uses ISO addresses (49.0001.0000.0000.000a.00) for these communications. ISO addresses are considerably longer then IP addresses but can be simply interpreted and will be explained in detail in the addressing section. Like OSPF, IS-IS makes use of areas but handles area management and communications differently. The IS in IS-IS stands for intermediate-system, which is a term used within the OSI networking model. This model also has end-systems (ES) and other defined protocols. IS-ES is one of these other protocols but is not covered in this manual or in Cisco exams. IS-IS uses two different levels to route. Level 1 routers are used to route within an area and Level 2 routers are used to route towards (between) areas. Level 1 routers route based on the ID portion of the ISO address, while Level 2 routers route based on the Area portion of the ISO address. IS-IS uses the Area portion of the ISO address to distinguish which routers are members of each area. Level 1 and Level 2 areas require a unique ID in order to communicate and establish adjacencies. All routing decisions within IS-IS are done via the SPF (Dijkstra) algorithm like OSPF, but the metrics are manually given per interface with the default being 10. Unlike OSPF, IS-IS routers are not considered in more than one Level-1 or Level-2 area at a time and the demarcation point for area membership is on the wire instead of at the router. Because of this, each individual interface must be placed in a specific area. By default, Cisco routers have a configuration of being L1/L2 routers, meaning that the router is able to communicate with both Level 1 and Level 2 areas. However, before IS-IS routing can start, an interface must be configured to work with IS-IS. All Level-1 routers within an area are aware of routes within the area only. If a Level-2 router is established with a connection to a Level-1 area, all routes from the Level-2 area are not inserted directly into the Level-1 routers route tables. Instead, a default route is inserted that points to the nearest Level-2 router. This Level-1/Level-2 router has a complete table of both the Level-1 network and the Level-2 network and routes accordingly.

Addressing

Addressing within IS-IS is probably the biggest hurdle in learning, as IS-IS is not directly built on IP. As stated before, because IS-IS was built as an OSI protocol, it relies on ISO addressing for router identification and some routing processes. ISO addresses are used in various other technologies like ATM. Another confusing point of ISO addresses is they can be of various lengths. For the purposes of IS-IS, ISO addresses can be broken down into two main parts, the Initial Domain Part (IDP) and the Domain Specific Part (DSP). The IDP is the part of the address that is standardized by the ISO and is that which specifies the format and authority responsible for assigning addresses. The DSP is assigned by the authority responsible specified in the IDP. The IDP is split into two sub-parts, the Authority and Format Identifier (AFI) and the Initial Domain Identifier (IDI). The DSP is split into three sub-parts, the High-order DSP (HO-DSP), System Identifier (ID), and the NSAP Selector (SEL). The IDP (AFI + IDI) and the HO-DSP join to become the Area identifier. The ID must be unique to each router within an area and the SEL within Cisco IS-IS is always set to 00.

IDP		DSP		
AFI	IDI	HO-DSP	ID	SEL

Within ISO, addressing an AFI set to 49 is considered private. Because these addresses typically are not used between external networks, using private addresses is normal in many situations. The main structure that Cisco uses for private IS-IS addresses is 49.0001.0000.0000.000a.00, with 49.0001 being the area, 0000.0000.000a being the ID and 00 being the SEL.

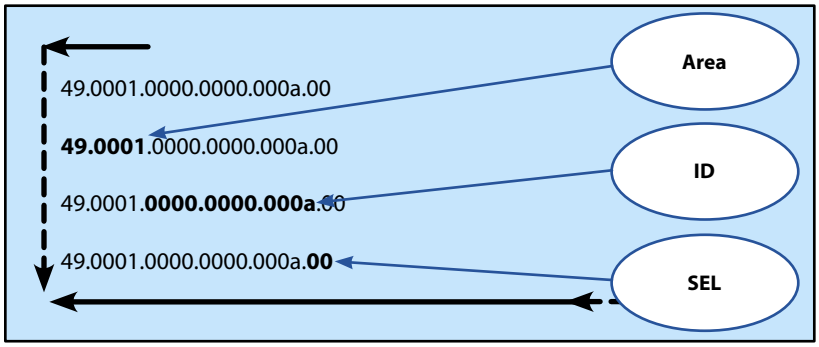


Figure 17 - IS-IS Addressing Example

Another example is using an AFI of 39, which is a Data Country Code (DCC) address used on some public networks. This address 39.0001.1111.0000.0000.000a.00 is longer then the previous example so the area ID is extended in size. For this address, the Area ID is 39.0001.1111, the ID is 0000.0000.000a and the SEL is 00.

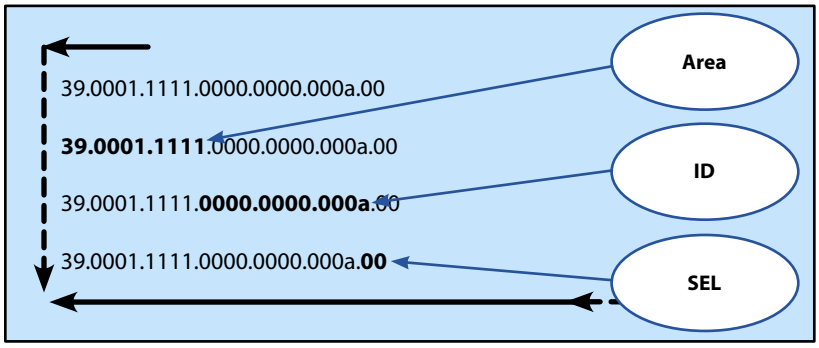


Figure 18 - IS-IS Addressing Example (DCC)

As shown in the following examples, the Area has flexibility in how large it is, but the AFI will always be 1 byte the ID will always be 6 bytes and the SEL will always be 1 byte.

Packet Types

Hello Packets

Like OSPF, IS-IS uses *hello* packets to communicate between routers to establish adjacencies. Within IS-IS, there are a number of different types of *hello* packets. Cisco specifically makes use of four different types of *hello* packet in its implementation. IS-IS handles *hello* packets differently on broadcast and point-to-point networks. Broadcast networks (LAN *hello*) two separate types of *hello* packet. One type of *hello* packet is used for Level-1 communications and one *hello* packet is used for Level-2 communications. On Point-to-point networks (WAN *hello*), two types of IS-IS *hello* packets are used, but for different situations. When a link is initialized, an Intermediate System Hello (ISH) packet is used to make sure there is a router on the other side of the link. Once this has been established an IS-IS *hello* packet is used for all other communications. On these point-to-point links this one *hello* packet is used for both Level-1 and Level-2 communications.

The *hello* packet timers by default are set to 10 seconds on non-DIS (Point-to-Point) networks and set to 3.333 seconds for DIS (Broadcast) networks (⅓ of the "Full" *hello* packet timer). On broadcast networks the *hello* packet timers can be changed for Level-1 and Level-2 areas independently, while on point-to-point networks there is only one *hello* packet timer. By default, the hold timers are set to three times the *hello* packet timer. Both the *hello* packet timers and the hold-time multiplier can be changed.

Link State Packets

Link state packets (LSP) are used to exchange link state information between adjacent routers. There are two different types of LSPs, one of which is used to communicate between Level-1 routers and one of which is used to communicate between Level-2 routers. LSP's transmit a list of all adjacencies known to the transmitting router.

Sequence Number Packets

Sequence number packets (SNP) are used to make sure that each router has the same information and this information has been acknowledged. Because of this, the SNP's can be considered similar to an acknowledge packet. However, IS-IS SNP's are split into four different types of packets that can all be used to verify multiple LSP's at once. These four types are:

Level-1 Complete Sequence Number Packets (CSNP) Level-2 Complete Sequence Number Packets (CSNP)	CSNP's are used to list the sequence numbers to all of the LSP's in the database. There are different CSNP's that are used for Level-1 and Level-2.
Level-1 Partial Sequence Number Packets (PSNP) Level-1 Partial Sequence Number Packets (PSNP)	PSNP's are used to list the sequence number to the last one or more LSP's. There are different PSNP's that are used for Level-1 and Level-2.

Adjacencies

Adjacencies within IS-IS are similar to neighbors within OSPF. These relationships are created to exchange reachability information. Within IS-IS there are two different types of adjacency, one for Level-1 routers and one for Level-2 routers. On all Level-1 routers, this relationship is used to exchange intraarea routing information. On all Level-2 routers, this relationship is used to exchange both intraarea and interarea routing information. Level-2 routers exchange information about which Level-1 networks they attach to and where to route traffic. Level-1 routers are not aware of other Level-1 area networks that are not in their shared area. They route all information destined for other Level-1 areas to the nearest Level-2 router through a default route, which is inserted into their routing tables by the locally attached Level-2 routers. If there is more than one Level-1 area inside a network there must be a router that exists both as a Level-1 and Level-2 router to pass off inter-area traffic. The Level-2 routers are designed to be all interconnected. There is no mechanism within IS-IS to pass Level-2 network traffic through a Level-1 area even if there is a connection possible. By default, Cisco routers are L1/L2 routers.

Certain conditions are required on all network types for an adjacency to establish.

The Maximum Transmission Unit (MTU) must patch.
The Level must be the same between the routers (L1/L2 routers have two separate adjacencies established).
The ID within the network must be unique.
Hello packet timers must match.
Hello packet dead timers must match (*hello* multiplier).

Broadcast Networks

Like on an OSPF network IS-IS limits the amount of router to router traffic through the use of a designated router. Each broadcast network is represented as a *pseudonode*, which has the responsibility to send out Level-1 and Level-2 adjacencies and update information. The physical router which performs the duties of the *pseudonode* is called the Designated Intermediate System (DIS). The DIS is used to create and update LSP's on the broadcast link as well as establish and keep track of adjacencies on the broadcast link. This relationship makes it so that redundant traffic from router to router is limited. Essentially the DIS makes the broadcast network into several virtual point-to-point links from DIS to each router.

Point-to-Point Networks

On a point-to-point link, IS-IS does not need to keep track of multiple routers on the media. Because of this a *pseudonode* is not needed on this type of link.

LSP Propagation

In order to exchange routing information, LSP's must be broadcast between area routers. There are two main things that must be determined. Whether the LSP's are from a Level-1 or Level-2 area and whether the received LSP is valid. All LSP's that are sent out for a specific Level will be ignored by routers that are not on that Level. All LSP's that match the Level and Area are accepted and interpreted. Initially, all LSP's are checked against a Checksum, which is included in the LSP. This checks to see if the LSP was damaged while in transit. After this, sequence numbers are checked in order to see if the information given has been received and inserted into the IS-IS database already. If it is already in the database the LSP is ignored. All LSP's that are entered into the database have a lifetime timer, which is set to refresh at 15 minutes. If an LSP has been in the database for 20 minutes then it is considered expired and the content is purged.

There are some differences in LSP propagation between broadcast and point-to-point network types. On point-to-point network types all LSP's are passed from router to router. On broadcast network types this information is originated to and from the DIS. It is the responsibility of the DIS to keep the *pseudonode* synchronized.

Metric Calculation

With IS-IS metric calculation is considerably easy compared to other protocols. By default, all interfaces regardless of speed are given a metric of 10. Metrics are given based upon the total metric cost across a path. The metric can be changed both by interface and by changing the default.

Route Summarization

Route summarization within IS-IS is very similar to OSPF. Networks can be summarized at the L1/L2 routers and the summary is entered into the Level-2 network only because the Level-1 network does not know about the Level-2 network. There is no Level-1 summarization allowed within the IS-IS protocol. All L1/L2 routers that are within an area must summarize or the router that is not summarizing will get all traffic because of longest match routing.

IS-IS Security

IS-IS has the ability to authenticate adjacencies on each of its routed interfaces. This is done either with a clear-text password or with a key that is used to create an MD5 hash, which is then transmitted with routing traffic. Using an MD5 hash provides for a secure method of authentication by not transmitting the password over the wire.

It is recommended that if you use authentication you use Message Digest Authentication (MD5). MD5 uses key chains and a variable number of keys that are used to create the MD5 hash, which is also called the "message digest". IS-IS uses key chains because they give the ability to utilize multiple keys for send and receives as well as allowing the use of send and receive lifetimes that can be configured to be used and then become inactive after a given amount of time. For configuration information please refer to the IS-IS configuration section in this manual.

IS-IS Configuration

router

The first most important command in configuring IS-IS is **router isis area-tag**, which tells the router that it will be using IS-IS as a routing protocol. The *area-tag* is a meaningful name for the routing process used to identify it, and if omitted a null reference tag is used.

Syntax:

router(config-router)#**router isis** *area-tag*

net

This command is used to assign a network entity title (NET). The NET is used to identify the router inside the IS-IS routing process. A maximum of three NET's can be set per router, although not typical.

Syntax:

router(config-router)#**net** *net*

ip router isis

This command is used to configure an IS-IS process for IP on a specific interface and the ability to attach the interface to a specific area. The *area-tag* is a meaningful name for the routing process that is defined with the **router isis** *area-tag* command. The *area-tag* of the command **ip router isis** must match the defined with the command **router isis** *area-tag*.

Syntax:

router(config-if)#**ip router isis** *area-tag*

isis metric

This command is used to set the metric of a specific interface into IS-IS. The *metric-value* is the specific metric set to the interface. The default is 10. If the **maximum** keyword is used the interface will be excluded from SPF calculations and thus not advertised. The **level-1** and **level-2** keywords are used to set only the metric for Level-1 or Level-2 routes individually.

Syntax:

router(config-if)#**isis metric** {*metric-value* | **maximum**} [**level-1** | **level-2**]

isis hello-interval

This command is used to set the *hello* interval on a specific interface. The *seconds* is the specific period in seconds that is being set for the interface. The default is 10 seconds. If the **minimum** keyword is used the *hello* interval is calculated based on the **isis hello-multiplier** command so that the resulting *hold-time* is equal to 1 second. The **level-1** and **level-2** keywords are used to set only the *hello* interval for level-1 or level-2 routes individually. On DIS interfaces only ⅓ of the *hello* interval is used.

Syntax:

router(config-if)#**isis hello-interval** {*seconds* | **minimal**} [**level-1** | **level-2**]

isis hello-multiplier

This command is used to set the hello multiplier that controls the *hold-time* on an interface. The *multiplier* is a specific number that is multiplied by the configured *hello* interval to reach the *hold-time*. The default is 3. The **level-1** and **level-2** keywords are used to set only the *multiplier* for level-1 or level-2 routes individually.

Syntax:

router(config-if)#**isis hello-multiplier** *multiplier* [**level-1** | **level-2**]

isis csnp-interval

This command is used to set the CSNP interval. The *seconds* is the amount of time between CSNP's on a multiaccss network. This interval only applies to the designated router (DIS). The default is 10 seconds. The **level-1** and **level-2** keywords are used to set only the seconds for level-1 or level-2 routes individually.

Syntax:

router(config-if)#**isis csnp-interval** *seconds* [**level-1** | **level-2**]

isis retransmit-interval

This command is used to set the LSP retransmission interval. The *seconds* is the amount of time between LSP retransmissions. The default is 5 seconds.

Syntax:

router(config-if)#**isis retransmit-interval** *seconds*

isis lsp-interval

This command is used to set the interval between successive LSP transmissions. The *milliseconds* is the amount of time delay between LSP transmissions. The default is 33.

Syntax:

router(config-if)#**isis lsp-interval** *milliseconds*

isis priority

This command is used to set the priority of the designated router. The *number-value* is a number between 1 and 127. The default is 64. The **level-1** and **level-2** keywords are used to set only the priority for level-1 or level-2 individually.

Syntax:

router(config-if)#**isis priority** *number-value* [**level-1** | **level-2**]

isis circuit-type

This command is used to set the type of adjacency that a specific interface can establish. The **level-1** keyword only allows Level-1 type adjacencies, the **level-1-2** keyword allows both Level-1 and Level-2 type adjacencies and the **level-2-only** keyword only allows Level-2 type adjacencies.

Syntax:

router(config-if)#**isis circuit-type** [**level-1** | **level-1-2** | **level-2-only**]

isis password

This command is used to set an authentication password on a specific interface. The *password* is used to set the password used for authentication. The **level-1** and **level-2** keywords are used to set only the password for level-1 or level-2 individually.

Syntax:

router(config-if)#**isis password** *password* [**level-1** | **level-2**]

area-password

This command is used to set the area authentication password. The *password* is used to set the password used for authentication. The **authenticate snp** keyword is used to tell the system to insert the password into sequence number PDU's (SNP's). The **validate** keyword is used to tell the system to insert the password into SNP's and check the password in SNP's received. The **send-only** keyword is used to tell the system to only insert the password into SNP's and not to check the password in SNP's received. Use this command to affect Level-1 routes.

Syntax:

router(config-router)#**area-password** *password* [**authenticate snp** {**validate** | **send-only**}]

domain-password

This command is used to set the domain authentication password. The *password* is used to set the password used for authentication. The **authenticate snp** keyword is used to tell the system to insert the password into sequence number PDU's (SNP's). The **validate** keyword is used to tell the system to insert the password into SNP's and check the password in SNP's received. The **send-only** keyword is used to tell the system to only insert the password into SNP's and not to check the password in SNP's received. Use this command to affect Level-2 routes.

Syntax:

router(config-router)# **domain-password** *password* [**authenticate snp** {**validate** | **send-only**}]

is-type

This command is used to set the routing level for the IS-IS instance. The **level-1** keyword only allows Level-1 type adjacencies, the **level-1-2** keyword allows both Level-1 and Level-2 type adjacencies and the **level-2-only** keyword only allows Level-2 type adjacencies. The difference between this command and the **isis circuit-type** command is this sets the routing level of the system. The **isis circuit-type** command sets what adjacencies are allowed on a specific interface.

Syntax:

router(config-router)#**is-type** [**level-1** | **level-1-2** | **level-2-only**]

metric

This command is used to set the default metric on a router. The *default-value* is used to set the metric value. The default is 10. The **level-1** and **level-2** keywords are used to set only the default metric for level-1 or level-2 individually.

Syntax:

router(config-router)#**metric** *default-value* [**level-1** | **level-2**]

isis protocol shutdown

This command is used to shutdown IS-IS on a specific interface.

Syntax:

router(config-if)#**isis protocol shutdown**

protocol shutdown

This command is used to shutdown IS-IS on a router without losing configuration information.

Syntax:

router(config-router)#**protocol shutdown**

summary-address

In order to configure external summarization, you must utilize the **summary-address** *address mask* {**level-1** | **level-1-2** | **level-2**} [**tag** *tag-number*] [**metric** *metric-value*] command. By using this command, IS-IS is summarized before its networks are advertised externally, limiting the overall amount of information sent to the external neighbor.

Syntax:

router(config-router)# **summary-address** *address mask* {**level-1** | **level-1-2** | **level-2**} [**tag** *tag-number*] [**metric** *metric-value*]

- *address* is the IP network that is going to be summarized.
- *mask* is the subnet mask of the boundary you want to summarize at.

- **level-1** is specified if only Level-1 routes are to be summarized.
- **level-1-2** is specified if only Level-1 and Level-2 routes are to be summarized.
- **level-2** is specified if only Level-2 routes are to be summarized.
- **tag** tag-number is an integer value used to tag a summary route.
- **metric** metric-value is used to specify the metric assigned to the summary route.

lsp-refresh-interval

This command is used to set the LSP refresh interval. The *seconds* is used to set the interval. The default is 900.

Syntax:

router(config-router)#**lsp-refresh-interval** *seconds*

authentication send-only

This command is used to specify that authentication is only to be performed on packets being sent for the whole IS-IS routing process. The **level-1** and **level-2** keywords are used to set authentication for level-1 or level-2 individually.

Syntax:

router(config-router)#**authentication send-only** [**level-1** | **level-2**]

isis authentication send-only

This command is used to specify that authentication is only to be performed on packets being sent for individual interfaces. The **level-1** and **level-2** keywords are used to set authentication for level-1 or level-2 individually.

Syntax:

router(config-if)#**isis authentication send-only** [**level-1** | **level-2**]

authentication mode

This command is used to specify the type of authentication used with the IS-IS routing process. The **md5** keyword is used to set the authentication to use an MD5 hash for exchange. The **text** keyword is used to set the authentication to exchange passwords in the clear as text. The **level-1** and **level-2** keywords are used to set authentication for level-1 or level-2 individually.

Syntax:

router(config-router)#**authentication mode** {**md5** | **text**} [**level-1** | **level-2**]

authentication key-chain

This command is used to enable authentication for the IS-IS routing process. The *name-of-chain* is used to specifiy authentication parameters and must match the **key chain** command. The **level-1** and **level-2** keywords are used to set authentication for level-1 or level-2 individually.

Syntax:

router(config-router)#**authentication key-chain** *name-of-chain* [**level-1** | **level-2**]

key chain

The **key chain** *name-of-chain* command is used to setup a key chain for use with IS-IS authentication. The *name-of-*chain is the name of the key chain and must match the name used in the **authentication key-chain** command above.

Syntax:

router(config)#**key chain** *name-of-chain*

key

The key *key*-id command is used to setup the keys for use with authentication. A number of keys can be used for authentication and used with different key-strings and use timers. The *key*-id is the number given to the key.

Syntax:

router(config-keychain)#**key** *key-id*

key-string

The **key-string** command is used in order to set a passphrase for authentication of other IS-IS routers within the AS. The *text* is the passphrase that will be used on the interface on both sides of the connection.

Syntax:

router(config-keychain-key)#**key-string** *text*

accept-lifetime

The **accept-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*} command is used to control how long a specific key will be considered valid for incoming communications. By default, keys will be accepted for an infinite amount of time. *Start-time* and *end-time* will be entered in the format *hh:mm:ss Month date year*.

Syntax:

router(config-keychain-key)#**accept-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*}

send-lifetime

The **send-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*} command is used to control how long a specific key will be considered valid for outgoing communications. By default, keys will be accepted for an infinite amount of time. *Start-time* and *end-time* will be entered in the format *hh:mm:ss Month date year*.

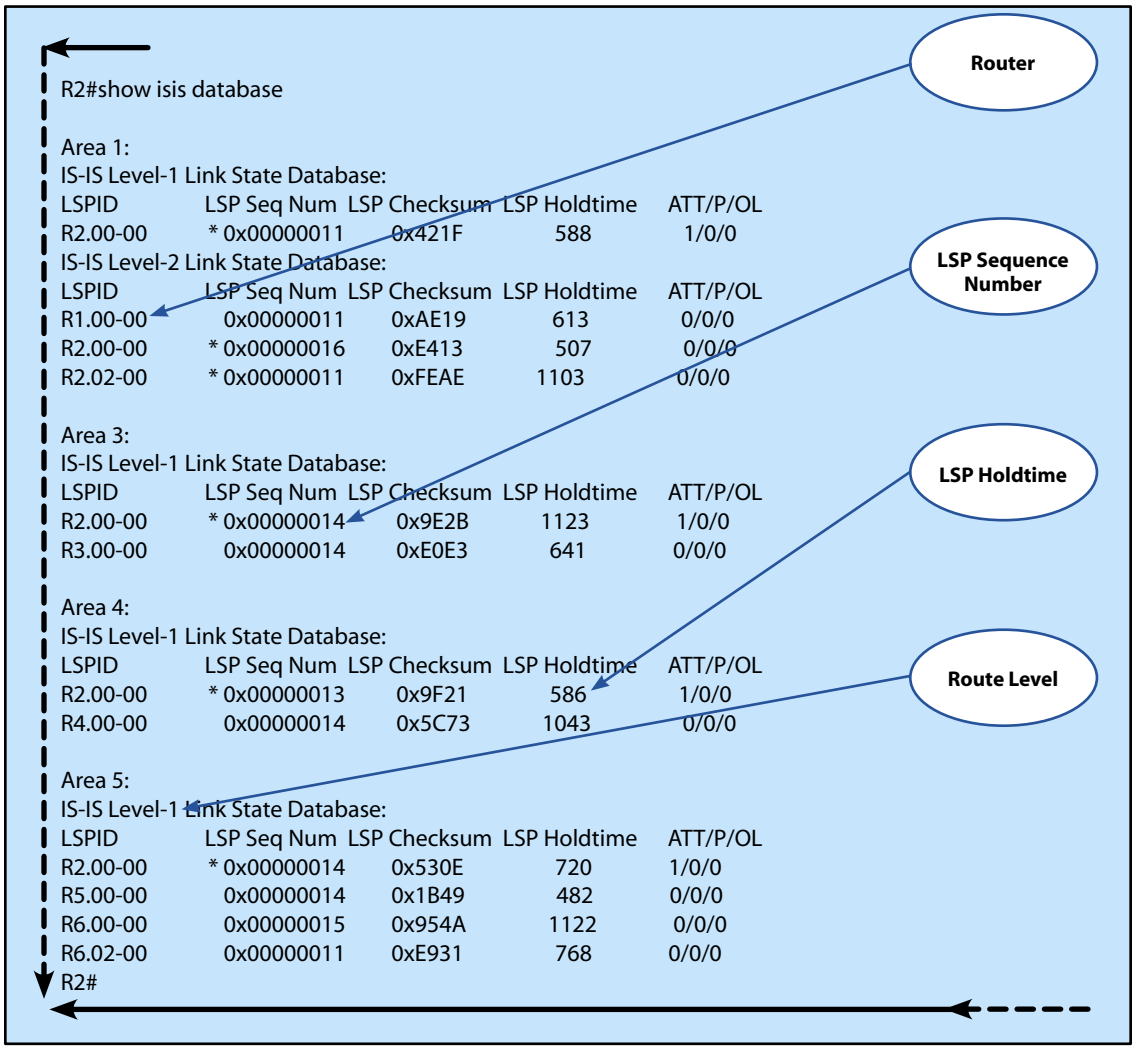
Syntax:

router(config-keychain-key)#**send-lifetime** *start-time* {**infinite** | *end-time* | **duration** *seconds*}

IS-IS Troubleshooting

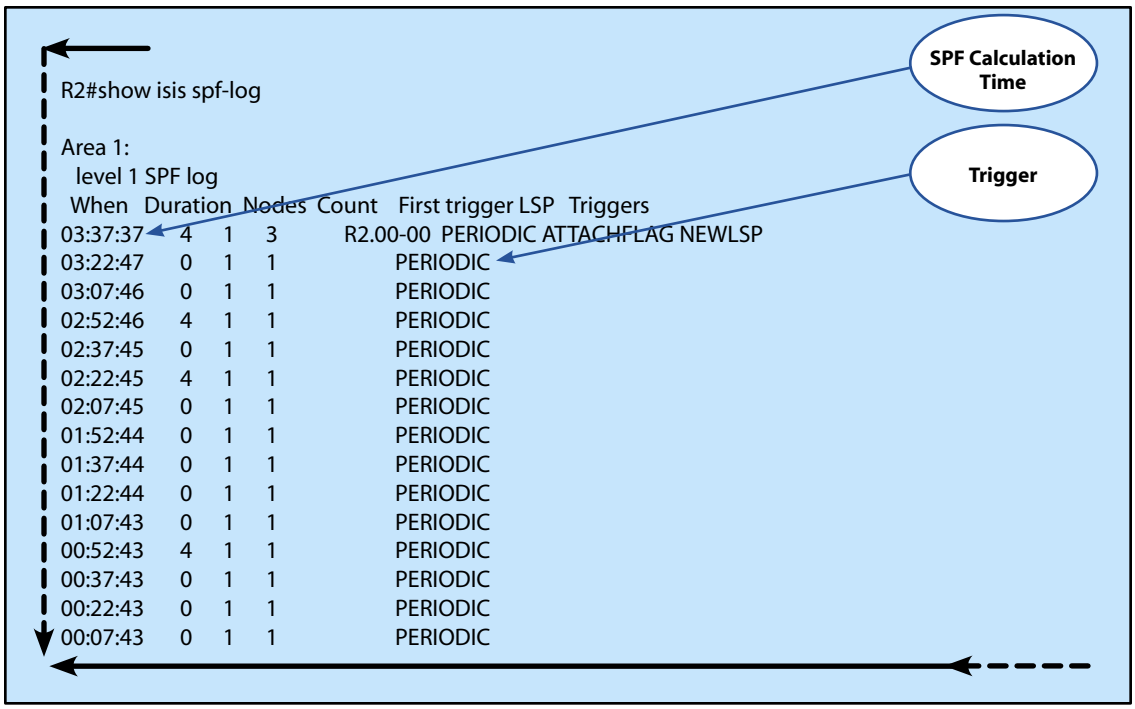
show isis database

This command is used to display information related to the IS-IS database. The following highlights the most important parts.



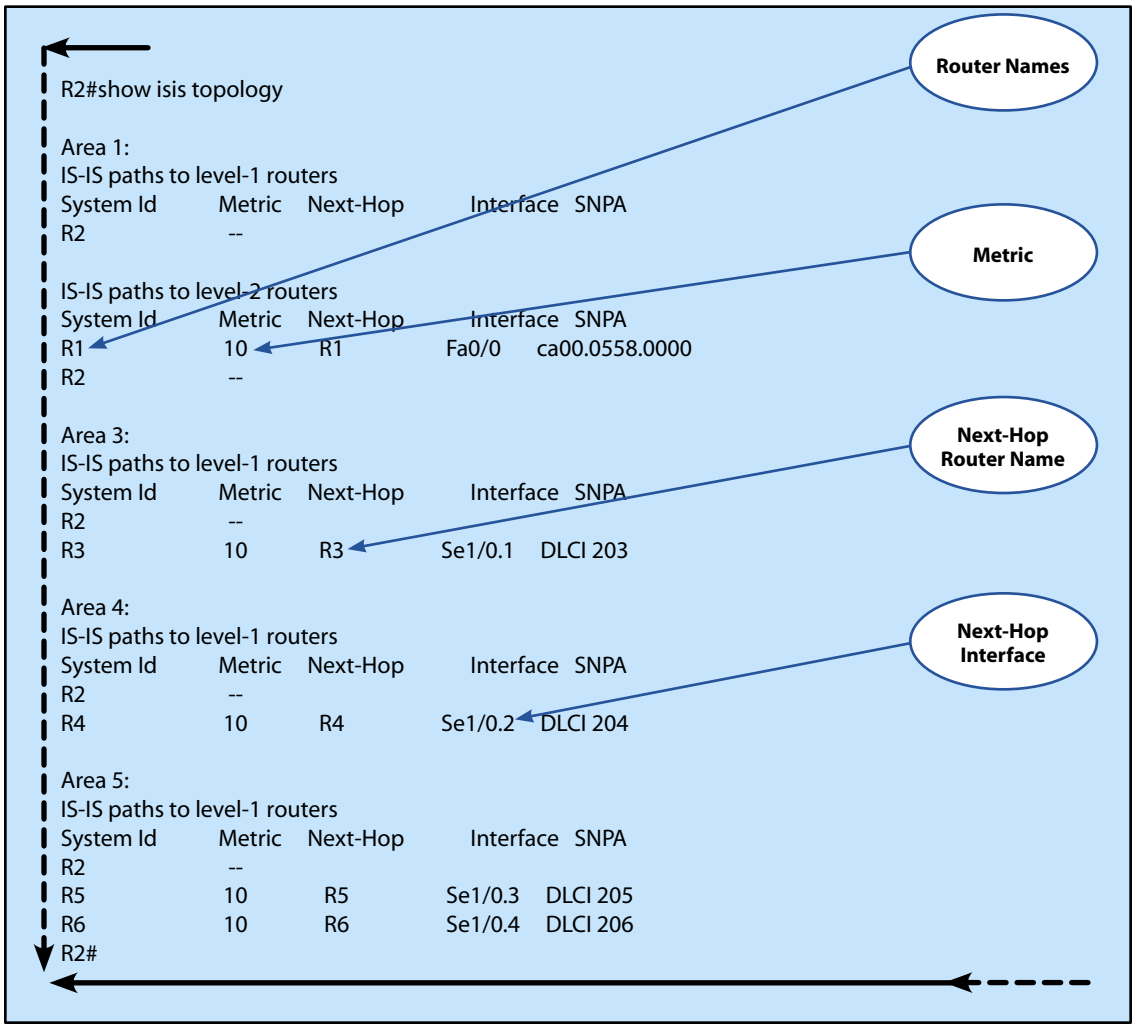
show isis spf-log

This command is used to display information related to the IS-IS SPF calculations. The following highlights the most important parts.



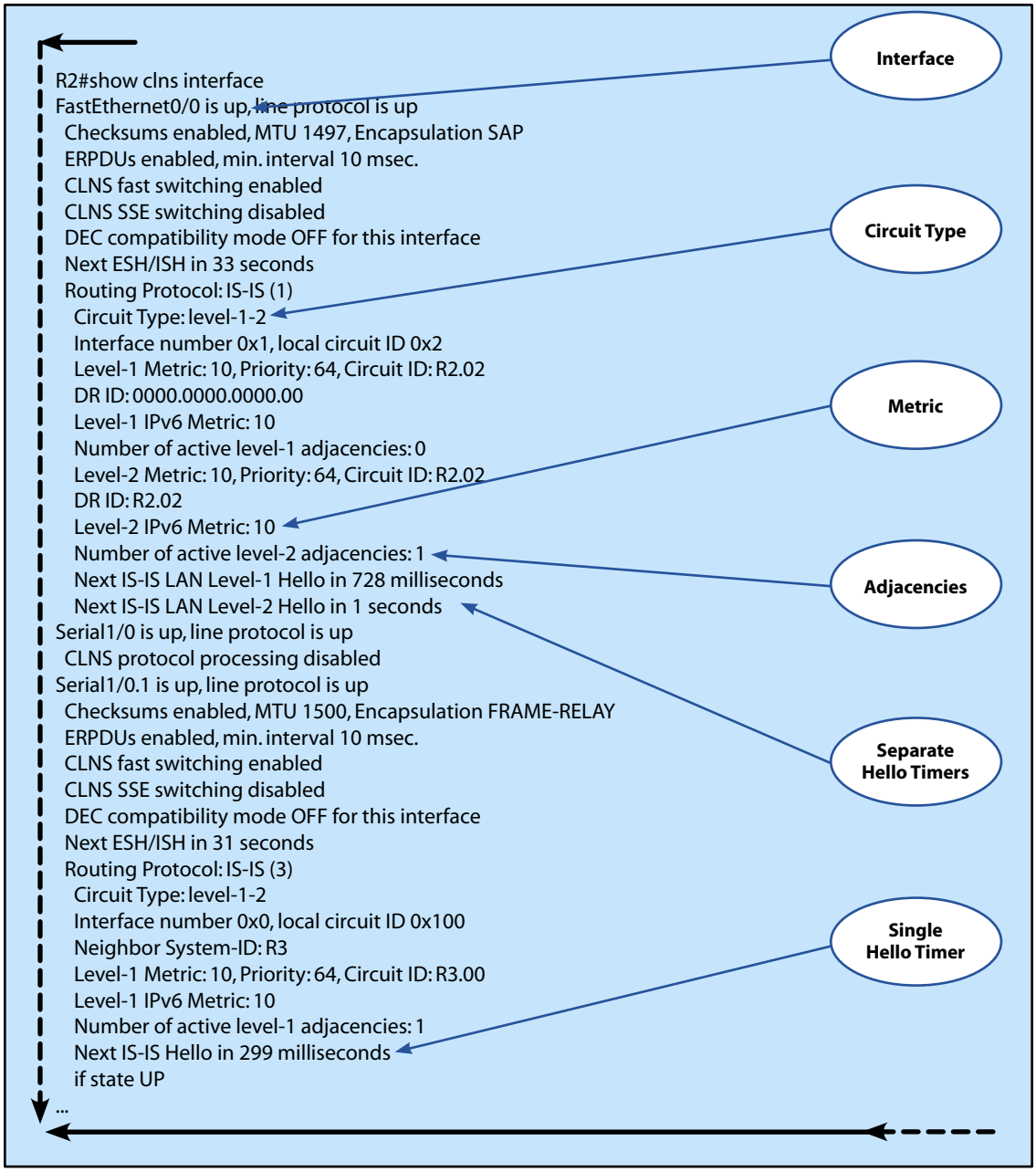
show isis topology

This command is used to display information related to the IS-IS topology. The following highlights the most important parts.



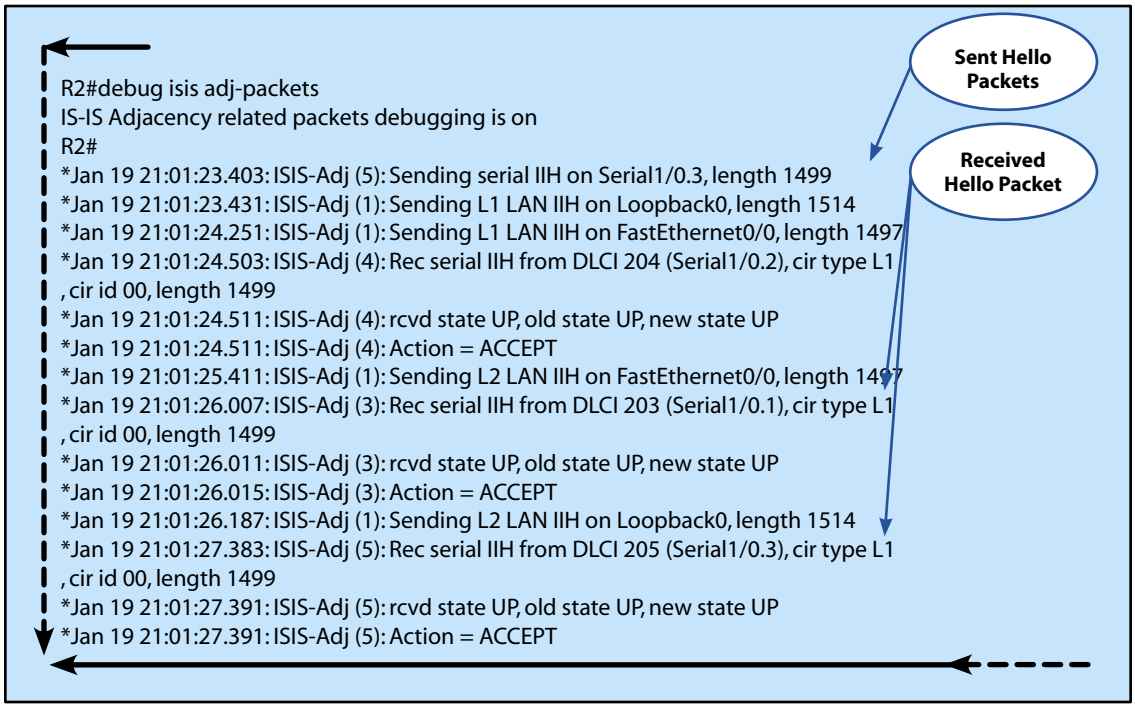
show clns interface

This command is used to display information related to CLNS. The following highlights the most important parts.



debug isis adj-packets

This command is used to debug IS-IS adjacencies through in-depth packet validation. The following highlights the most important parts.



Domain 4 – Implement Cisco IOS Routing Features

Route Maps

Route maps are a very flexible way of controlling a number of operations within a Cisco device. This includes Policy Based Routing (PBR), route redistribution and route filtering, among others. Route maps work on a programming if…then condition. A specific condition is tested; if the condition is matched then the specific set is used. Within route maps this is done through the **match** and the **set** commands. Several **set** commands can be given from one **match** command. The following is a sample of the route map logic:

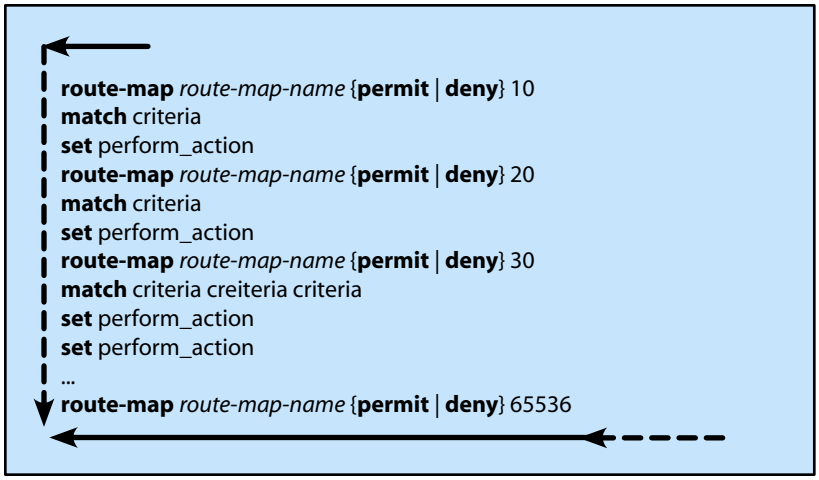


Figure 19 - Route Map Logic

Redistribution

Route redistribution is an important concept to be familiar with when dealing with networks that use multiple routing protocols. Redistribution is the act of taking routes found by one routing protocol and redistributing them into another routing protocol. While the concept is simple, a couple of different types of redistribution exist. One-Way redistribution is the act of redistributing protocol one into protocol two but not redistributing protocol two into protocol one. Two-Way redistribution is the act of redistributing protocol one into protocol two and redistributing protocol two into protocol one.

One-way redistribution is easy to configure and does not have any real caveats; two-way redistribution on the other hand has some caveats that must be considered. In order to avoid routing loops when redistributing routing protocols with higher administrative distance into routing protocols with lower administrative distance, route filtering must be used so the redistributing router does not prefer the newly redistributed route to the original route.

Redistribution allows a lot of flexibility in how routes are distributed. Within the redistribution commands many options exist to control the traffic flow. This is done through the application of route metrics in conjunction with route maps using match and set attributes to define the specific traffic behavior.

When redistributing into OSPF, the subnets options keyword is needed to make sure that all routes which are subnetted are correctly redistributed into the OSPF network.

By default, IS-IS caches all redistributed routes into a local redistribution cache.

Route Filtering

Route filtering is used to limit which routes are distributed from router to router or that are redistributed into another routing protocol. This is done through the use of the **distribute-list** command. The **distribute-list** command is either used for inbound or outbound traffic going out of the router. An optional interface can also be specified onto the command.

Another type of filtering is done through the use of the **passive-interface** command. This command tells the router not to advertise routes out a specific interface. This command does not restrict remote routers from talking with the router through the same interface; it only affects the advertisement of routes out the interface. The **passive-interface** command can also be used with the **default** keyword, which makes all interfaces passive by default.

Policy Based Routing

Policy based routing is used when the route of specific traffic needs to be altered based on a variety of conditions. Under normal routing protocols, the route of traffic is based on the destination of the traffic. For most routing protocols this is the destination network. For BGP, routing is based on the destination AS. Policy based routing gives the option to change the route based on several criteria including protocol, source or destination network, and Quality of Service (QoS) among others. This flexibility is used via route maps. These route maps are configured to match based on a criteria using the **match** command and then the traffic is altered and/or routed along a predetermined path via **set** commands.

DHCP Services

The Dynamic Host Configuration Protocol (DHCP) is used to administer dynamic IP assignment. On Cisco equipment this involves three different potential roles: DHCP server, DHCP client and DHCP relay agent. When acting as a DHCP server, the Cisco equipment is configured with a list of assignable IP addresses that are given to requesting DHCP clients on the interfaces configured. When acting as a DHCP client, the Cisco equipment is configured to request a dynamic IP address from another DHCP server. When acting as a DHCP relay agent, the Cisco equipment is configured to relay DHCP client broadcasts to a DHCP server via unicast.

DHCP uses broadcasts to communicate with clients. Under normal conditions, this means that no DHCP requests or replies traverse the router. If the router is configured to be a relay agent this allows the router to relay these DHCP requests and responses.

Being a DHCP server also gives you the ability to configure different IP options to DHCP clients, which include setting default gateways, Domain Name Server's (DNS), Windows Internet Name Service (WINS) servers, and NetBIOS node type among others.

Configuration

route-map

The **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] command is used to create route-maps for various uses. The *map-tag* is a name used to reference this route-map; this name is used for all statements in the same route-map. The **permit** and **deny** parameters are used to dictate whether the matching traffic will be permitted or denied. The *sequence-number* is used to configure the order that each route-map statement is run.

Syntax:

router(config)#**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]

match interface

The **match interface** *interface-type interface-number* [*... interface-type interface-number*] command is used to match traffic destined to go out a specified interface. The *interface-type* parameters are specifying the matching interfaces.

Syntax:

router(config-route-map)#**match interface** *interface-type interface-number* [*... interface-type interface-number*]

match ip address

The **match ip address** {*access-list-number* | *access-list-name … access-list-number* | *access-list-name}* command is used to match an IP address or IP address range. The *access-list-number or access-list-name* parameter specifies the access list that will be used to specify the IP addresses to match.

Syntax:

router(config-route-map)#**match ip address** {*access-list-number* | *access-list-name … access-list-number* | *access-list-name}*

match ip next-hop

The **match ip next-hop** {*access-list-number* | *access-list-name … access-list-number* | *access-list-name}* command is used to match the next-hop IP address. The *access-list-number or access-list-name* parameter specifies the access list that will be used to specify the IP addresses to match.

Syntax:

router(config-route-map)#**match ip next-hop** {*access-list-number* | *access-list-name … access-list-number* | *access-list-name}*

match metric

The **match metric** *metric-value* [**+-** *deviation-number*] command is used to match a specific protocol metric. The *metric-value* parameter specifies the metric that is to be matched. The *deviation-number* specifies a way to select a range of metrics by selecting a positive and/or negative about of metric from the configured *metric*.

Syntax:

router(config-route-map)#**match metric** *metric-value* [**+-** *deviation-number*]

match route-type

The **match route-type** {**local** | **internal** | **external** [**type-1** | **type-2**] | **level-1** | **level-2**} command is used to match a specific type of route type. The **local** parameter is used to specify locally generated BGP routes. The **internal** parameter is used to specify internal interarea or intraarea OSPF routes as well as internal EIGRP routes. The **external** parameter is used to specify external OSPF or EIGRP routes. The **type-1** and **type-2** parameters are used to specify the specific type of OSPF external route. The **level-1** and **level-2** parameters are used to specify IS-IS Level-1 or Level-2 routes.

Syntax:

router(config-route-map)#match route-type {local | internal | external [type-1 | type-2] | level-1 | level-2}

match tag

The **match tag** *tag-value* [*...tag-value*] command is used to match specific tagged traffic. The *tag-value* specifies the tag value to match.

Syntax:

match tag *tag-value* [*...tag-value*]

set interface

The **set interface** *type number* [*...type number*] command is used to set the interface that matches traffic will go out. The *type* specifies the interface type and the *number* specifies the number of the interface.

Syntax:

router(config-route-map)#**set interface** *type number* [*...type number*]

set level

The **set level** {**level-1** | **level-2** | **level-1-2** | **stub-area** | **backbone**} command is used to specify what level that matched traffic is imported as. The **level-1**, **level-2** and **level-1-2** parameters specify IS-IS routing levels. The **stub-area** and **backbone** parameters specify OSPF routing levels.

Syntax:

router(config-route-map)#set level {level-1 | level-2 | level-1-2 | stub-area | backbone}

set metric-type

The **set metric-type** {**internal** | **external** | **type-1** | **type-2**} command is used to specify what metric type that the matched traffic is imported as. The **internal** parameter is used to specify an IS-IS internal metric or for BGP to import a metric as the MED. The **external** parameter is used to specify an IS-IS external metric. The **type-1** and **type-2** parameters are used to specify OSPF external traffic types either 1 or 2.

Syntax:

```
router(config-route-map)#set metric-type {internal | external | type-1 | type-2}
```

set metric (BGP-OSPF-IS-IS)

The **set metric** *metric-value* command is used to specify a metric that matched traffic will be imported with. The *metric-value* is the metric value that will be imported.

Syntax:

```
router(config-route-map)#set metric metric-value
```

set metric (EIGRP)

The **set metric** *bandwidth delay reliability loading mtu* command is used to specify a metric that matched traffic will be imported with. The *bandwidth*, *delay*, *reliability*, *loading*, and *mtu* are specified as the composite EIGRP metric parameters.

Syntax:

```
router(config-route-map)#set metric bandwidth delay reliability loading mtu
```

set tag

The **set tag** *tag-value* command is used to specify a tag that matched traffic will be imported with. The *tag-value* is a value from 0 to 4294967295.

Syntax:

```
router(config-route-map)#set tag tag-value
```

passive-interface

The **passive-interface** [**default**] *interface-type interface-number* command is used to specify interfaces that routing advertisements will be disabled on, or if the **default** parameter is used, all interfaces do not send advertisements. The *interface-type* specifies the interface type; the *interface-number* specifies the number of the interface.

Syntax:

```
router(config-router)#passive-interface [default] interface-type interface-number
```

distribute-list in

The **distribute-list** [[*access-list-number* | *name*] | [**route-map** *map-tag*]] **in** [*interface-type* | *interface-number*] command is used to configure the filtering of incoming routing updates. The *access-list-number* or *name* specifies an access list that specifies which address ranges are accepted and which are suppressed. The **route-map** *map-tag* specifies a route map that is used to specify which traffic is accepted or suppressed based on various criteria. The *interface-type* specifies the interface type and the *interface-number* specifies the number of the interface.

Syntax:

router(config-router)#**distribute-list** [[*access-list-number* | *name*] | [**route-map** *map-tag*]] **in** [*interface-type* | *interface-number*]

distribute-list out

The **distribute-list** {*access-list-number* | *access-list-name*} **out** [*interface-name* | *routing-process* | *as-number*] command is used to configure the filtering of outgoing routing updates. The *access-list-number* or *access-list-name* specifies an access list that specifies which address ranges are accepted and which are suppressed. The *interface-name* specifies the name of an outgoing interface. The *routing-process* specifies the name of a routing process or **static** or **connected**. The *as-number* specifies an autonomous system number.

Syntax:

router(config-router)#**distribute-list** {*access-list-number* | *access-list-name*} **out** [*interface-name* | *routing-process* | *as-number*]

redistribute (general)

In order to configure redistribution of other routing protocols into OSPF, the **redistribute** *protocol* [*process-id*] {**level-1** | **level-1-2** | **level-2**} [*as-number*] [**metric** {*metric-value* | **transparent**}] [**metric-type** *type-value*] [**match** {**internal** | **external 1** | **external 2**}] [**tag** *tag-value*] [**route-map** *map-tag*] [**subnets**] command must be used.

Syntax(general):

redistribute *protocol* [*process-id*] {**level-1** | **level-1-2** | **level-2**} [*as-number*] [**metric** {*metric-value* | **transparent**}] [**metric-type** *type-value*] [**match** {**internal** | **external 1** | **external 2**}] [**tag** *tag-value*] [**route-map** *map-tag*] [**subnets**]

- *protocol* is the source routing protocol from which routes are being redistributed.
- *process-id* is used if you are redistributing bgp or eigrp in order to determine which process to redistribute.
- **level-1** is used when redistributing IS-IS. This command specifies level 1 IS-IS routes for redistribution.
- **level-1-2** is used when redistributing IS-IS. This command specifies level 1 and level 2 IS-IS routes for redistribution.
- **level-2** is used when redistributing IS-IS. This command specifies level 2 IS-IS routes for redistribution.

- *as-number* is an optional parameter that specifies the AS for the redistributed route.
- **metric** is an optional parameter that allows you to specify the metric of the routes being redistributed into OSPF. If you are redistributing from another OSPF process on the same router, the metrics will be transferred by default. If you are redistributing from another routing process, the default metric is 20.
- **transparent** is an optional parameter that causes RIP to use the routing table metric for redistributed routes.
- **metric-type** is an optional parameter that specifies how routes are redistributed into the routing protocol. For OSPF, you can specify either **1** or **2**; these options specify either E1 or E2 routes. By default, Cisco ios uses E2 routes. For IS-IS, you can specify either **internal** or **external**, with the default being internal.
- **match** is an optional parameter that specifies the criteria OSPF uses to redistribute into other routing protocols. The options are **internal**, **external 1**, and **external 2**.
- **tag** *tag-value* is an optional parameter that allows a value to be attached to each external route. If none is specified, the AS number is used for BGP and EGP and zero is used for all other protocols.
    - *map-tag* is an optional parameter that specifies an identifier for route-maps.
- **subnets** is an optional parameter that specifies whether subnetted routes are redistributed into OSPF.

ip dhcp pool

The **ip dhcp pool** *name* command is used to setup a DHCP address pool. The *name* is either a number or name that identifies the pool.

Syntax:

router(config)#**ip dhcp pool** *name*

network

The **network** *network-number mask* command is used to specify the range of IP address that will be given out by the router. The *network* specifies the IP network that the IP addresses come from; the *mask* specifies the subnet mask. This combination specifies the addresses that are given out.

Syntax:

router(dhcp-config)#**network** *network-number mask*

dns-server

The **dns-server** *address* [*address...address*] command is used to specify the DNS server addresses that are given out to DHCP clients. The *address* is the IP address of the DNS servers. Up to eight IP addresses can be specified.

Syntax:

router(dhcp-config)#**dns-server** *address* [*address...address*]

netbios-name-server

The **netbios-name-server** *address* [*address...address*] command is used to specify the WINS server addresses that are given out to DHCP clients. The *address* is the IP address of the WINS servers. Up to eight IP addresses can be specified.

Syntax:

router(dhcp-config)#**netbios-name-server** *address* [*address...address*]

netbios-node-type

The **netbios-node-type** *type* command is used to specify the NetBIOS node type that is given out to DHCP clients. The *type* parameter specifies the node type. The options are **b-node**, **p-node**, **m-node**, **h-node**.

Syntax:

router(dhcp-config)#**netbios-node-type** *type*

default-router

The **default-router** *address* [*address...address*] command is used to specify the default gateways that are given out to DHCP clients. The *address* is the IP address of the default gateways. Up to eight addresses can be specified.

Syntax:

router(dhcp-config)#**default-router** *address* [*address...address*]

lease

The **lease** {*days* [*hours* [*minutes*]] | **infinite**} command is used to specify the length of time that the DHCP lease will be considered valid. By default, the lease is set to 1 day. The *days*, *hours*, and *minutes* specify the amount of time that the lease is valid. The **infinite** parameter specifies a lease of unlimited length.

Syntax:

router(dhcp-config)#**lease** {*days* [*hours* [*minutes*]] | **infinite**}

domain-name

The **domain-name** *domain* command specifies the domain name that is given out to DHCP clients. The *domain* specifies the hostname that is given out.

Syntax:

router(dhcp-config)#**domain-name** *domain*

ip dhcp excluded-address

The **ip dhcp excluded-address** *low-address* [*high-address*] command is used to specify IP addresses that will never be given out. The *low-address* is the excluded IP address or the first in a range of excluded IP addresses. The *high-address* is the last in a range of excluded IP addresses.

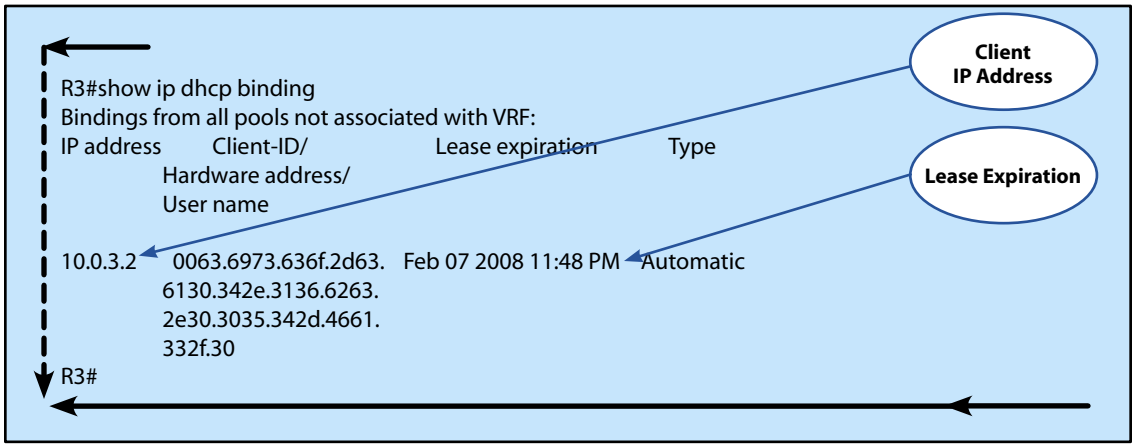
Syntax:

router(config)#**ip dhcp excluded-address** *low-address* [*high-address*]

Troubleshooting

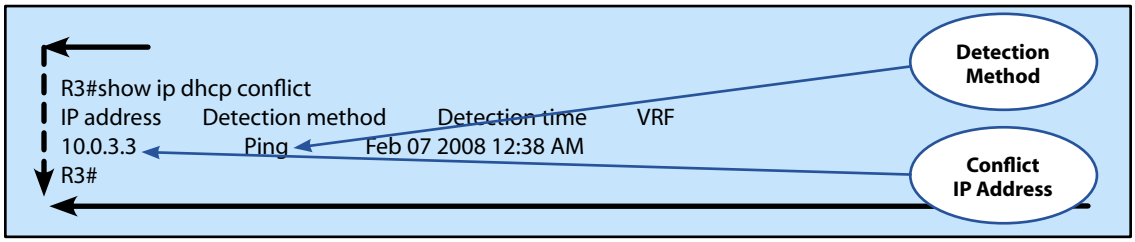
show ip dhcp binding

This command is used to show all of the current IP address assignments; the following highlights the most important parts.



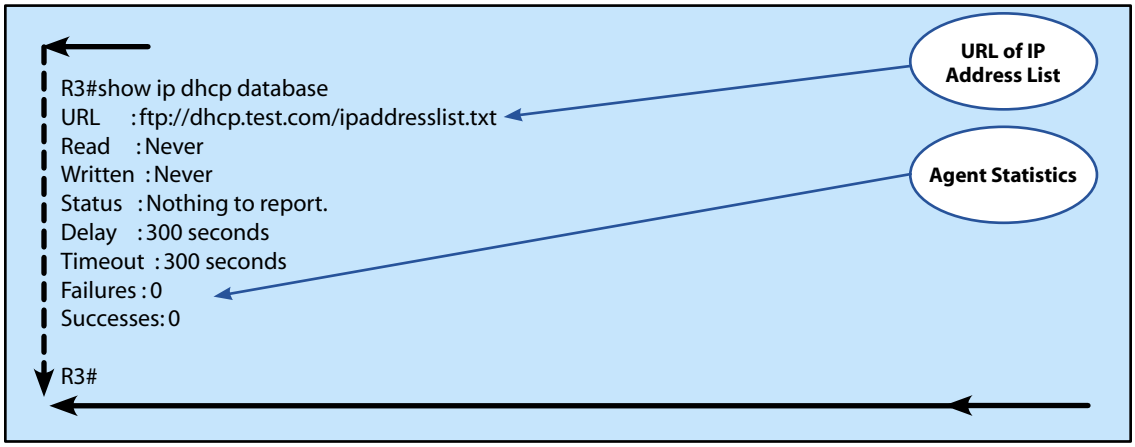
show ip dhcp conflict

This command is used to show if there are any IP address conflicts detected on a network; the following highlights the most important parts.



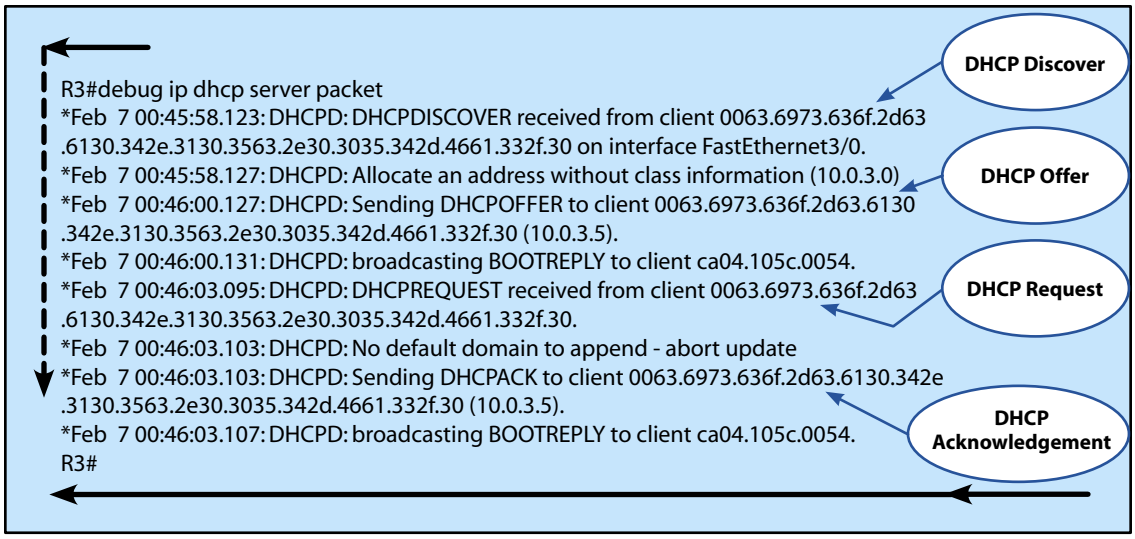
show ip dhcp database

This command is used to show DHCP agent information, statistics and setup; the following highlights the most important parts.



debug ip dhcp server packet

This command is used to debug DHCP server packets; the following highlights the most important parts.



Domain 5 – Implement BGP for Enterprise ISP Connectivity

BGP Overview

In previous domains of this manual, EIGRP, OSPF and IS-IS have been reviewed. These protocols are considered in the networking world to be Interior Gateway Protocols (IGP's). Border Gateway Protocol (BGP) is quite different from these others because it is an Exterior Gateway Protocol (EGP). Now what this essentially means is that IGP's are intended to be controlling routing information inside an autonomous system (AS), whereas an EGP's are intended to be controlling routing information between AS's. This distinction is quite important to understand when studying BGP. Trying to understand the concepts of BGP is considerably more confusing if directly compared to any IGP. BGP specifically keeps a routing table that is computed based on the distance vectors between AS's, not between routers, and it keeps a map of what AS's are required to be traversed in order to get to a specific IP network. A BGP specific application is to be run between large autonomous networks that can be between ISP peers such as Verizon (ISP) and AT&T (ISP) or between two distinct networks within a single company or between a company and an ISP such as Verizon (ISP).

BGP Basics

Terminology

Before we go any further a number of terms must be explained so that learning about BGP is easier to understand. There are a number of terms within BGP that are interchangeable:

Interchangeable BGP Terms		
Peer	Neighbor	Session
Router	Speaker	Link

Term	Explanation
Peer/Neighbor	A peer or neighbor in BGP is a router that has a direct connection and thus routing relationship with another router.
Router/Speaker	A router is also referred to as a speaker within BGP, referencing the physical router.
Session/Link	A Link or Session is a connection between two BGP peers.
Routing Information Base (RIB)	This is the BGP term for its routing table.

Basics

The basics of BGP start with an understanding of the BGP AS's. BGP has two different types of AS's that are defined: the public portion of the AS's, which exist from 1 to 64511 and are assigned by the Internet Assigned Numbers Authority (IANA), and a private portion, which is from 64512 to 65534 that can be used on private networks. The private AS's work just like the public ones only they are not allowed to be routed onto the Internet. BGP also has two different versions that work with each other depending on the situation. External BGP (eBGP) is used to communicate between two different BGP AS's and Internal BGP (iBGP) is used to communicate between BGP routers within an AS.

These two work with slightly different rules between them which can cause confusion. eBGP and iBGP differ in a couple of ways. Routes learned from eBGP can be advertised to a neighbor within iBGP, but routes learned from iBGP cannot be propagated to an iBGP neighbors. A simpler way to say this might be that iBGP routes can only be advertised to their iBGP neighbors, but the neighbors cannot readvertise these routes. This is considered the BGP split horizon rule. This is why iBGP requires a full mesh configuration in order to have a complete iBGP network. Since eBGP links are typically only between two routers, this is not an issue with eBGP. Because a full mesh is required, a large amount of links between routers is required. Now an important thing to point out here is that BGP runs on TCP port 179 and because of this, a full mesh is only virtually required (want to explain why here?), which means that physical links are not required to be in full mesh. While running BGP over TCP offers reliable transport, it also requires a corresponding IGP protocol if any network router in between iBGP routers does not run iBGP directly. As seen in Figure 21, if two routers inside an iBGP network are connected via a non-iBGP participating router, an IGP route between the two is required for the BGP traffic to propagate.

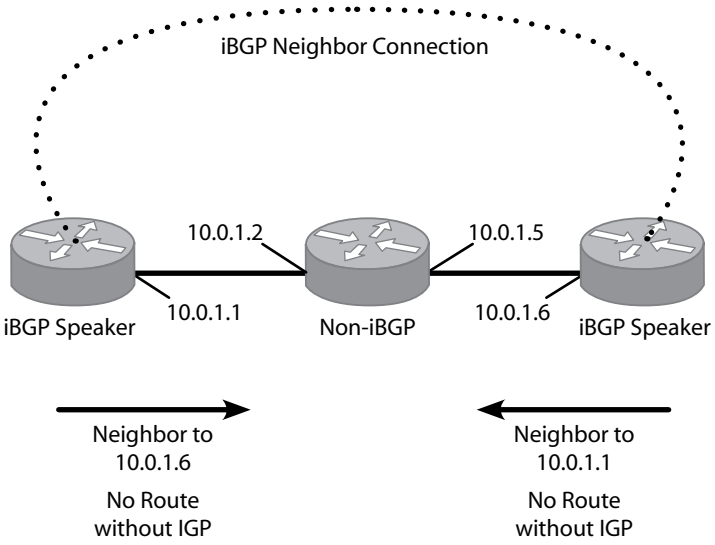


Figure 20 - iBGP IGP Requirements

In this situation where iBGP is not run on all routers, IGP synchronization must be enabled. It should also be noted here that in previous IOS versions (prior to 12.2(8)T) IGP synchronization was enabled by default, but in IOS 12.4 IGP synchronization is disabled by default. What IGP synchronization does is add an additional rule to iBGP. When enabled, iBGP will not insert a route into the routing table unless the IGP also has a route to the destination. This requires BGP to be run over IP if any of the non-iBGP enabled routers don't have a route to the neighboring iBGP router. Then the BGP neighbor relationship will never come up, even though the route within the BGP prefix tables can be seen S(show bgp).

BGP Path Selection

BGP is similar to EIGRP in that it is a hybrid – distance vector protocol. BGP speakers use a mixture of path vectors and attributes to choose a route. The route selection process is as follows:

Step 1	If the path is internal, synchronization is on, and the route is not synchronized. Do not consider it. (If route is not in the IGP routing tables).
Step 2	If the next-hop address of a route is not reachable, do not consider it.
Step 3	Prefer the route with the highest weight. (weight is Cisco proprietary).
Step 4	If multiple routes have the same weight, prefer the route with the highest local preference.
Step 5	If multiple routers have the same local preference, prefer the router that was originated by the local router.
Step 6	If multiple routers have the same local preference, or if no route was originated by the local router, prefer the route with the shortest AS-path.

Route selection process cont'd on next page.

Route selection process cont'd.

Step 7	If the AS-path length is the same, prefer the lowest origin code (IGP < EGP < incomplete).
Step 8	If all origin codes are the same, prefer the path with the lowest Multi-exit-discriminator (MED).
Step 9	If the routes have the same MED, prefer external paths (eBGP) over internal paths (iBGP).
Step 10	If synchronization is disabled and only internal paths remain, prefer the path through the closest IGP neighbor. This means that the router will prefer the shortest internal path with the AS to reach the destination.
Step 11	For eBGP paths, select the oldest route to minimize the effect of routes going up and down.
Step 12	Prefer the route with the lowest neighbor BGP router ID value.
Step 13	If the BGP router ID's are the same, prefer the router with the lowest neighbor IP address.

Table 1 - BGP Route Selection

BGP Message types

Open

The Open message type is used when two BGP speakers start to communicate. This message is acknowledged with a Keepalive message. In order for two BGP speakers to continue to communicate, their versions must match. Once the open has been established, update messages will be exchanged.

Version	8-bit field which includes the BGP version number.
AS	16-bit field which includes the BGP AS.
Hold Time	16-bit field which includes the maximum number of seconds between successive Keepalive or update messages. Upon receiving an Open message, the router selects between the configured hold time and the hold time of the receiving Open message, selecting the smaller of the two.
BGP ID (Router ID)	32-bit field which includes the BGP identifier; on a Cisco router this is the router ID.
Optional Parameters Length	Length of the optional parameters field, if it is present.
Optional Parameters	A variable length field including BGP optional parameters. Currently the only optional parameter defined is for authentication.

Update

The Update message type is used to exchange path information, one path at a time. All of the attributes that are included in the Update message are related to that specific patch only.

Withdrawn Routes Length	Total length of the Withdrawn routes field.
Withdrawn routes	A list of route prefixes which are being withdrawn from the route tables.
Total Path Attribute Length	Total length of the Path Attributes field.
Path Attributes	The path attributes affect the way a path is selected over one another. These will be covered in detail in their own section.
Network Layer Reachability Information	This contains a list of route prefixes which can be reached through this path.

Keepalive

The Keepalive message is sent between BGP neighbors to establish that the neighbor is still reachable. This is by default set to 60 seconds on Cisco equipment, with a default dead timer set to 180 seconds.

Notification

The Notification message is used to send error conditions within BGP. There are a number of errors that can be sent and can be referenced in the RFC.

BGP Neighbor States

This is a very simple description of the BGP neighbor states. Many details are left out for clarity. If you want to know the exact details of each state, reference RFC 1771.

Idle

In the idle state, BGP refuses all incoming connections and no resources are allocated before a BGP start event.

Connect

In the connect state, BGP has been given a start event and is waiting for the transport protocol connection to complete. If the connection succeeds, an Open message is sent and the BGP state transitions to OpenSent. If the connection fails, BGP resets its timers and transitions to the Active state. If the BGP timers elapse, BGP restarts the timers and reinitiates to the BGP peer and stays in the Connect state. In any other event, BGP transitions back to the Idle state.

Active

In the Active state, BGP is actively initiating connection to the peer. If this connection succeeds, an Open message is sent and the BGP state transitions to OpenSent. If the connection fails and the timers expire, the connection is reinitiated and the BGP state transitions to Connect. In any other event BGP transitions back to the Idle state.

OpenSent

In the OpenSent state, BGP is waiting on an Open message type from its neighbor. If an Open message is received in error, a notification message is sent and the state transitions to Idle. If there was no error with the Open message, the hold timers change to the negotiated times and the AS's are compared to determine internal or external neighborship. Once this is complete, the state transitions to OpenConfirm.

OpenConfirm

In the OpenConfirm state, BGP is waiting on a Keepalive or Notification message from its neighbor. If a Keepalive message is received, the state transitions to established. If any Notification message is received, the state transitions to idle. If the hold timer expires, the state transitions to idle. If the Keepalive timer expires, BGP resets the keepalive timer.

Established

In the Established state, BGP can send Update, Keepalive and Notification messages. Under normal operations Update and Keepalive messages are exchanges. If a Notification message is received, the state transitions to Idle. If the hold timer expires, BGP sends a notification message to its neighbor and the state transitions to idle.

BGP Attributes

BGP path attributes are used to relay information between BGP speakers. These attributes are exchanged via Update messages. BGP path attributes are divided into several combinations that determine how these attributes are treated. These combinations are as follows:

Well Known	
Mandatory	Discretionary
Optional	
Transitive	NonTransitive

Well Known attributes are required by the BGP specification to be supported by all routers. Optional attributes are not required to be supported by the BGP specification; optional attributes also can include proprietary attributes. Well Known attributes are all transitive and can be either Mandatory or Discretionary. Well Known Mandatory attributes must be in every update message. Well Known Discretionary attributes do not have to be in every update message. Optional attributes can be either Transitive or NonTransitive. A Transitive attribute is passed from BGP neighbor to neighbor throughout the BGP network, while a NonTransitive attribute is only passed to the initial neighbor and not passed any further.

Cisco implements a number of different attributes that fit into all of these categories. The following attributes are used by Cisco:

Well Known, Mandatory Attributes	
AS-Path	Next-Hop
Origin	
Well Known, Discretionary Attributes	
Local Preference	Atomic aggregate
Optional, Transitive Attributes	
Aggregator	Community
Optional, Nontransitive Attribute	
Multi-exit-discriminator (MED)	

AS-Path

The AS-Path attribute contains the path to the route prefix; this is listed in the attribute in the reverse order. What this means is that whenever a route passes through an AS, that AS number is prepended to the AS-Path. AS numbers are only added to the AS-Path on eBGP routers exiting the AS.

The As-Path attribute is encoded into a sequence of AS-Path segments. There are two different types of segments: AS-SET and AS-SEQUENCE. The AS-SET is typically used when aggregating routes and contains an unordered set of AS's. The AS-SEQUENCE is typically used and contains an ordered set of AS's.

Next-Hop

The Next-Hop attribute contains the next hop IP address to the advertised prefix. For eBGP this is simply the IP address of the eBGP neighbor. For iBGP this is going to be the IP address of eBGP entry router. This is shown in the following figure:

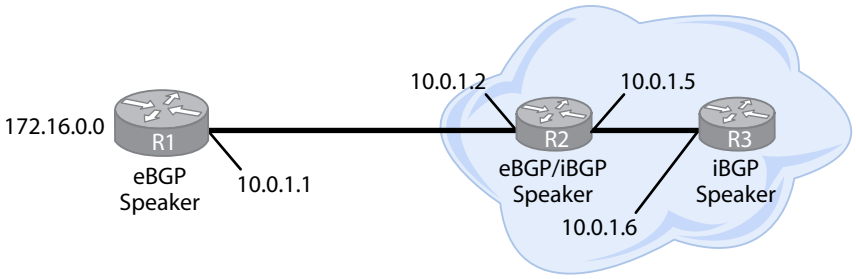


Figure 21 - Next-Hop Example

The next hop IP to the 172.16.0.0 network on R2 is 10.0.1.1. The next hop IP to the 172.16.0.0 network on R3 is also 10.0.1.1.

Origin

The origin attribute defines the origin of the path information. The three options that are in the origin attribute are IGP, EGP and Incomplete. If the origin attribute is IGP, the route originated inside the AS. If the origin attribute is EGP, the route originated outside the AS. If the origin attribute is Incomplete, then the route origin is unknown. This typically happens when the route was redistributed into BGP.

Local Preference

The Local Preference attribute is used to set a preferred BGP exit path. The local preference is set per prefix typically with route maps. The Local in Local Preference means that this preference is only propagated inside the AS and is stripped at the AS boundary. By default, the Local Preference is 100; the higher the preference the better.

Atomic Aggregate

The Atomic aggregate attribute is used to notify the neighbor AS that the routes have been aggregated.

Aggregator

The Aggregator attribute specifies the BGP router ID and AS number of the router that performed the aggregation.

Community

Within BGP the concept of communities is used to group together a number of BGP routers and tag the routes. These tags can be used to filter incoming and outgoing routes or to select preferred routes via the tag. By default, the community is relayed through any non-supporting router. However, it must be configured to be relayed through any supporting router.

Multi-Exit-Discriminator (MED)

The MED is used to try to influence which path external neighbors use to route into the AS. A router will not pay attention to the MED unless the neighboring AS is the same for all routes. By default the MED is 0; the lower the MED the better.

Originator ID

The Originator ID is only used when route reflectors are used. The Originator ID is inserted by the route reflector and carries the router ID of the originator of the route in the local AS. If an update comes back to the originator that sent it, then the update is ignored. If this happens a poor configuration exists, as this should not happen.

Cluster ID

The Cluster ID is only used when route reflectors are used. The Cluster ID is used to identify the route reflectors in a cluster. Typically there is only one route reflector in a cluster and the cluster is identified by the router ID of the route reflector, but if more then one reflector is used for redundancy, then they must be configured with a Cluster ID.

Weight (Cisco Proprietary)

The weight attribute is used on Cisco routers only and is only considered on the local router and is not propagated to any neighbor. The weight attribute is considered per neighbor. By default, the weight of locally originated routes is 32768 and any other path's weight is 0. The higher the weight the better when routes to the same destination exist.

BGP Peers Groups

BGP peer groups are used to group together a number of neighbors, which are configured with the same update policies. By being part of a peer group, a template can be created that will apply to all members of the peer group. Members of a peer group inherit all options from the peer group; however only incoming update policies can be overridden on the local router. Outgoing update policies cannot be overridden.

iBGP Full Mesh Alternatives

iBGP requires a full mesh in order to prevent routing irregularities in the network. If there are a large number of iBGP routers, the number of links required goes up exponentially. In order to work around this requirement, two different, separate solutions have been developed to work so a full mesh is not required. These include route reflection and router confederations. These will be detailed in the following sections.

BGP Route Reflectors

BGP route reflection changes the rules of iBGP to enable the use of a smaller amount of links between AS routers. This is done through the use of route reflectors and route reflector clients. On the route reflectors, the BGP split horizon rule is ignored and they are allowed to pass along iBGP updates to its clients and pass client updates onto other route reflectors in the AS. In order to reduce the amount of links, route reflectors are grouped together with a number of route reflector clients. This is called a cluster. This route reflector is responsible for the updates, incoming and outgoing from its clients and thus doesn't require the Reflector clients to be meshed. The route reflector clients require no special configuration. The route reflector passes these updates to and from the other route reflectors in the AS. A sample configuration is shown below:

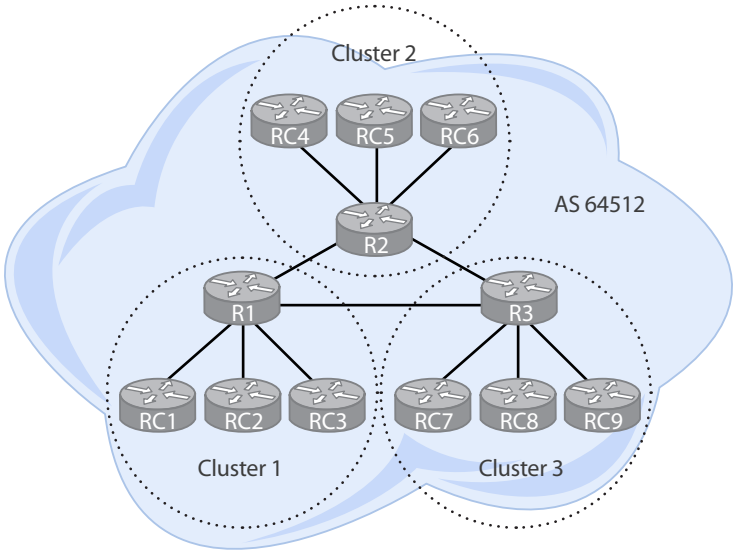


Figure 22 - Route Reflector Example

BGP Confederations

BGP confederations are the second technology that can be used to get around the iBGP full mesh requirement. What confederations do is split an AS up into sub-AS's; the rules inside the sub-AS's are the same as normal AS's. The connections that are established between the sub-AS's are similar to eBGP sessions; sometimes these sessions are notated as EIBGP sessions. The main difference between these sessions and normal eBGP sessions is that a couple of attributes are preserved. These attributes are Next-Hop, MED and Local Preference. Confederations allow a single IGP to be run throughout the AS (and sub-AS's) to propagate next-hop information.

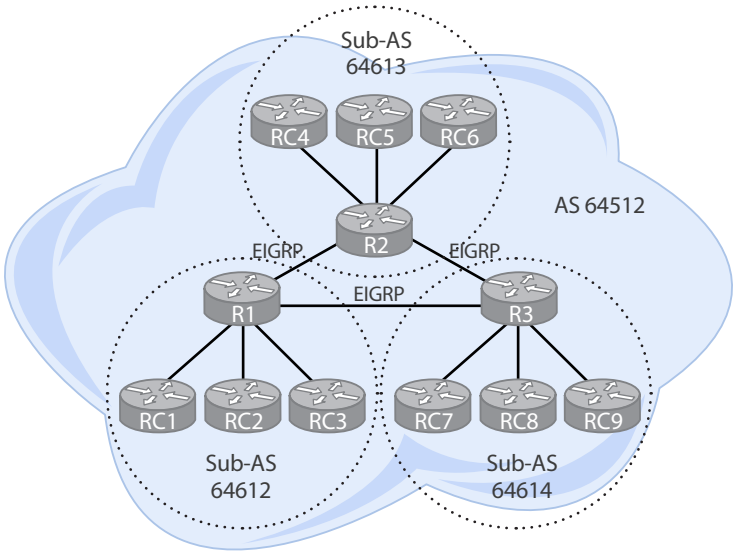


Figure 23 - Route Confederations Example

BGP Security

BGP security is rather easy compared with other routing protocols. Since BGP runs over TCP, it can take advantage of TCP's MD5 built in hash mechanism for security. Simply configure a password for each neighbor connection. This password is turned into an MD5 hash over the wire and authenticated with the neighbor.

BGP Summary (CIDR)

Since BGP is an EGP, summarization is done typically with Classless interdomain routing (CIDR). CIDR enables the ability to summarize an IP address range at a level higher than the class level, sometimes referred to supernetting. A simple example would be summarize several C class addresses with a B class mask, i.e. summarize 172.16.1.0 to 172.16.255.0 by using 172.16.0.0/16.

Within BGP, summarization using CIDR is done through the use of aggregate addresses. There are two main ways to configure aggregate addresses, either through a redistributed static route pointing to interface null0 or through the **aggregate-address** command.

BGP Redistribution

The only type of BGP redistribution which is recommended is the redistribution of static routes into BGP for aggregation. The redistribution of IGP's into BGP is not recommended, although possible.

Route Dampening

BGP has a mechanism that has been put in place to limit the amount of potential route changes due to flapping routes. When a route flaps, the BGP routers are forced to recalculate routes in and out of the routing tables. In order to limit this, route dampening can be enabled. Route dampening allows the router to pay attention to routes that are having problems and issue a penalty to these routes. If the routes are given excessive penalties they are suppressed for an amount of time to allow the route to become stable. For every flap of a route it is given a penalty of 1000 and for each attribute change it is given a penalty of 500.

BGP Configuration

router

The first command is **router bgp** *autonomous-system-number*, which tells the router that it will be using BGP as a routing protocol. The *autonomous-system-number* is a number that identifies the AS this router is in. It is not possible for a Cisco router to be in more than one AS.

Syntax:

router(config)# **router bgp** *autonomous-system-number*

network

The second most important command is **network** {*network-number* [**mask** *network-mask*]}, which tells the router which networks are to be advertised by BGP. The *network-number* is used to denote the network which is to be advertised; typically this is the subnet address of the network (192.168.1.0 for the 192.168.1.0/24 network). The *network-mask* is the subnet mask of the network to be advertised.

Syntax:

router(config-router)#**network** {*network-number* [**mask** *network-mask*]}

neighbor remote-as

The **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*] command is used to setup the neighbors for BGP to communicate with. The *ip-address* is the IP address of the remote router. The *peer-group-name* is used to denote a neighboring peer group. The *autonomous-system-number* is used to denote the remote AS number and the **alternate-as** *autonomous-system-number* is used to denote an alternate AS that can be identified with the neighbor.

Syntax:

router(config-router)# **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*]

synchronization

The **synchronization** command is used to enable IGP synchronization. By default, synchronization is turned off on all IOS versions after 12.2(8)T.

Syntax:

router(config-router)#**synchronization**

clear ip bgp

The **clear ip bgp** {***** | **all** | *autonomous-system-number* | *neighbor-address*} [**in**] [**out**]} [**soft** [**in** | **out**] command is used to reset BGP connections. The * and **all** reset all BGP connections. The *autonomous-system-number* is specified to clear only the connections associated with a specific AS, and the *neighbor-address* is used to clear all connections with a specific neighbor. The **in** and **out** parameters specify resetting either inbound or outbound connections only; by default both are reset. The **soft** parameter specifies a reset without the teardown of the connection.

Syntax:

router# **clear ip bgp** {***** | **all** | *autonomous-system-number* | *neighbor-address*} [**in**] [**out**]} [**soft** [**in** | **out**]

neighbor next-hop-self

The **neighbor** {*ip-address* | *peer-group-name*} **next-hop-self** command is used to configure the router as the next hop for a neighbor or peer-group. The *ip-address* is the IP address of the neighbor and the *peer-group-name* is the neighboring peer group.

Syntax:

router(config-router)#**neighbor** {*ip-address* | *peer-group-name*} **next-hop-self**

neighbor route-reflector-client

The **neighbor** {*ip-address* | *peer-group-name*} **route-reflector-client** command is used to setup a route reflector. This command is used on the route reflector and configures which neighbors that are going to be reflector clients. The *ip-address* is the IP address of the route reflector client neighbor. The *peer-group-name* specifies a route reflector client peer group.

Syntax:

router(config-router)#**neighbor** {*ip-address* | *peer-group-name*} **route-reflector-client**

bgp cluster-id

The **bgp cluster-id** *cluster-id* command is used to specify a BGP cluster ID. The *cluster-id* specifies the BGP cluster ID for route reflectors. A BGP cluster ID should be used when there is more then one route reflector in a cluster. This cluster ID must be the same on all cluster reflectors.

Syntax:

router(config-router)#**bgp cluster-id** *cluster-id*

bgp client-to-client

The **bgp client-to-client reflection** command is used when route reflectors are used and the reflector clients are full meshed. A route reflector setup does not require that clients be meshed; however, if they are, this command can be used so the route reflector does not duplicate routes to the clients.

Syntax:

router(config-router)#**bgp client-to-client reflection**

bgp confederation identifier

The **bgp confederation identifier** *as-number* command is used to configure the BGP confederation identifier. The *as-number* is the confederation identifier. To the outside networks this number represents the main AS.

Syntax:

router(config-router)#**bgp confederation identifier** *as-number*

bgp confederation peers

The **bgp confederation peers** *as-number* [... *as-number*] command is used to configure the sub-AS's that will be part of the confederation. The *as-number* is the sub-AS number's that exist in the confederation.

Syntax:

router(config-router)#**bgp confederation peers** *as-number* [... *as-number*]

neighbor weight

the **neighbor** {*ip-address* | *peer-group-name*} **weight** *number* command is used to assign a weight to a specific neighbor. The *ip-address* is the IP address of the neighbor. The *peer-group-name* is the neighboring peer group. The *weight* is the specific weight given to a neighbor. By default, routes not originating at the current router are given a weight of 0 and routes which are originated on the current router are given a weight of 32768.

Syntax:

router(config-router)#**neighbor** {*ip-address* | *peer-group-name*} **weight** *number*

timers bgp

The **timers bgp** *keepalive holdtime* command is used to configure different values for BGP timers globally. The *keepalive* parameter controls how often BGP sends keepalives to neighbors. By default the keepalive is set to 60 seconds. The *holdtime* parameter is used to control the BGP hold time. By default the hold time is set to 180 seconds.

Syntax:

router(config-router)#**timers bgp** *keepalive holdtime*

neighbor timers

The **timers** [*ip-address* | *peer-group-name*] **timers** *keepalive holdtime* command is used to configure different values for BGP timers per neighbor. The *ip*-address is the IP address of the neighbor and the *peer-group-name* is the name of the neighboring peer group. The *keepalive* parameter controls how often BGP sends keepalives to neighbors. By default the keepalive is set to 60 seconds. The *holdtime* parameter is used to control the BGP hold time. By default the hold time is set to 180 seconds.

Syntax:

router(config-router)#**neighbor** [*ip-address* | *peer-group-name*] **timers** *keepalive holdtime*

bgp default local-preference

The **bgp default local-preference** *number* command is used to configure the default local preference attribute. The *number* is a number from 0 to 4294967295. By default it is set to 100.

Syntax:

router(router-config)#**bgp default local-preference** *number*

neighbor password

The **neighbor** {*ip-address* | *peer-group-name*} **password** *string* command is used to enable the use of an TCP MD5 hash between peers. The *ip-address* is the IP address of the neighbor and the *peer-group-name* is the name of a neighboring peer group. The *string* is the password used for encryption.

Syntax:

router(router-config)#**neighbor** {*ip-address* | *peer-group-name*} **password** *string*

aggregate-address

The **aggregate-address** *address mask* [**as-set**] [**summary-only**] command used to setup an aggregate address for advertisement. The *address* is the IP network aggregate point and the *mask* is the network mask at the aggregate point. The **as-set** parameter is used to make the advertised path an AS-SET instead of an AS-SEQUENCE. The **summary-only** parameter limits the advertisements to only the aggregate route and not to include the more specific routes.

Syntax:

router(config-router)#**aggregate-address** *address mask* [**as-set**] [**summary-only**]

bgp dampening

The **bgp dampening** *half-life reuse suppress max-suppress-time* command is used to enable route dampening. The *half-life* parameter is used to set at what time a penalty for a route is cut in half. By default, this is set to 15 minutes. The *reuse* parameter is used to set at what point the route is unsuppressed. By default, this is set to 750. The *suppress* parameter sets at what point a route is dampened. By default, this is set to 2000. The *max-suppress-time* sets the max amount of time that a route can stay dampened. By default, this is set to 4 times the half-life; however this parameter is set in minutes.

Syntax:

router(config-router)#**bgp dampening** [*half-life reuse suppress max-suppress-time*

network backdoor

The **network** *ip-address* **backdoor** command is used when IGP routes are preferred. Because eBGP routes are given an administrative distance of 20 by default, this never happens. This command is used to change the administrative distance of eBGP routes to 200 (the iBGP value). This enables the IGP routes to have preference. The *ip-address* is the network you want to have the IGP routes preferred.

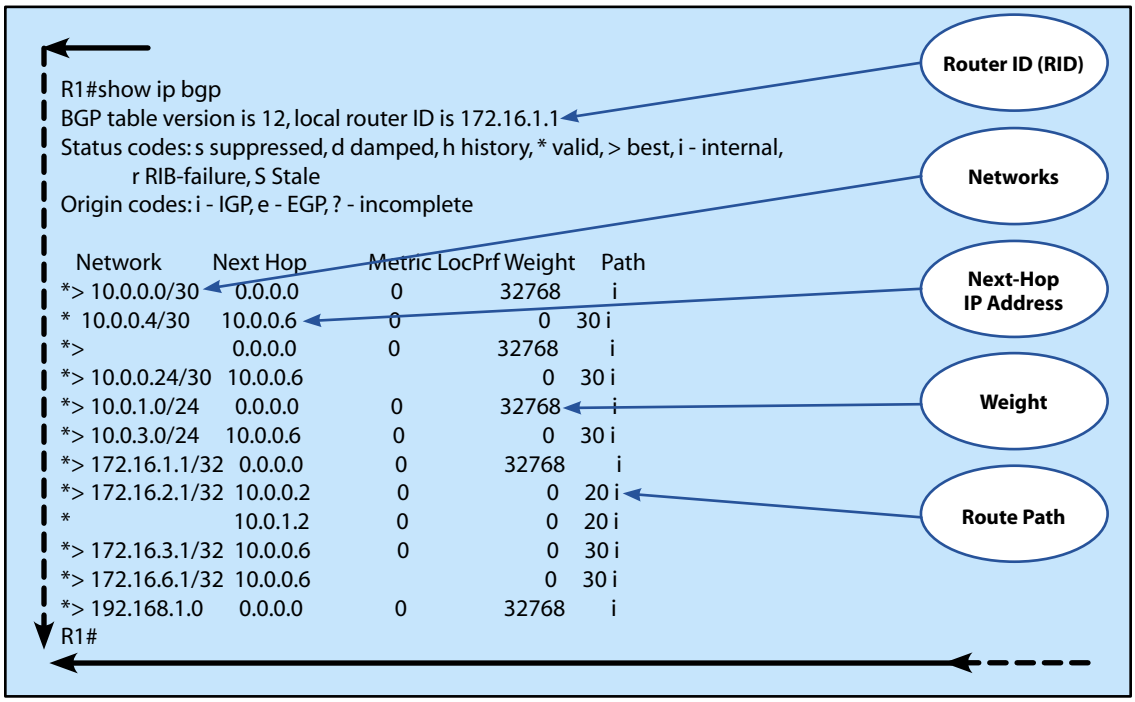
Syntax:

router(config-router)#**network** *ip-address* **backdoor**

BGP Troubleshooting

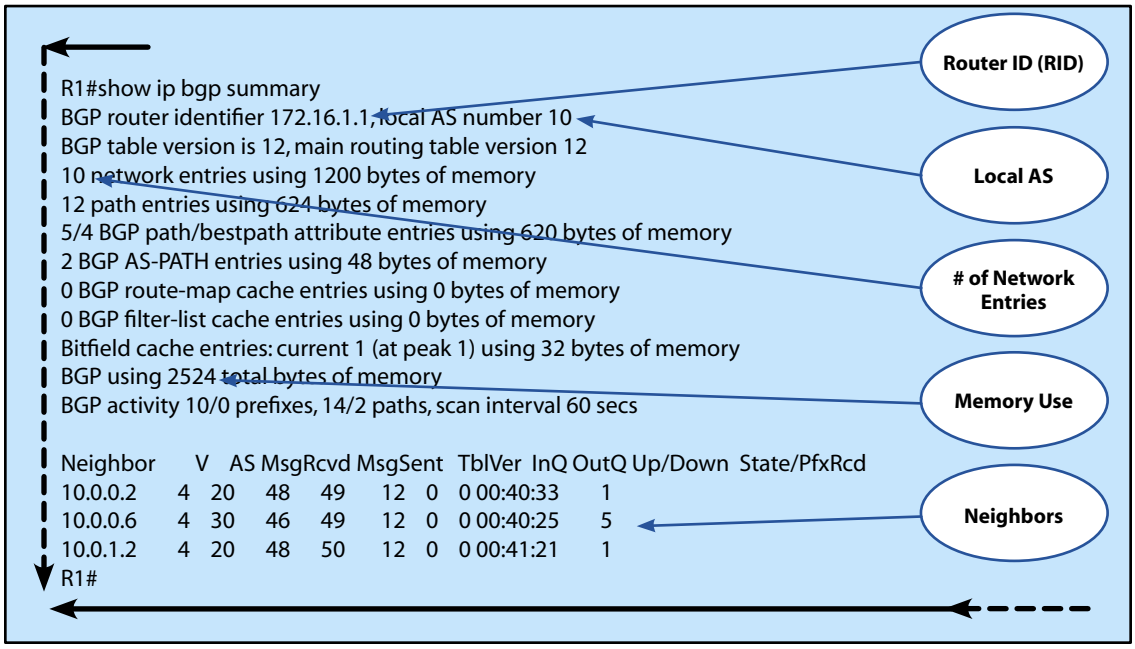
show ip bgp

This command is used to show BGP routable prefixes. The following highlights the most important parts.



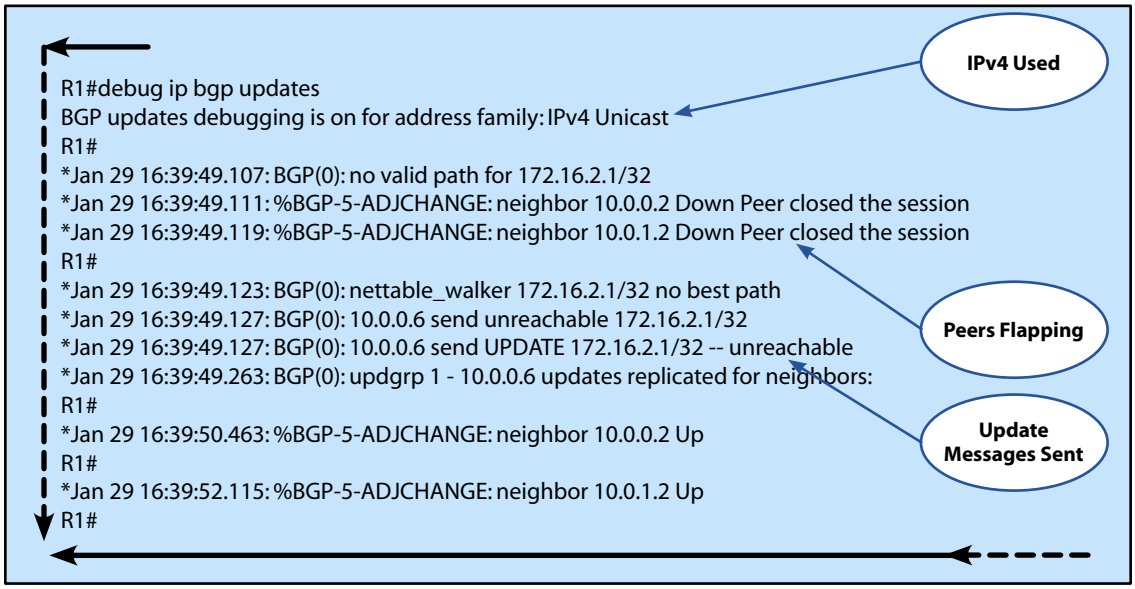
show ip bgp summary

This command is used to show a summary of BGP information. The following highlights the most important parts.



debug ip bgp updates

This command is used to debug BGP adjacencies. The following highlights the most important parts.



Domain 6 – Implement Multicast Forwarding

Multicast Addressing

To begin with, an understanding of multicast addressing and what makes them different from other types of addressing is necessary. Multicast is grouped together within IP networks with two other types of traffic: unicast and broadcast. Unicast traffic is where there is a one to one relationship from host to host or host to server. This is typical for most traffic on the Internet. Broadcast traffic is where a device broadcasts a message to everyone on the local subnet. This works well in LAN environments but it has two problems. Many devices don't need to pay attention to the broadcast traffic but have to, causing extra CPU usage on each host. And broadcast traffic is not routable, so these transmissions are only valid on the local network. Multicast traffic is used when a message is meant for only select hosts on a network; this message can be an audio or video feed among other things.

Multicast traffic is routable and thus can perform some functions similar to unicast and some similar to broadcast. Multicast traffic is targeted at a group multicast IP address. This group is joined by all members wanting to receive the content of the message (or watch the video). Group membership is controlled through Internet Group Management Protocol (IGMP). IGMP is used by listening to host group requests and allowing the routing and switching equipment to know that the host wanted to join or remove itself from a multicast group. This request is processed by the router or switch and the multicast group traffic is forwarded to the joining host.

At layer 3, a block of IP addresses are reserved for multicast, from 224.0.0.0 to 239.255.255.255. This range is split into a couple of groups that dictate what the addresses are used for and where they are used. The following table shows the different multicast address groupings:

Range	Purpose
224.0.0.0/24	Local-local addresses – These addresses are strictly used on the local network and are limited to a TTL of 1.
224.0.1.0 - 231.255.255.255 234.0.0.0 - 238.255.255.255	Globally scoped addresses – These addresses are intended to be used across organizations and the Internet.
232.0.0.0/8	Source-specific multicast – These addresses are used as an extension of multicast where the hosts only receive traffic from a particular server instead of from any server using multicast.
233.0.0.0/8	GLOP addresses – These addresses are allocated to each registered autonomous system (AS). This 16 bit number is used as the second and third octets of the address. An example, 233.127.127.0/24 is for AS 32639. 127 is equal to 7F in hex, since the second and third octet is equal to 127.127 then the AS is equal to 7F7F which is equal to 32639.
239.0.0.0/8	Administrative addresses – These addresses are used for private multicast domains. These addresses are intended to be used only on private networks and are not meant to be routed on the Internet.

Multicast IP Address to MAC Conversion

At level 2, multicast traffic works in the same way but is translated into a MAC address. This translation takes a multicast IP and translates it into a MAC address. This translation is done as follows:

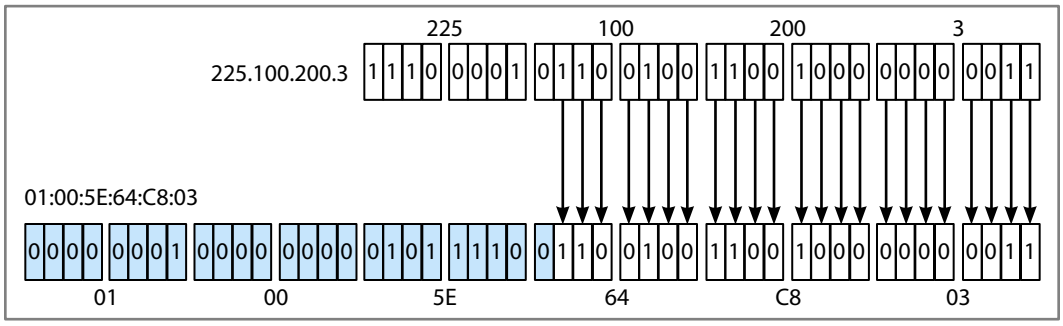


Figure 24 - Multicast IP to MAC

The first 3 bytes of the MAC will always be 0100.5E and the 25th bit will always be 0. The last 23 bits are directly translated from the IP address. Because there are 5-bits of the IP address that are not translated, there are some multicast addresses which have the same MAC address. For every MAC address, it translates to 32 different IP addresses. Caution should be taken to choose multicast addresses that don't overlap.

Since multicast is addresses at layer 2 as well, this enables smart network interface cards to ignore or accept multicast traffic before processing it and thus limiting extra CPU usage.

IGMP (Layer 3)

The Internet Group Management Protocol (IGMP) is used by hosts to identify themselves to routers and to request multicast services.

IGMPv1

IGMPv1 was the initial version of IGMP and allows hosts to join a multicast group. This is done in IGMPv1 via the router (the querier). The router queries the network segment every 60 seconds to look for interested hosts. This query is sent out every 60 seconds on multicast address 224.0.0.1 (All hosts Multicast Group) to verify multicast group membership. Any host can join a group by sending a membership to the router; however, in IGMPv1 there is no multicast group mechanism. Because there is no mechanism for leaving a group, a host gets automatically removed from a group if it does not respond to three consecutive queries.

IGMPv2

IGMPv2 adds some additional features to IGMPv1: this includes group specific queries, a leave group mechanism, a querier mechanism and a query-response timer. Instead of querying the whole group to verify membership requests, IGMPv2 sends requests out to each specific group that it is checking on. This limits the amount of extra traffic processed on a network by hosts that are not group members. IGMPv2 also provides a mechanism for hosts to dynamically leave a multicast group. This is done through the use of a leave group message being sent to 224.0.0.2 (The All Routers Multicast Group). The querier election process allows a querier to be elected based on the highest IP address. The query-response timer provides a timing mechanism for query response. This allows hosts on a group to response anytime in the timer interval.

IGMPv3

IGMPv3 adds another feature onto IGMPv2. This feature is multicast source filtering. What source filtering does is allow a host to request membership in a group **and** request only connection to specific multicast sources. This feature is also called Source Specific Multicast (SSM).

Data-Link Level Support (Layer 2)

At Layer 2, dumb switches handle multicast transmissions like they do broadcast transmissions by sending all multicast traffic out all ports. This quickly can become a large waste of network resources. Smarter switches like the Cisco Catalyst can be programmed to assign multicast groups to specific ports which are interested in multicast group traffic. This can be done in three ways on Cisco switches: through static configuration, through Cisco Group Management Protocol (CGMP) or through IGMP snooping (Preferred).

Static Mappings

Switch ports on Cisco switches can be statically configured to forward multicast group messages. While this is the easiest method for configuring switch ports to multicast groups, it is not at all efficient. For a more efficient method, two dynamic methods for mapping multicast groups to switch ports are available, CGMP and IGMP Snooping.

CGMP

Cisco Group Management Protocol is run on the router and on the switch. The router sends out messages which are meant to be read by the switch to map the specific multicast groups to the needed ports. It can be used in place or with IGMP snooping, but the preferred method is IGMP snooping.

IGMP Snooping

IGMP snooping works differently than CGMP. It is run on the switch. The switch running IGMP snooping listens in on the IGMP messages between the router and the hosts. From this information the switch configures dynamic tables which assign multicast groups to switch ports.

Multicast Routing (PIM)

There are a couple of different multicast routing protocols, but the primary one that will be focused on is Protocol Independent Multicast (PIM). Within multicast routing there are couple of key concepts that need to be covered to understand how routing decisions are made. These include the concept of Reverse Path Forwarding (RPF) and multicast trees.

Like all routing protocols, one of the primary things that needs to be avoided is loops within the routing structure and tables. With multicast routing this is done through RPF. RPF checks every multicast packet that comes into a router. This check takes the source address of the multicast and checks to see which interface the router would take to route *back* to the source. If the interface to route *back* to the source is the same interface the packet was received on, then the router assumes that it has correctly come from the source and not a looped interface and forwards the multicast out all other interfaces based on PIM routing.

Multicast trees are used to show a multicast stream as a tree. Within multicast there are two types of tree: a shared tree and a shortest path tree, also called a source tree. The shared tree provides a specific route through the network where all multicast traffic goes by default. This route may not be the most efficient but it is predetermined. This route is denoted as (*,G), whi h means from any source to a specific group.

The second type of tree is a source tree. A source tree uses the most efficient path through the network and is denoted as (S,G), which means from a specific source to a specific group.

PIM has two main modes that are used for routing. These two work in opposite ways from each other. Which type is used is based on the situation and topology of the multicast network. These two types are dense mode and sparse mode. Cisco has also developed a third hybrid of the two called sparse-dense mode.

Dense Mode

Dense mode (PIM-DM) assumes that there is a large amount of hosts listening to the multicast groups. Because of this assumption, dense mode defaults by sending to every dense mode router on the network. The routers that find that none of the hosts they are attached to are interested in multicast group traffic send a prune message back to the source. This default configuration does create a window of unneeded traffic being sent to the multicast routers on the network, but the key point is that dense mode is used when there are more interested parties than not interested. In dense mode networks, the multicast tree is built from the source to the hosts.

Sparse Mode

Sparse mode (PIM-SM) assumes that there is no one interested in listening to the multicast groups until the hosts request it. Interested hosts send join messages via IGMP to their router. Their router locates the PIM-SM root router. Within PIM-SM there is a central router that serves also as a root router within PIM-SM; this router is also called a rendezvous point (RP). Until traffic flows between the multicast source and the interested host, the tree structure is shared. All setup traffic is routed through the RP, however once the traffic starts between multicast source and host, the tree changes to the shortest path tree model. Sparse mode is intended where network bandwidth is minimal and needed for more than multicast traffic. Because of this, sparse mode should be used in this situation.

Sparse-Dense Mode

Sparse-Dense mode is a Cisco-created hybrid of both dense and sparse modes. Since each multicast group can be chosen to be either dense or sparse, Cisco has accommodated both by allowing traffic from both Dense and Sparse mode sources. Once multicast group traffic is received on an interface, the router checks to see if it has a configured RP. If it does not, then it will configure the group into dense mode. If the multicast group has a configured RP, then the group is configured as in sparse mode.

Rendezvous Points

With Cisco, there are three main ways to setup a rendezvous point on your network. This can be done statically; the RP must be configured and all other routers must be configured to point to the RP router. The second way is Cisco specific and uses Auto-RP. Auto RP works by using two specific multicast addresses, 224.0.1.39 and 224.0.1.40. Within Auto-RP a mapping agent is configured which is centrally located in your network. Each candidate RP router is configured. The mapping agent receives candidate RP announcements via 224.0.1.39; with this information the mapping agent sends RP-Group mapping information to all PIM routers. The third way is only valid with IGMPv2 and above and is called the bootstrap router method; this method works similarly to Auto-RP but is standards based.

Configuration

ip multicast-routing

The **ip multicast-routing** command is used to enable multicast routing on a router.

Syntax:

```
router(config)#ip multicast-routing
```

ip pim

The **ip pim** {**sparse-mode** | **sparse-dense-mode** | **dense-mode}** command is used to configure PIM on an interface. The **sparse-mode** parameter is used to enable PIM-SM on the interface. The **sparse-dense-mode** parameter is used to enable Cisco sparse-dense mode on an interface. The **dense-mode** parameter is used to enable PIM-DM on the interface.

Syntax:

```
router(config-if)#ip pim {sparse-mode | sparse-dense-mode | dense-mode}
```

ip pim version

The **ip pim version** [**1** | **2**] command is used to configure the PIM version per interface.

Syntax:

```
router(config-if)#ip pim version [1 | 2]
```

ip pim rp-address

The **ip pim rp-address** *rp-address* [*access-list*] [**override**] command is used to configure the RP and the multicast groups it takes care of. The *rp-address* is the IP address of the RP. The *access-list* parameter is the number or name of an access list that lists the multicast groups which the use this RP. The **override** parameter is used if a static RP is also used with Auto-RP. This parameter makes the static RP have priority over the RP learned through Auto-RP.

Syntax:

```
router(config)#ip pim rp-address rp-address [access-list] [override]
```

ip pim send-rp-discovery scope

The **ip pim send-rp-discovery** [*interface-type interface-number*] **scope** *ttl-value* command is used to configure the PIM-SM mapping agent for Auto-RP. The *interface-type* parameter is used to define the interface type; the *interface-number* parameter is used to define the interface number. The **scope** *ttl-value* parameter specifies the TTL value Auto-RP discovery messages.

Syntax:

```
router(config)#ip pim send-rp-discovery [interface-type interface-number] scope ttl-value
```

ip pim send-rp-announce

The **ip pim send-rp-announce** *interface-type interface-number* **scope** *ttl-value* [**group-list** *access-list*] [**interval** *seconds*] command is used to configure a router as a candidate RP for Auto-RP. The *interface-type* parameter is used to define the interface type; the *interface-number* parameter is used to define the interface number. The **scope** *ttl-value* parameter specifies the TTL value Auto-RP announcements. The **group-list** *access-list* parameter is used to define the multicast groups that are associated with this candidate RP. The **interval** *seconds* parameter defines the amount of time between announcements. The default is 60 seconds.

Syntax:

router(config)#**ip pim send-rp-announce** *interface-type interface-number* **scope** *ttl-value* [**group-list** *access-list*] [**interval** *seconds*]

ip pim bsr-candidate

The **ip pim bsr-candidate** *interface-type interface-number* [*hash-mask-length*] [*priority*] command is used to configure a bootstrap router. The *interface-type* parameter is used to define the interface type, and the *interface-number* parameter is used to define the interface number. The *hash-mask-length* parameter is used to specify which multicast groups are covered by this BSR. This works similarly to a subnet mask. The numbers of the multicast group that are masked will determine the multicast groups covered by this router. The *priority* parameter configures the BSR with a priority. The default is 0.

Syntax:

router(config)#**ip pim bsr-candidate** *interface-type interface-number* [*hash-mask-length*] [*priority*]

ip pim rp-candidate

The **ip pim rp-candidate** *interface-type interface-number* [**group-list** *access-list*] [**interval** *seconds*] [**priority** *value*] command is used to configure a router as an RP candidate. The *interface-type* parameter is used to define the interface type. The *interface-number* parameter is used to define the interface number. The **group-list** *access-list* parameter is used to define the multicast groups that are associated with this candidate RP. The **interval** *seconds* parameter defines the amount of time between announcements. The default is 60 seconds. The **priority** *value* parameter configures the BSR with a priority. The default is 0.

Syntax:

router(config)#**ip pim rp-candidate** *interface-type interface-number* [**group-list** *access-list*] [**interval** *seconds*] [**priority** *value*]

ip pim border

The **ip pim bsr-border** command is used to specify an interface that BSR messages will not go out.

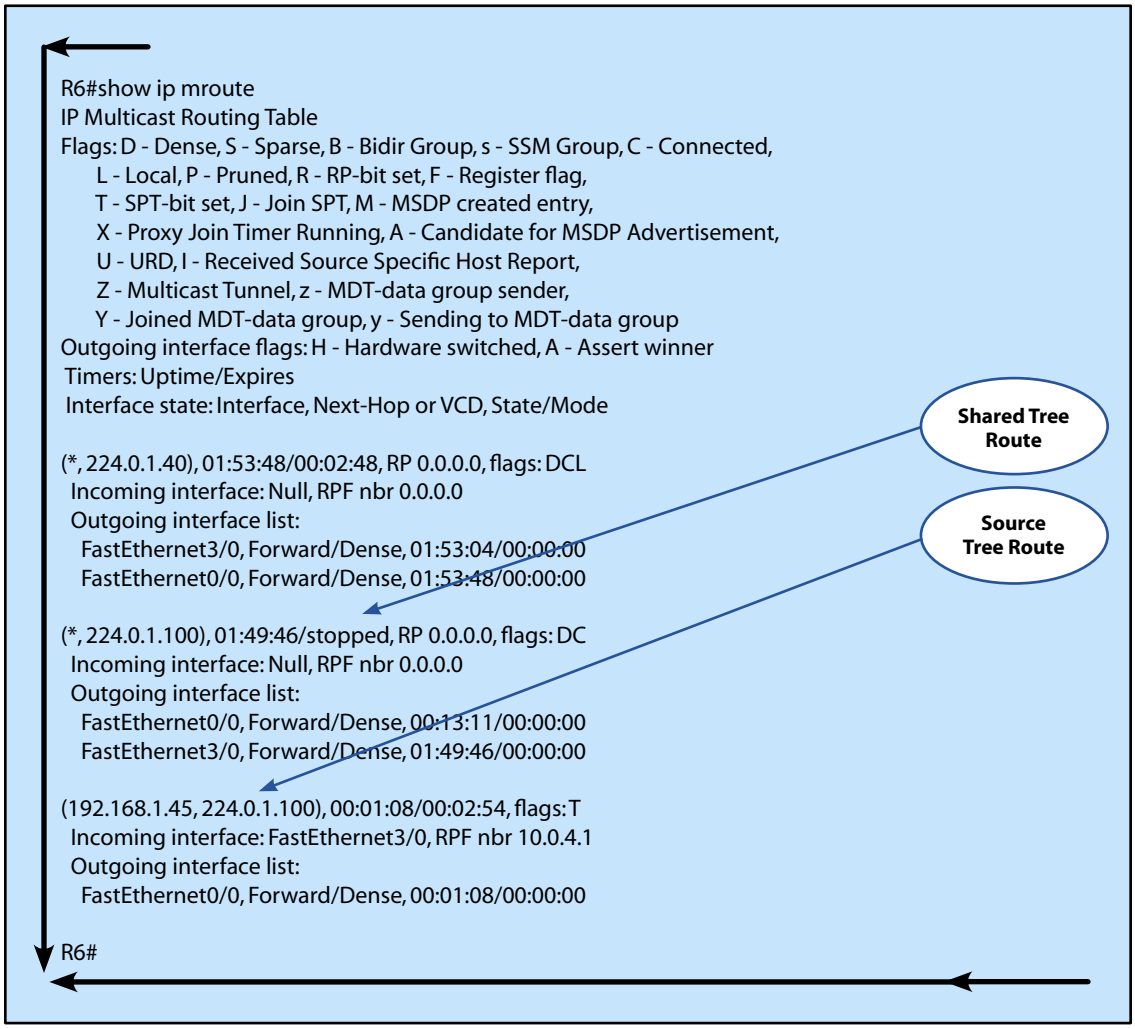
Syntax:

router(config)#**ip pim bsr-border**

Troubleshooting

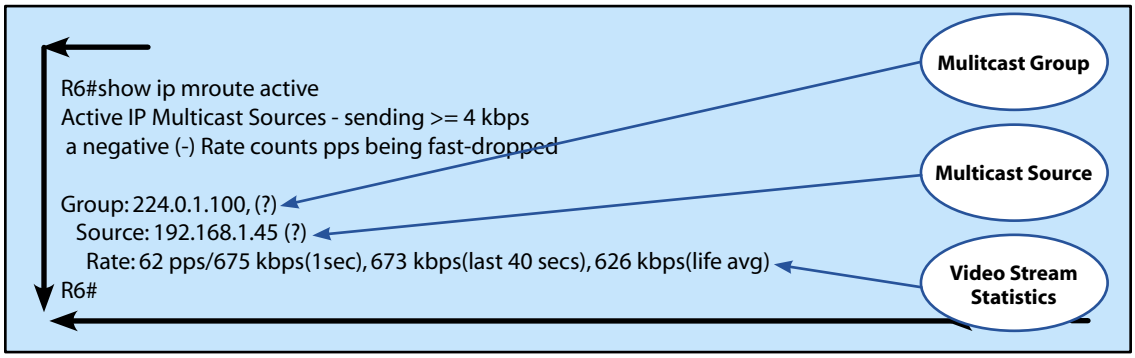
show ip mroute

This command is used to show BGP routable prefixes. The following highlights the most important parts.



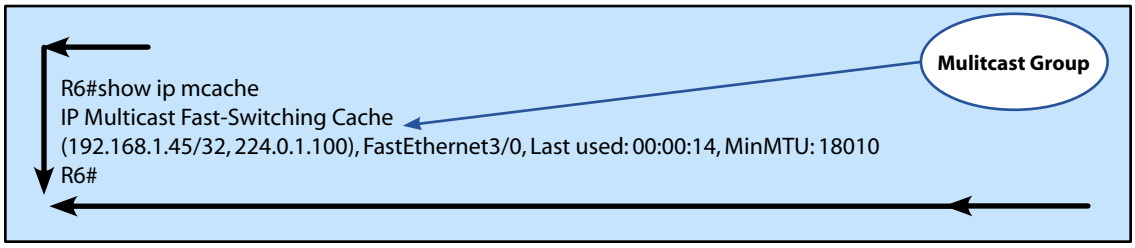
show ip mroute active

This command is used to show all active multicast groups. The following highlights the most important parts.



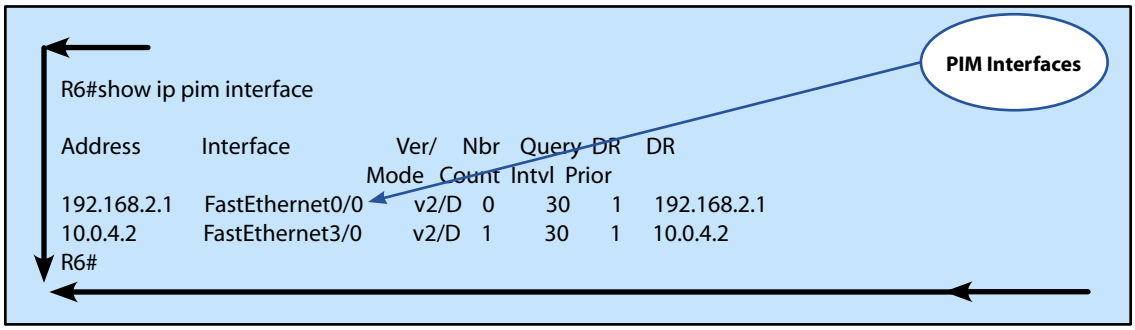
show ip mcache

This command is used to show the multicast fast switch cache. The following highlights the most important parts.



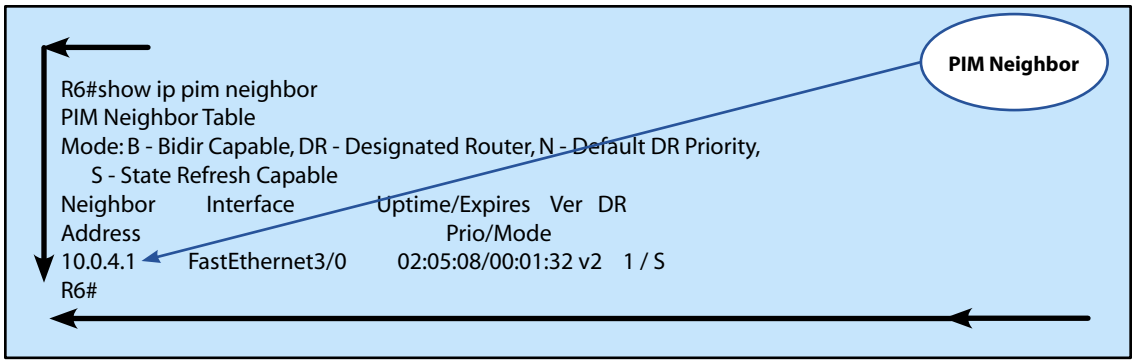
show ip pim interface

This command is used to show all PIM interfaces on a router. The following highlights the most important parts.



show ip pim neighbor

This command is used to show all PIM neighbors. The following highlights the most important parts.



Domain 7 – Implement Ipv6

IPv4 – IPv6 Changes

Since the people reading this manual are already familiar with IPv4, it is prudent to review the changes between IPv4 and IPv6 and relate what has changed between the two. This being said, we will show the IPv4 and the IPv6 headers and review the changes.

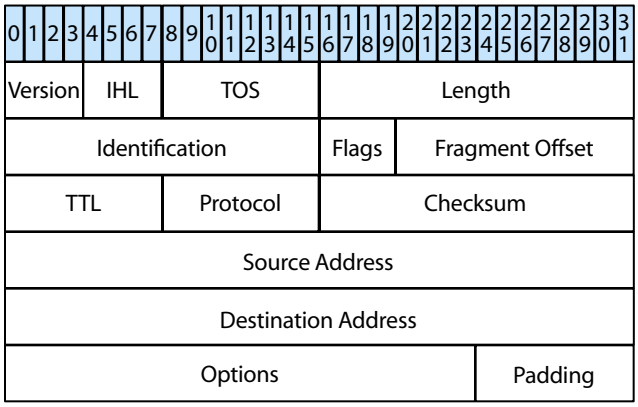


Figure 25 - IPv4 Header

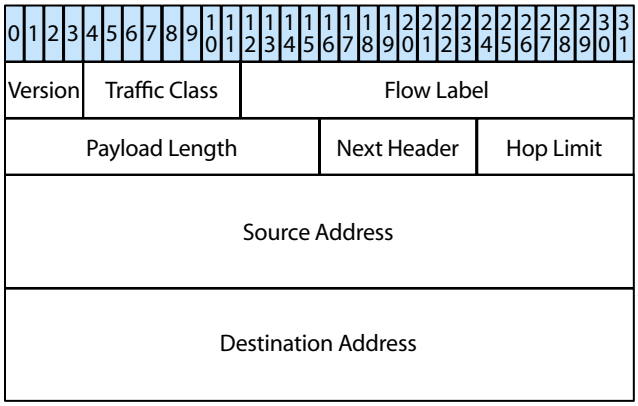


Figure 26 - IPv6 Header

The first obvious difference between the two headers is that the new IPv6 header is more simplified than the IPv4 header, which allows easier forwarding of traffic and less tax on the networking equipment. The next thing that is noticed is that the field names have changed; however, most have the same information in them as in the older header. The following is an overview of the different new fields:

Version	This provides the same information as in IPv4. In this case it states that the packet is IP Version 6.
Traffic Class	The traffic class field works similarly to the ToS field in IPv4. This allows different distinctions of class and priority of traffic.
Flow Label	This field provides two general duties at this time: to allow specific traffic (a flow) to be handled faster on a per-flow basis and also can be used to relate traffic classes. The purpose of this field will evolve as time goes on.
Payload Length	This is the overall length of the data in the packet.
Next Header	This works like the IPv4 protocol field. Specifically it states how the traffic coming after this header will be identified. For example, will it be TCP or UDP traffic.
Hot Limit	This field works like the TTL field in IPv4; however, instead of being based on time, it is decremented every time the packet is forwarded.
Source Address	The Source IPv6 Address.
Destination Address	The Destination IPv6 Address.

IPv6 also has allowed for the addition of extension headers to come after the destination address. This is only there if specified. Something else that changes between the IPv4 and IPv6 headers is the header checksum. On IPv4 traffic, everytime the header needs to be rewritten, the checksum had to be recalculated. This included the decrement of the TTL. This extra processing adds up over time. With IPv6 the assumption is made that checksums will be done at the higher layers and is thus not required at layer 3.

How IPv6 handles fragmentation is also different with IPv4. This was handled in the main IPv4 header but with IPv6 it is handled with an extension header if needed. Fragmentation is done at the source node only; all intermediate routers do not fragment packets anymore.

IPv6 Addresses

One of the main purposes in creating a new IP standard was to allow for a larger range of addresses. The current standard (IPv4) is being used up quickly and a new larger address base standard needed to be created. With the IPv4 standard, the total amount of addresses available was 2^{32} or 4,294,967,296 addresses. While this sounds like a lot, when the world's population is considered and how many potential devices each of those people may have which are IP capable, this number can be reached quickly. As the Internet is not slowing down but speeding up as more users become capable of getting online, a solution was standardized. With IPv6 the amount of addresses went from 2^{32} to 2^{128} or 340,282,366,920,938,000,000,000,000,000,000,000,000 addresses, certainly enough for now.

Something that needed to be changed also between IPv4 and IPv6 was the way that it was notated. IPv4 addresses are reasonably easy to remember because they are notated in a neat 4 octet format (a.b.c.d). But since the addresses with IPv6 were so much larger notating IPv6 in a decimal format is not practical. Because of this it was decided to use the hex format. With the hex notation the address is notated in eight 16-bit fields. This makes the address look like this:

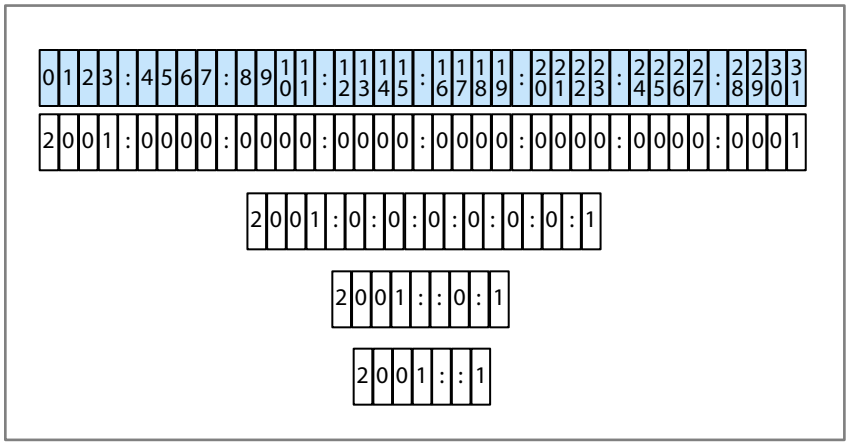


Figure 27 - IPv6 Address Examples

The top address is a full written out IPv6 address; however, because the address space is so large there will be a number of addresses that have several strings of zeros. Because of this, a special notation has been created to deal with these zeros. First, leading zeros can be omitted with each 16-bit section; secondly, the use of the double colon "::" can be used to notate a large string of zeros that extend past the 16-bit sections. However, the one exception is that the double colon can only be used once. Because of these zero notations, all of the addresses in the figure are notating the same address.

Address Types

Within IPv6 there are a number of different types of addresses like in IPv4. Within IPv4 there are unicast, broadcast and multicast type addresses. Within IPv6 this has changed a little but they have similar functions. The IPv6 types are unicast, multicast and anycast. The unicast type is the same as with IPv4. These are used for traffic from one to one. The multicast type works the same as with IPv4 but it has two differences. The multicast functionality is built into IPv6; it is not an addendum. And multicast with IPv6 works for all the same things as IPv4 and provides the functionality of IPv4 broadcast. The anycast group provides a functionality that did not exist with IPv4; devices are assigned to an anycast group. When traffic is sent to an anycast group the traffic only goes to the closest neighbor.

Like IPv4, all IPv6 devices have a loopback address. In IPv6, this address is ::1.

Unicast

Within the unicast type there are a couple of different formats the addresses can be in. These are laid out as follows:

Aggregatable Global Address

This type of address is used for normal public IPv6 traffic and is assigned like typical IPv4 addresses today. The reason they are called aggregatable is because the address structure allows easy address summarization. The following shows the address format:

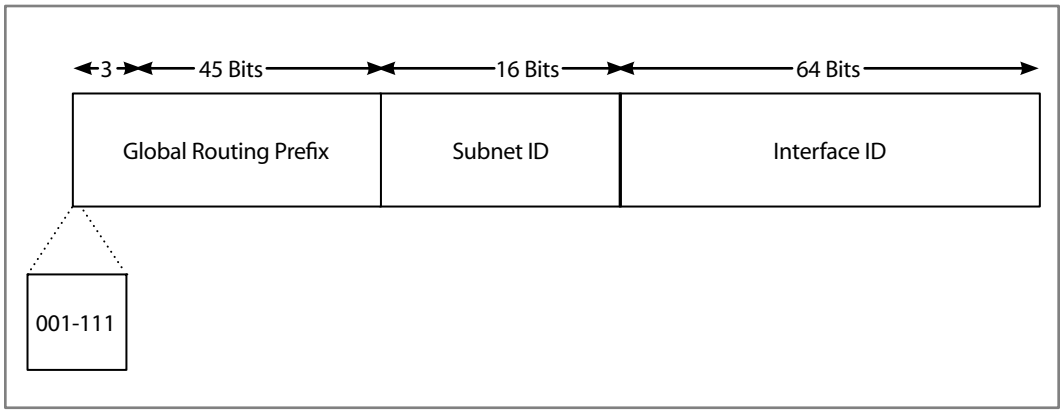


Figure 28 - Aggregatable Global Address

This type of address has a couple of main sections: Global Routing Prefix, Subnet ID, and Interface ID. The Global Routing Prefix is assigned by the provider or site. The generally accepted assignment policy states that the first 12 Bits are assigned by the Registry, the next 20 Bits by the ISP, and the next 16 Bits by the Site. The Subnet ID works similarly to an IPv4 subnet mask. This area is used to create subnets of different sizes. The Interface ID is equivalent to the node identifier. This identifier must be constructed in a modified EUI-64 format. The Interface ID is created from the MAC address on Ethernet interfaces and other interfaces choose the first from the pool of MAC addresses on the router (Ethernet Interfaces).

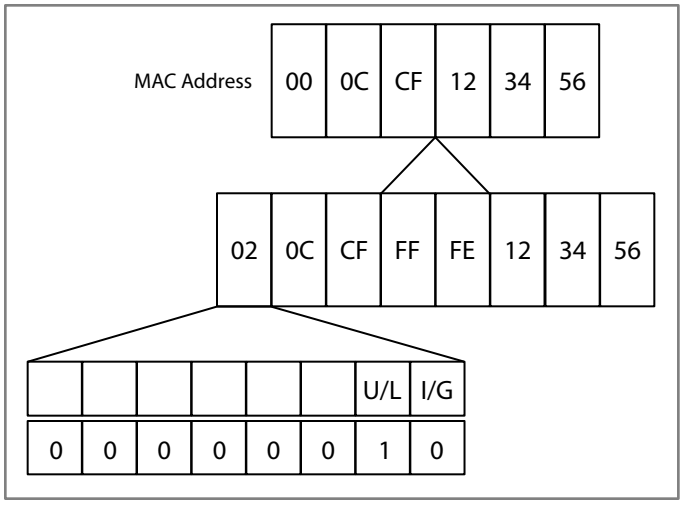


Figure 29 - EUI-64 Format

As seen in the figure, FFFE is inserted in the middle of the MAC address. The U/L Bit is used to notate whether the Interface ID is locally or universally unique. Addresses created from a unique MAC address are considered universal. The I/G bit is used to specify whether the interface is for an individual or a group (multicast).

Link-Local Addresses
This type of address is equated to the 169.254/16 IPv4 range, except on IPv6 these are auto-configured to every interface. The Interface ID is configured in the same way as with Aggregatable Global Addresses in EUI-64 format.

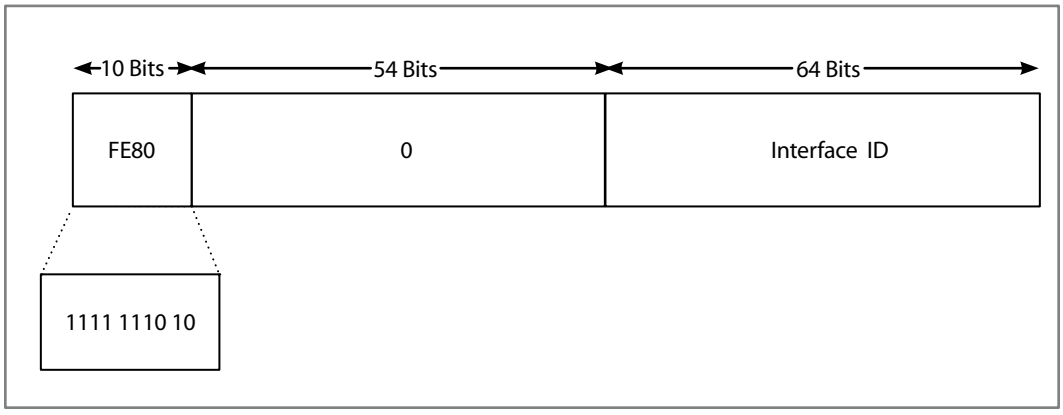


Figure 30 - Link-Local Addresses

Anycast Addresses

Anycast addresses use the Aggregatable Global Address format and are assigned as multiple addresses on an interface, at least one being unicast. All members in an anycast group can talk to each other. The purpose of the anycast is to reach the closest neighbor.

Multicast Addresses

Multicast addresses within IPv6 are used for the same purposes as with IPv4. IPv6 assigns a specific range of addresses to be used for multicast and a specific format as follows:

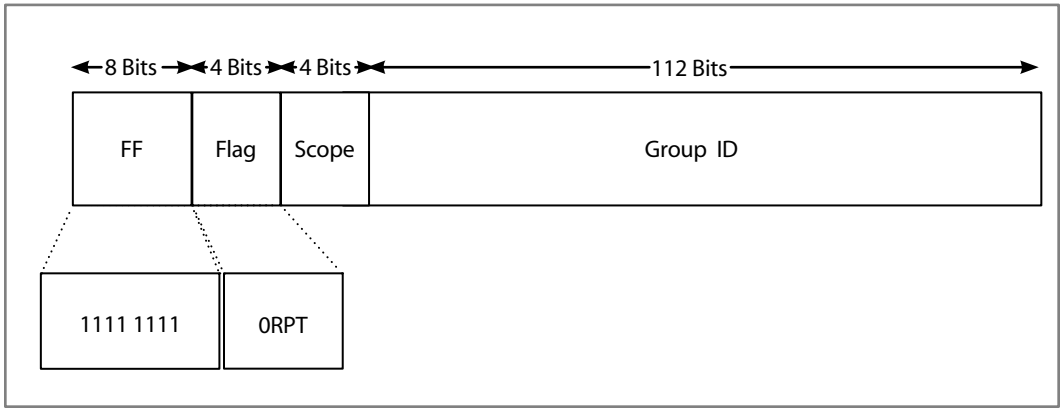


Figure 31 - Multicast Address

The first eight bits of every multicast address are the same and set to FF (1111 1111). The next four bits are assigned as flags: the first bit is currently undefined; the second bit is the "R" bit and is set to 1 if the rendezvous point is embedded in the address; the third bit is the "P" bit and is set to 1 if the address is based on the unicast prefix; and the fourth bit is the "T" bit and is set to 0 if the address is permanently assigned, and to 1 if it is temporary. The next four bits of the address are assigned as the scope of the address. The following shows the different defined scope options:

Scope	Purpose
1	Interface-local, used as a Loopback address
2	Link-local, used like a unicast link-local address
4	Admin-local scope, which must be configured
5	Site-local scope, spans the entire site
8	Organization-local scope, spans the entire organization
E	Global Scope, which is not limited

All devices are required to recognize and respond to the all-nodes addresses, FF01::1 and FF02::1, which are interface-local and link-local respectively.

IPv6 Address Assignment

Within IPv6, there are three different ways to assign an IP address. This is done through either stateless autoconfiguration, which uses a link-local address and is mapped to the device MAC address, through static configuration or through stateful DHCPv6. Stateful DHCPv6 address assignment does not have to be linked to a MAC address. However, there is a specification for stateless DHCPv6 that requires the IP address be assigned through stateless means and other information like DNS is retrieved through the DHCPv6 servers.

IPv4 – IPv6 Transition

Obviously there is not going to be instant transition between IPv4 and IPv6, which requires a mechanism to be defined to transition between the two. Because of this requirement, three different solutions have been proposed; Dual Stack, Tunneling, and Translation.

Dual Stack

Dual Stack simply runs both IPv4 and IPv6 separately on devices; they do not talk to each other. If a device is reachable through both IPv4 and IPv6, IPv6 will be preferred. Dual Stack is configured simply by assigning both an IPv4 address and an IPv6 address to an interface.

Tunneling

Tunneling has been defined because most core networks still only run IPv4. Because of this Dual Stack would not work because IPv6 is not run on every router. Tunneling solves this by tunneling IPv6 within an IPv4 packet. There are four different types of tunneling that have been defined: manual, 6-to-4, Teredo, and Intra-Site Automatic Tunnel Addressing Protocol (ISATAP).

Manual

Manual tunneling is relatively obvious; a manual tunnel is configured from the IPv4/IPv4 edge router and terminates on the far end IPv4/IPv6 edge router.

6-to-4

A 6-to-4 tunnel works the same as a manual tunnel but it is setup automatically. 6-to-4 tunnels use IPv6 addresses that concatenate 2002:: with the IPv4 address of the edge router, which creates a 48 bit prefix (a.b.c.d = 2002:a:b:c:d::). A static route between both IPv4 interfaces (The Tunnel) tells the router where to route traffic.

Teredo

Teredo works similarly to the other tunnels but encapsulates IPv6 traffic into IPv4/UDP segments, which allows the traffic to traverse Network Address Translation (NAT) devices and firewalls.

ISATAP

ISATAP treats the network like it is NBMA and specifies a protocol to automatically tunnel traffic over IPv4.

Translation

The problem with tunneling is that if IPv6 hosts want to speak with IPv4 clients a dual stack is still required. This has caused a couple of translation methods to be developed. Stateless IP/ICMP Translation (SIIT) has been defined to translate IP header fields and NAT Protocol Translation (NAT-PT) has been defined to map IPv6 addresses to IPv4 addresses. Some other technologies have also been developed to do some translations on a protocol by protocol basis on the host.

OSPF with IPv6

Fundamentally OSPFv2 and OSPFv3 (IPv6 Version) are very similar, with a few differences covered here. OSPFv2 and OSPFv3 do not work together but they can both be run on the same router.

OSPFv3 Differences

OSPFv2 uses two different multicast addresses to talk to devices. One is 224.0.0.5, which is used to talk to all OSPFv2 routers. The other is 224.0.0.6, which is used to talk between DR and BDR routers. In OSPFv3, these addresses change to FF02::5 and FF02:6, respectively. OSPFv3 routers are expected to support more then one IP address per interface, including the link-local address, multicast address and a global unicast address. OSPFv3 uses its link-local address as the source for all advertisements. OSPFv3 also uses a slightly different mechanism for selecting which networks to route. Instead of routing a network or subnet, it routes per link or interface. Because of this, configuration is done on the interface, not in router configuration mode like OSPFv2. OSPFv3 also does not support authentication directly. It relies on the built-in authentication of IPv6, using extension headers.

There are a couple of modest differences in existing LSA's that change in OSPFv3. LSA's 1 and 2 no longer contain route prefixes but instead contain 32-bit ID's.

OSPFv3 also defines two new LSA's that do not exist in OSPFv2. These include LSA 8 and LSA 9. LSA 8 is used to advertise a router's link-local address to all neighbors on a link and inform all neighbors on the link of what prefixes are associated with the link. LSA 9 is used to advertise prefixes per router ID.

Configuration

ipv6 unicast-routing

The **ipv6 unicast-routing** command is used to enable the forwarding of IPv6 unicast packets.

Syntax:

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

ipv6 cef

The **ipv6 cef** command is used to enable Cisco Express Forwarding for IPv6.

Syntax:

```
router(config)#ipv6 cef
```

ipv6 address

The **ipv6 address** {*ipv6-address/prefix-length*} ***{link-local | anycast | eui-64}*** command is used to assign an IPv6 address to an interface. The *ipv6-address* parameter specifies the IPv6 address. The *prefix-length* specifies the length of the masked prefix in decimal. The ***link-local, anycast and eui-64*** parameters specify what type of address you are configuring. If nothing is specified it will go off the address.

Syntax:

router(config-if)#**ipv6 address** {*ipv6-address/prefix-length*} {***link-local*** | ***anycast*** | ***eui-64***}

tunnel source

The **tunnel source** *ipv6-address* command is used to set the source address of a tunnel. The *ipv6-address* parameter specifies the IPv6 source address.

Syntax:

router(config-if)#**tunnel source** *ipv6-address*

tunnel destination

The **tunnel destination** *ipv6-address* command is used to set the destination address of a tunnel. The *ipv6-address* parameter specifies the IPv6 destination address.

Syntax:

router(config-if)#**tunnel destination** *ipv6-address*

tunnel mode ipv6ip

The **tunnel mode ipv6ip** [**6to4** | **isatap**] command is used to specify what type of tunnel is being configured. The **6to4** parameter is used to specify a 6-to-4 tunnel. The **isatap** parameter is used to specify an ISATAP tunnel.

Syntax:

router(config-if)#**tunnel mode ipv6ip** [**6to4** | **isatap**]

ipv6 router ospf

The ipv6 r**outer ospf** *process-id* command is used to enable OSPFv3 on a router. The *process-id* is a number which internally identifies this instance of OSPF on this router. It is only relevant to the local router and does not have to be the same across the network. It is possible, although not common, to run more than one instance of OSPF on the same router.

Syntax:

router(config)#ipv6 **router ospf** *process-id*

router-id

The router-id {*ip-address* | *ipv6-address*} command is used to specify the OSPF router ID. The *ip-address* parameter is used to specify an IPv4 router ID. The *ipv6-address* parameter is used to specify an IPv6 router ID.

Syntax:

router(config-router)#router-id {*ip-address* | *ipv6-address*}

area range

In order to configure internal summarization, the **area** *area-id* **range** *ipv6-prefix* **/***prefix-length* [**advertise** | **not-advertise**] [**cost** *cost*] command must be utilized. The command is only to be used on the ABR. Specifically, internal route summarization limits the routes advertised from one area into the other areas. By using this command summary, routes can be used to consolidate all individual routes into a limited amount of summary routes.

Syntax:

router(config-router)# **area** *area-id* **range** ipv6-*prefix* /*prefix-length* [**advertise** | **not-advertise**] [**cost** *cost*]

- *area-id* is the area which is to be summarized.
- *ipv6-prefix* is the IP network which is going to be summarized.
- *prefix-length* is the masked prefix length of the boundary to be summarized at.
- **advertise** is an optional parameter which enables the advertisement of the route via Type-3 LSA's.
- **not-advertise** is an optional parameter which disables the advertisements of the route summary and all component routes are hidden from other areas.
- **cost** *cost* is the cost which is associated with the summary route into the other areas for SPF calculations.

ipv6 ospf area

The i**pv6 ospf** *process-id* **area** *area-id* [**instance** *instance-id*] command is used to specify the area that an interface is configured in. The *process-id* is a number that internally identifies this instance of OSPF on this router. It is only relevant to the local router and does not have to be the same across the network. It is possible, although not common, to run more than one instance of OSPF on the same router. The *area-id* is simply the area number you want to include this network into. The *instance-id* is an optional parameter used to specify different instances.

Syntax:

router(config-if)#**ipv6 ospf** *process-id* **area** *area-id* [**instance** *instance-id*]

ipv6 ospf priority

The **ipv6 ospf priority** *number-value* command is used to set the router priority. The *number-value* parameter is a number from 0 to 255. By default this is set to 1.

Syntax:

router(config-if)#**ipv6 ospf priority** *number-value*

ipv6 ospf cost

The i**pv6 ospf cost** *interface-cost* command is used to set the interface cost. The *interface-cost* parameter is a number from 1 to 65535.

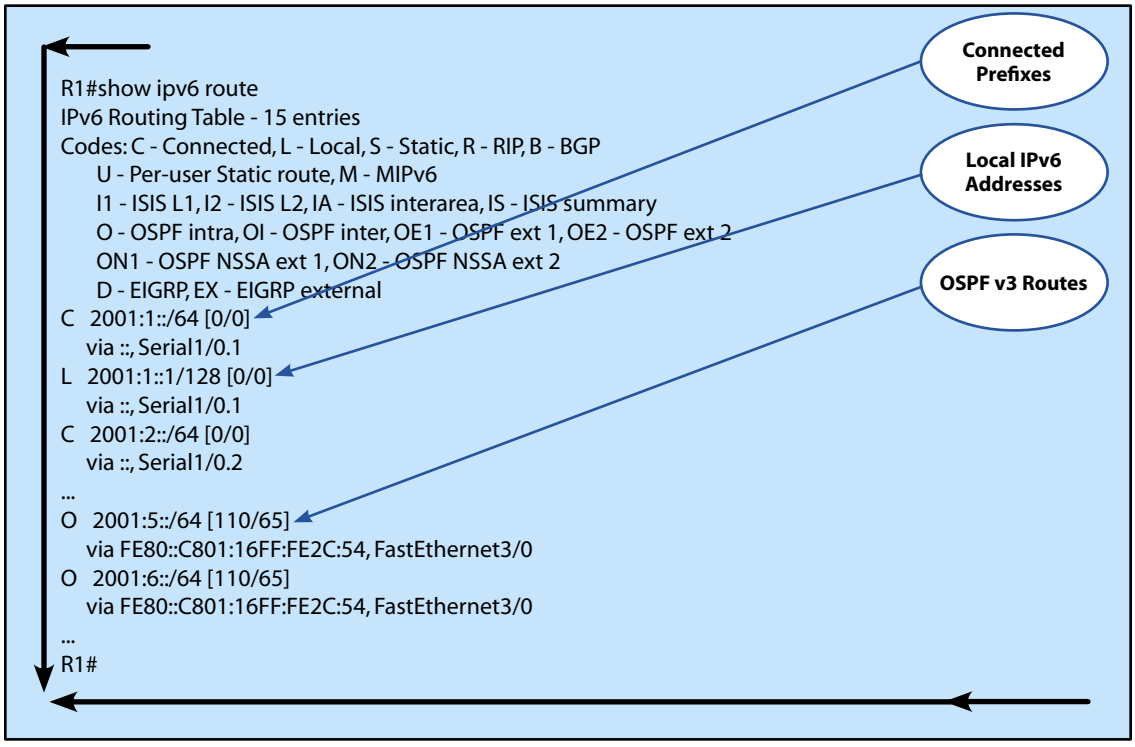
Syntax:

router(config-if)#**ipv6 ospf cost** *interface-cost*

Troubleshooting

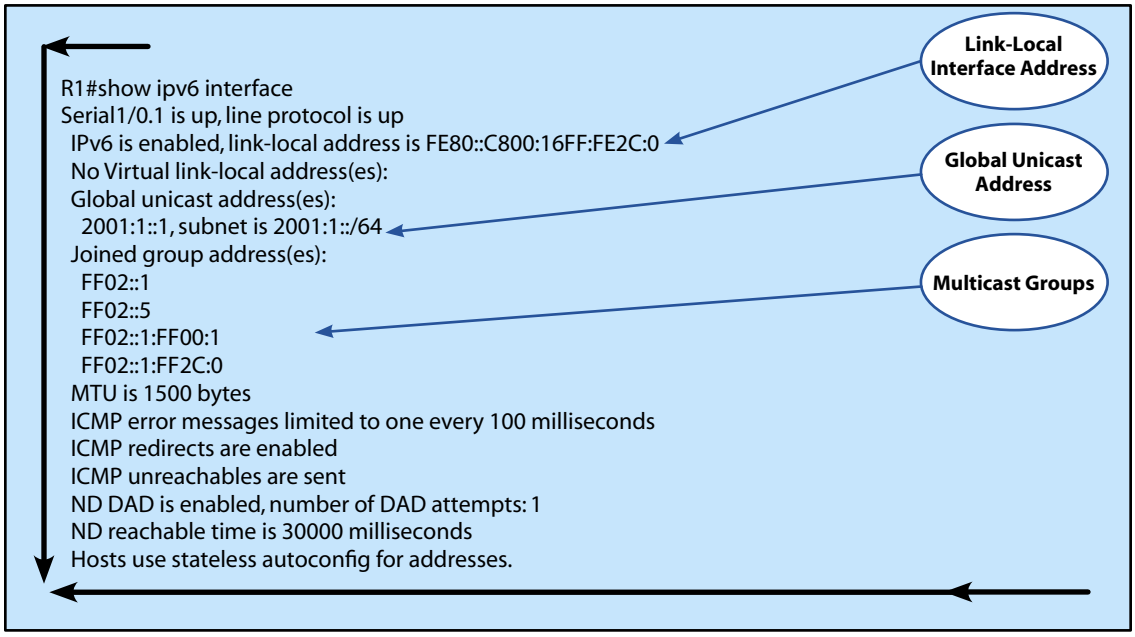
show ipv6 route

This command is used to show all IPv6 routes. The following highlights the most important parts.



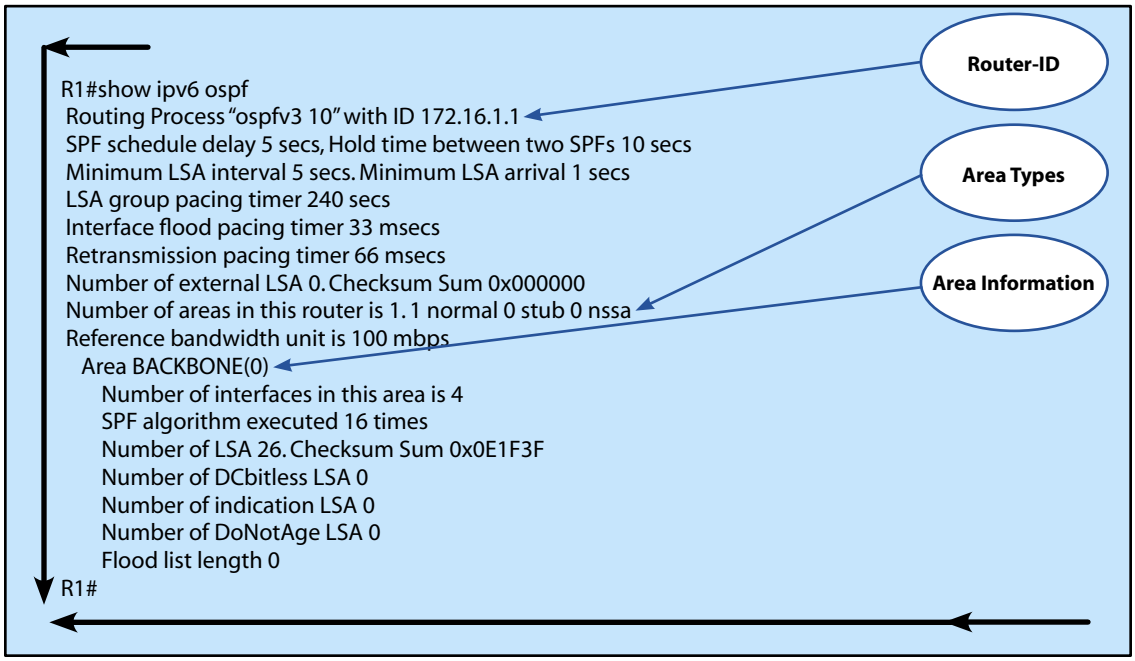
show ipv6 interface

This command is used to show all IPv6 interfaces on a router. The following highlights the most important parts.



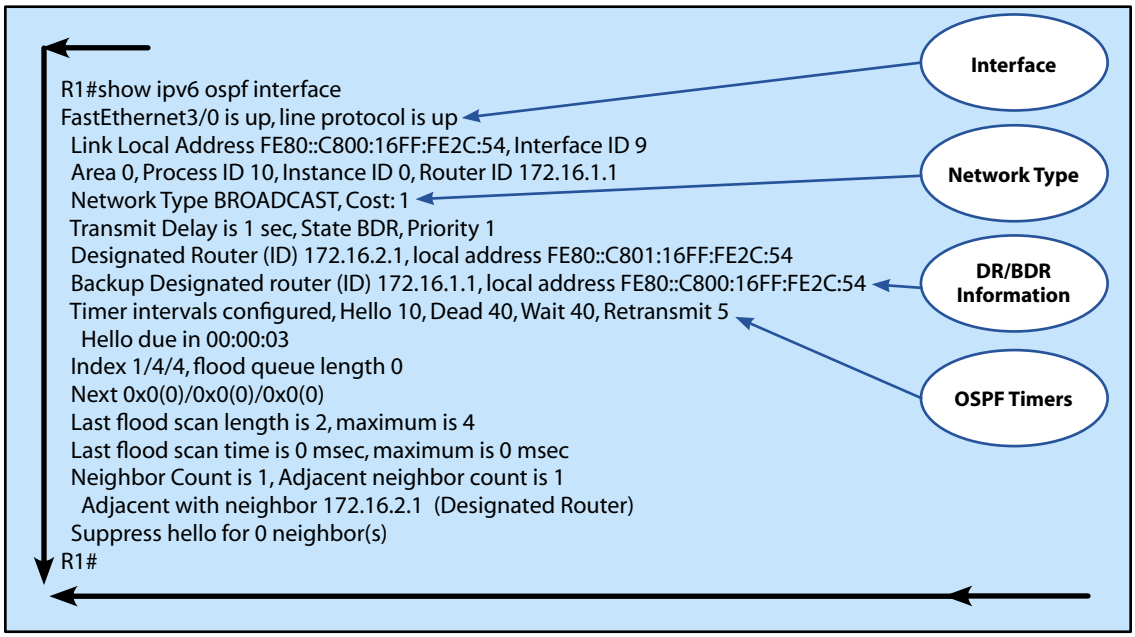
show ipv6 ospf

This command is used to show OSPFv3 process information. The following highlights the most important parts.



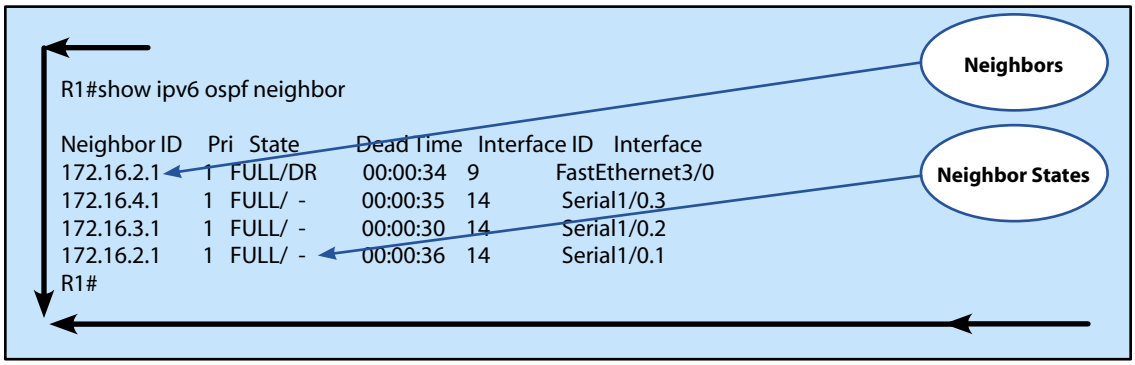
show ipv6 ospf interface

This command is used to show all OSPFv3 interfaces on a router. The following highlights the most important parts.



show ipv6 ospf neighbor

This command is used to show all OSPFv3 neighbors on a router. The following highlights the most important parts.



show ipv6 ospf database

This command is used to show the OSPFv3 database on a router. The following highlights the most important parts.