

- Enhanced Interior Gateway Routing Protocol -

EIGRP (Enhanced Interior Gateway Routing Protocol)

EIGRP is a Cisco-proprietary Hybrid routing protocol, incorporating features of both Distance-Vector and Link-State routing protocols.

EIGRP adheres to the following Hybrid characteristics:

- EIGRP uses **Diffusing Update Algorithm (DUAL)** to determine the best path among all “feasible” paths. DUAL also helps ensure a loop-free routing environment.
- EIGRP will form **neighbor** relationships with adjacent routers in the same **Autonomous System (AS)**.
- EIGRP traffic is either sent as unicasts, or as multicasts on address **224.0.0.10**, depending on the EIGRP packet type.
- Reliable Transport Protocol (RTP) is used to ensure delivery of most EIGRP packets.
- EIGRP routers **do not** send periodic, full-table routing updates. Updates are sent when a change occurs, and include *only* the change.
- EIGRP is a classless protocol, and thus supports VLSMs.

Other characteristics of EIGRP include:

- EIGRP supports IP, IPX, and Appletalk routing.
- EIGRP applies an Administrative Distance of **90** for routes originating *within* the local Autonomous System.
- EIGRP applies an Administrative Distance of **170** for external routes coming from *outside* the local Autonomous System
- EIGRP uses **Bandwidth** and **Delay of the Line**, by default, to calculate its distance metric. It also supports three other parameters to calculate its metric: **Reliability**, **Load**, and **MTU**.
- EIGRP has a maximum hop-count of **224**, though the default maximum hop-count is set to **100**.

EIGRP, much like OSPF, builds three separate tables:

- **Neighbor table** – list of all neighboring routers. Neighbors must belong to the same **Autonomous System**
- **Topology table** – list of *all* routes in the Autonomous System
- **Routing table** – contains the *best* route for each known network

* * *

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Neighbors

EIGRP forms neighbor relationships, called **adjacencies**, with other routers in the same AS by exchanging **Hello** packets. Only after an adjacency is formed can routers share routing information. Hello packets are sent as multicasts to address 224.0.0.10.

By default, on LAN and high-speed WAN interfaces, EIGRP Hellos are sent every **5 seconds**. On slower WAN links (T1 speed or slower), EIGRP Hellos are sent every **60 seconds** by default.

The EIGRP Hello timer can be adjusted on a *per interface* basis:

```
Router(config-if)# ip hello-interval eigrp 10 7
```

The above command allows us to change the *hello* timer to 7 seconds for Autonomous System 10.

In addition to the Hello timer, EIGRP neighbors are stamped with a **Hold timer**. The Hold timer indicates how long a router should wait before marking a neighbor inactive, if it stops receiving hello packets from that neighbor.

By default, the Hold timer is **three times** the Hello timer. Thus, on high-speed links the timer is set to **15 seconds**, and on slower WAN links the timer is set to **180 seconds**.

The Hold timer can also be adjusted on a *per interface* basis:

```
Router(config-if)# ip hold-interval eigrp 10 21
```

The above command allows us to change the *hold* timer to 21 seconds for Autonomous System 10.

Changing the Hello timer **does not** automatically change the Hold timer. Additionally, Hello and Hold timers **do not** need to match between routers for an EIGRP neighbor relationship to form.

(Reference: http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a0080093f07.shtml#eigrp_work)

* * *

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Neighbors (continued)

A **neighbor table** is constructed from the EIGRP Hello packets, which includes the following information:

- The IP address of the neighboring router.
- The local interface that received the neighbor's Hello packet.
- The Hold timer.
- A sequence number indicating the order neighbors were learned.

Adjacencies will not form unless the **primary IP addresses** on connecting interfaces are on the same subnet. Neighbors *cannot* be formed on secondary addresses.

If connecting interfaces are on *different* subnets, an EIGRP router will log the following error to console when a multicast Hello is received:

```
00:11:22: IP-EIGRP: Neighbor 172.16.1.1 not on common
subnet for Serial0
```

Always ensure that primary IP addresses belong to the same subnet between EIGRP neighbors.

To log all neighbor messages and errors to console, use the following two commands:

```
Router(config)# router eigrp 10
Router(config-router)# eigrp log-neighbor-changes
Router(config-router)# eigrp log-neighbor-warnings
```

(Reference: http://www.cisco.com/en/US/tech/tk365/technologies_configuration_example09186a0080093f09.shtml)

* * *

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

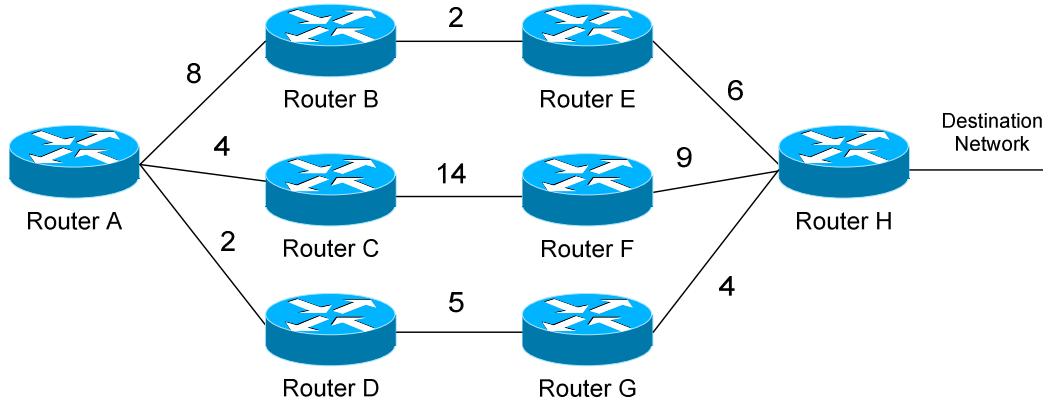
The EIGRP Topology Table

Once EIGRP neighbors form adjacencies, they will begin to share routing information. Each router's update contains a list of all routes known by that router, and the respective metrics for those routes.

All such routes are added to an EIGRP router's topology table. The route with the lowest metric to each network will become the **Feasible Distance (FD)**. The Feasible Distance for each network will be installed into the routing table.

The Feasible Distance is derived from the **Advertised Distance** of the router *sending* the update, and the local router's metric to the advertising router.

Confused? Consider the following example:



Router A has three separate paths to the Destination Network, either through Router B, C, or D. If we add up the metrics to form a “distance” (the metrics are greatly simplified in this example), we can determine the following:

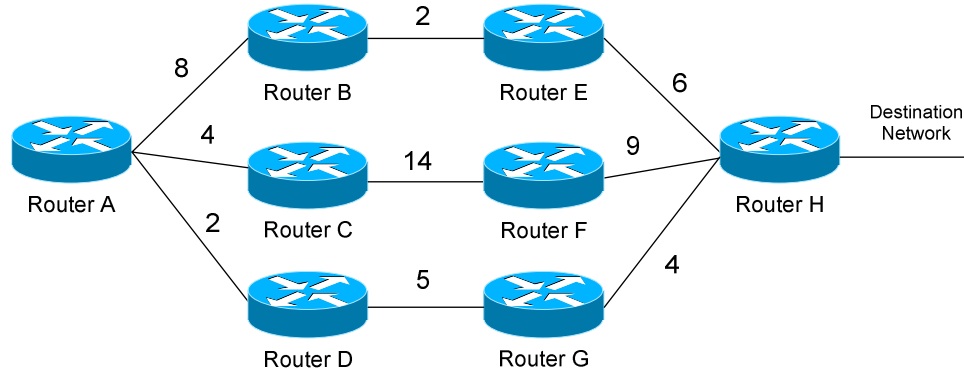
- Router B's Feasible Distance to the Destination Network is 8.
- Router C's Feasible Distance to the Destination Network is 23.
- Router D's Feasible Distance to the Destination Network is 9.

Router B sends an update to Router A, it will provide an **Advertised Distance** of 8 to the Destination Network. Router C will provide an AD of 23, and D will provide an AD of 9.

Router A calculates the total distance to the Destination network by adding the AD of the advertising router, with its own distance to *reach* that advertising router. For example, Router A's metric to Router B is 8; thus, the *total* distance will be 16 to reach the Destination Network through Router B.

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

The EIGRP Topology Table (continued)

Remember, however, that Router A's Feasible Distance must be the route with the lowest metric. If we add the Advertised Distance with the local metric between each router, we would see that:

- The route through Router B has a distance of **16** to the destination
- The route through Router C has a distance of **27** to the destination
- The route through Router D has a distance of **11** to the destination

Thus, the route through Router D (metric of 11) would become the **Feasible Distance** for Router A, and is added to the routing table as the *best* route. This route is identified as the **Successor**.

To allow convergence to occur quickly if a link fails, EIGRP includes backup routes in the topology table called **Feasible Successors (FS)**. A route will only become a Successor if its Advertised Distance is *less* than the current Feasible Distance. This is known as a **Feasible Condition (FC)**.

For example, we determined that Router A's Feasible Distance to the destination is 11, through Router D. Router C's Advertised Distance is 23, and thus *would not* become a Feasible Successor, as it has a higher metric than Router A's current Feasible Distance. Routes that are not Feasible Successors become route **Possibilities**.

Router B's Advertised Distance is 8, which is less than Router A's current Feasible Distance. Thus, the route through Router B to the Destination Network *would* become a Feasible Successor.

Feasible Successors provide EIGRP with redundancy, without forcing routers to re-converge (thus stopping the flow of traffic) when a topology change occurs. If no Feasible Successor exists and a link fails, a route will enter an **Active** (converging) state until an alternate route is found.

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Packet Types

EIGRP employs five packet types:

- **Hello packets** - *multicast*
- **Update packets** – *unicast or multicast*
- **Query packets** – *multicast*
- **Reply packets** – *unicast*
- **Acknowledgement packets** - *unicast*

Hello packets are used to form neighbor relationships, and were explained in detail previously. Hello packets are always multicast to address 224.0.0.10.

Update packets are sent between neighbors to build the topology and routing tables. Updates sent to *new* neighbors are sent as unicasts. However, if a route's metric is changed, the update is sent out as a multicast to address 224.0.0.10.

Query packets are sent by a router when a Successor route fails, and there are no Feasible Successors in the topology table. The router places the route in an **Active state**, and queries its neighbors for an alternative route. Query packets are sent as a multicast to address 224.0.0.10.

Reply packets are sent in response to Query packets, assuming the responding router has an alternative route (feasible successor). Reply packets are sent as a unicast to the querying router.

Recall that EIGRP utilizes the **Reliable Transport Protocol (RTP)** to ensure reliable delivery of most EIGRP packets. Delivery is guaranteed by having packets *acknowledged* using.....**Acknowledgment packets!**

Acknowledgment packets (also known as **ACK's**) are simply Hello packets with no data, other than an acknowledgment number. ACK's are always sent as unicasts. The following packet types employ RTP to ensure reliable delivery via ACK's:

- Update Packets
- Query Packets
- Reply Packets

Hello and Acknowledgments (ha!) packets do not utilize RTP, and thus do not require acknowledgement.

* * *

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Route States

An EIGRP route can exist in one of two states, in the topology table:

- **Active state**
- **Passive State**

A **Passive state** indicates that a route is reachable, and that EIGRP is fully converged. A stable EIGRP network will have all routes in a Passive state.

A route is placed in an **Active state** when the Successor and any Feasible Successors fail, forcing the EIGRP to send out Query packets and re-converge. Multiple routes in an Active state indicate an unstable EIGRP network. If a Feasible Successor exists, a route should *never* enter an Active state.

Routes will become **Stuck-in-Active (SIA)** when a router sends out a Query packet, but does not receive a Reply packet within **three minutes**. In other words, a route will become SIA if EIGRP fails to re-converge. The local router will clear the neighbor adjacency with any router(s) that has failed to Reply, and will place all routes from that neighbor(s) in an Active state.

To view the current state of routes in the EIGRP topology table:

```
Router# show ip eigrp topology
```

```
IP-EIGRP Topology Table for AS(10)/ID(172.19.1.1)
```

```
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status
```

```
P 10.3.0.0/16, 1 successors, FD is 2297856
   via 172.16.1.2 (2297856/128256), Serial0
P 172.19.0.0/16, 1 successors, FD is 281600
   via Connected, Serial 1
```

To view only *active* routes in the topology table:

```
Router# show ip eigrp topology active
```

```
IP-EIGRP Topology Table for AS(10)/ID(172.19.1.1)
```

```
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - Reply status
```

```
A 172.19.0.0/16, 1 successors, FD is 23456056 1 replies,
   active 0:00:38, query-origin: Multiple Origins
```

(Reference: http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a008010f016.shtml)

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Metrics

EIGRP can utilize 5 separate metrics to determine the best route to a destination:

- **Bandwidth** (K1) – Slowest link in the route path, measured in kilobits
- **Load** (K2) – Cumulative load of all outgoing interfaces in the path, given as a fraction of 255
- **Delay of the Line** (K3) – Cumulative delay of all outgoing interfaces in the path in tens of microseconds
- **Reliability** (K4) – Average reliability of all outgoing interfaces in the path, given as a fraction of 255
- **MTU** (K5) – The smallest Maximum Transmission Unit in the path. The MTU value is actually *never* used to calculate the metric

By default, only **Bandwidth** and **Delay of the Line** are used. This is identical to IGRP, except that EIGRP provides a more granular metric by multiplying the bandwidth and delay by 256. Bandwidth and delay are determined by the interfaces that lead towards the destination network.

By default, the full formula for determining the EIGRP metric is:

$$[10000000/\text{bandwidth} + \text{delay}] * 256$$

The bandwidth value represents the link with the *lowest* bandwidth in the path, in kilobits. The delay is the total delay of all outgoing interfaces in the path.

As indicated above, each metric is symbolized with a “K” and then a number. When configuring EIGRP metrics, we actually identify which metrics we want EIGRP to consider. Again, by default, only Bandwidth and Delay are considered. Thus, using on/off logic:

$$K1 = 1, K2 = 0, K3 = 1, K4 = 0, K5 = 0$$

If all metrics were set to “on,” the full formula for determining the EIGRP metric would be:

$$[K1 * \text{bandwidth} * 256 + (K2 * \text{bandwidth}) / (256 - \text{load}) + K3 * \text{delay} * 256] * [K5 / (\text{reliability} + K4)]$$

Remember, the “K” value is either set to on (“1”) or off (“0”).

* * *

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

Configuring EIGRP Metrics

EIGRP allows us to identify which metrics the protocol should consider, using the following commands:

```
Router(config)# router eigrp 10
Router(config-router)# metric weights 0 1 1 1 0 0
```

The first command enables the *EIGRP* process for Autonomous System *10*. The second actually identifies which EIGRP metrics to use. The first number (*0*) is for Type of Service, and should always be zero. The next numbers, in order, are K1 (*1*), K2 (*1*), K3 (*1*), K4 (*0*), and K5 (*0*). Thus, we are instructing EIGRP to use bandwidth, load, and delay to calculate the total metric, but not reliability or MTU.

Our formula would thus be:

$$[K1 * \text{bandwidth} * 256 + (K2 * \text{bandwidth}) / (256 - \text{load}) + K3 * \text{delay} * 256]$$

The actual values of our metrics (such as bandwidth or delay) must be configured indirectly. To adjust the bandwidth (in Kbps) of an interface:

```
Router(config)# int s0/0
Router(config-if)# bandwidth 64
Router(config-if)# ip bandwidth-percent eigrp 10 30
```

However, this command does not actually dictate the *physical* speed of the interface. It merely controls how EIGRP *considers* this interface. Best practice is to set the bandwidth to the actual physical speed of the interface. By default, a *serial* interface will have a bandwidth of **1.544 Mbps (1544)**.

The *ip bandwidth-percent eigrp* command limits the percentage of bandwidth EIGRP can use on an interface. The percentage is based on the configured *bandwidth* value. By default, EIGRP will use **up to 50%** of the bandwidth of an interface. The above command adjusts this to **30%** for Autonomous System *10*. **Percentages over 100%** can be used.

If adjustments to the EIGRP metric need to be made, the delay metric (in tens of microseconds) on an interface should be used:

```
Router(config)# int s0/0
Router(config-if)# delay 10000
```

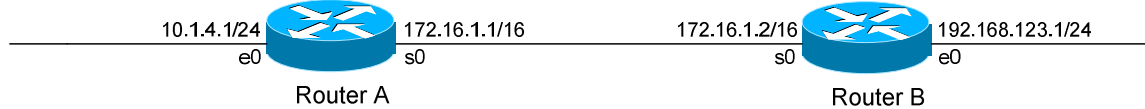
Metric settings must be **identical** on the connecting interfaces of two routers; otherwise they will not form a neighbor relationship.

* * *

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

Configuring Basic EIGRP



Routing protocol configuration occurs in Global Configuration mode. On Router A, to configure EIGRP, we would type:

```
RouterA(config)# router eigrp 10
RouterA(config-router)# network 172.16.0.0
RouterA(config-router)# network 10.0.0.0
```

The first command, *router eigrp 10*, enables the EIGRP process. The “10” indicates the Autonomous System number that we are using. The Autonomous System number can range from 1 to 65535.

Only other EIGRP routers in Autonomous System *10* will form neighbor adjacencies and share updates with this router.

The *network* statements serve two purposes in EIGRP:

- First, they identify which *networks* you wish to advertise to other EIGRP routers (similar to RIP).
- Second, they identify which *interfaces* on the local router to attempt to form neighbor relationships out of (similar to OSPF).

Prior to IOS version 12.0(4), the *network* statements were classful, despite the fact that EIGRP is a classless routing protocol. For example, the above *network 10.0.0.0* command would advertise the networks of directly-connected interfaces belonging to the 10.0.0.0/8 network and its subnets. It would further attempt to form neighbor relationships out of these interfaces.

IOS version **12.0(4) and later** provided us with more granular control of our *network* statements. It introduced a *wildcard mask* parameter, which allows us to choose the networks to advertise in a classless fashion:

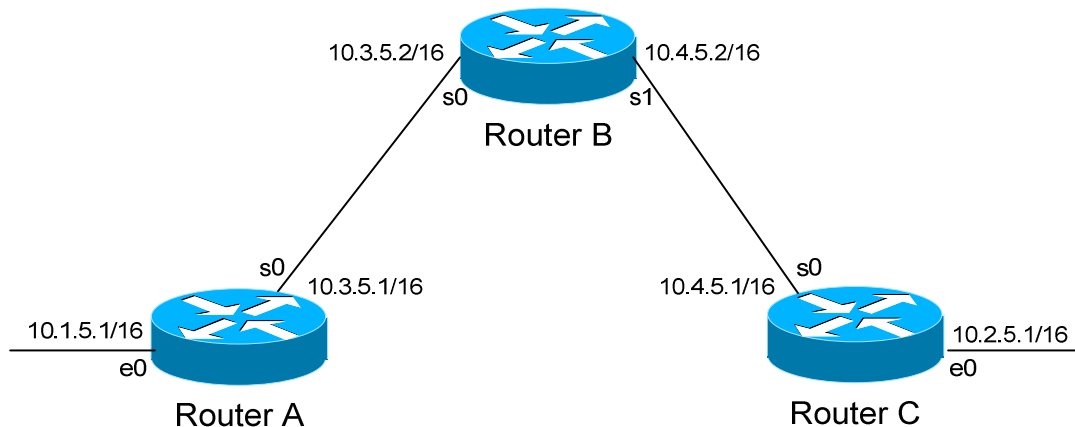
```
RouterA(config)# router eigrp 10
RouterA(config-router)# network 172.16.0.0 0.0.255.255
RouterA(config-router)# network 10.1.4.0 0.0.0.255
```

* * *

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Passive Interfaces



It is possible to control which router interfaces will participate in the EIGRP process. Just as with RIP, we can use the *passive-interface* command.

However, please note that the *passive-interface* command works differently with EIGRP than with RIP or IGRP. EIGRP will no longer form neighbor relationships out of a “passive” interface, thus this command prevents updates from being *sent* or *received* out of this interface:

```

RouterC(config)# router eigrp 10
RouterC(config-router)# network 10.4.0.0
RouterC(config-router)# network 10.2.0.0
RouterC(config-router)# passive-interface s0
  
```

Router C will not form a neighbor adjacency with Router B.

We can configure **all** interfaces to be passive using the *passive-interface default* command, and then individually use the *no passive-interface* command on the interfaces we **do** want neighbors to be formed on:

```

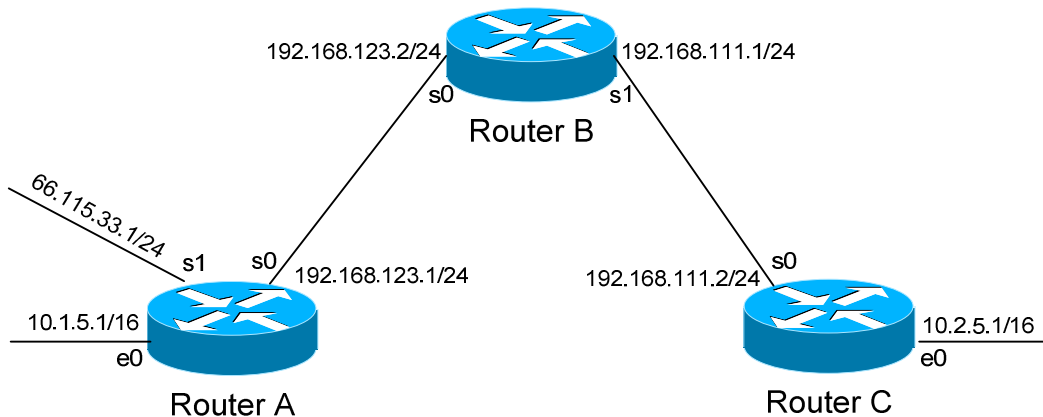
RouterC(config)# router eigrp 10
RouterC(config-router)# network 10.4.0.0
RouterC(config-router)# network 10.2.0.0
RouterC(config-router)# passive-interface default
RouterC(config-router)# no passive-interface e0
  
```

Always remember, that the *passive-interface* command will prevent EIGRP (and OSPF) from forming neighbor relationships out of that interface. **No** routing updates are passed in either direction.

* * *

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Auto-Summarization

EIGRP is a classless routing protocol that supports Variable Length Subnet Masks (VLSMs). The above example would pose no problem for EIGRP.

However, EIGRP will still **automatically summarize** when crossing **major network boundaries**.

For example, when Router A sends an EIGRP update to Router B via Serial0, by default it will still **summarize** the 10.1.0.0/16 network to 10.0.0.0/8. This is because the 10.1.0.0/16 and 192.168.123.0/24 networks *do not* belong to the same major network. Likewise, the 66.115.33.0/24 network will be summarized to 66.0.0.0/8.

An auto-summary route will be advertised as a normal *internal* EIGRP route. The *best* (lowest) metric from among the summarized routes will be applied to this summary route.

The router that *performed* the auto-summarization will also add the summary route to its routing table, with a next hop of the *Null0* interface. This is to prevent routing loops.

This **auto-summarization** can be disabled:

```

RouterA(config)# router eigrp 10
RouterA(config-router)# no auto-summary
  
```

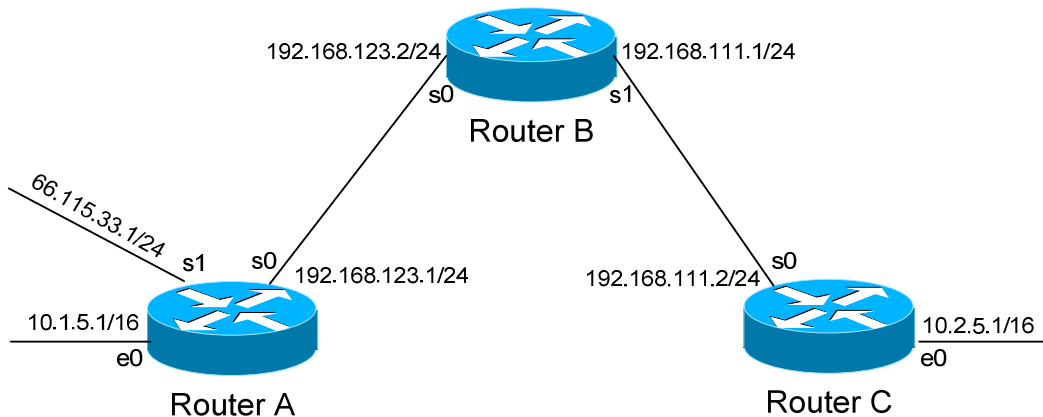
The *no auto-summary* command will prevent Router A from summarizing the 10.1.0.0/16 and 66.115.33.0/24 networks.

(Reference: http://www.cisco.com/en/US/tech/tk365/technologies_white_paper09186a0080094cb7.shtml#summarization)

* * *

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Manual Summarization

In some instances, it is necessary to *manually* summarize networks.

For example, you may not want certain networks to be auto-summarized, but other specific networks *should* be summarized. In this instance, summarization can be manually applied using the following interface configuration command:

```

RouterA(config)# int s0
RouterA(config-if)# ip summary-address eigrp 10 66.0.0.0 255.0.0.0

```

Recall that auto-summarization had been previously disabled on Router A to allow the 10.1.0.0/16 network to be advertised correctly. However, this would also mean that the 66.115.33.0/24 network would *not* be summarized as well.

The *ip summary-address* command allows us to **manually** summarize this network. Notice that we configure this on the interface that will be **advertising** this network to the other routers.

The manually-created summary route is *not* advertised as an internal EIGRP route, but instead is classified as an EIGRP summary route. An EIGRP summary route has an Administrative Distance of **5**, as opposed to an AD of **90** for internal routes.

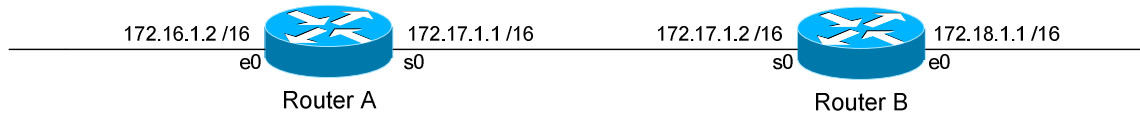
As with auto-summarization, the router performing *manual* summarization will add the summary route to its routing table, with a next hop of the *Null0* interface.

The summary route will only stay in the routing table if a more specific route still exists.

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Authentication



EIGRP supports authentication to secure routing updates.

The first step is creating a shared authentication *key* that must be identical on both routers. This is accomplished in global configuration mode:

```

RouterA(config)# key chain MYCHAIN
RouterA(config-keychain)# key 1
RouterA(config-keychain-key)# key-string MYPASSWORD

RouterB(config)# key chain MYCHAIN
RouterB(config-keychain)# key 1
RouterB(config-keychain-key)# key-string MYPASSWORD

```

The first command creates a *key chain* called *MYCHAIN*. We must then associate a *key* to our keychain. Then we actually configure the shared key using the *key-string* command.

We then apply our key chain to the interface connecting to the other router:

```

RouterA(config)# interface s0
RouterA(config-if)# ip authentication key-chain eigrp 10 MYCHAIN

RouterB(config)# interface s0
RouterB(config-if)# ip authentication key-chain eigrp 10 MYCHAIN

```

If there was another router off of Router B's Ethernet port, we could create a *separate* key chain with a different key-string. Every router on the EIGRP domain does not need to use the same key chain, only interfaces directly connecting two (or more) routers.

The final step in configuring authentication is identifying which encryption to use. Unlike RIP, EIGRP only supports MD5 encryption:

```

RouterA(config)# interface s0
RouterA(config-if)# ip authentication mode eigrp 10 md5

```

Please note that configuring authentication for EIGRP is similar to that of RIP, but there are slight variations in the commands, including the addition of the EIGRP Autonomous System Number.

* * *

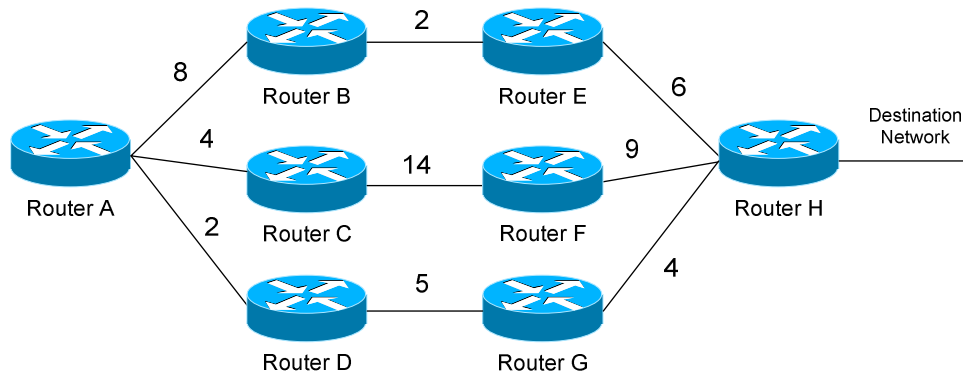
All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Load-Balancing

By default, EIGRP will automatically load-balance across equal-metric routes (four by default, six maximum). EIGRP also supports load-balancing across routes with an *unequal* metric.

Consider the following example:



Earlier in this section, we established that Router A would choose the route through Router D as its **Feasible Distance** to the destination network. The route through Router B became a **Feasible Successor**.

By default, EIGRP will *not* load-balance between these two routes, as their metrics are different (11 through Router D, 16 through Router B). We must use the *variance* command to tell EIGRP to load-balance across these unequal-metric links:

```

RouterA(config)# router eigrp 10
RouterA(config-router)# variance 2
RouterA(config-router)# maximum-paths 6
  
```

The *variance* command assigns a “multiplier,” in this instance of 2. We multiply this *variance* value by the metric of our Feasible Distance ($2 \times 11 = 22$). Thus, any Feasible Successors with a metric within twice that of our Feasible Distance (i.e. 12 through 22) will now be used for load balancing by EIGRP.

Remember, only Feasible Successors can be used for load balancing, not Possibilities (such as the route through Router C).

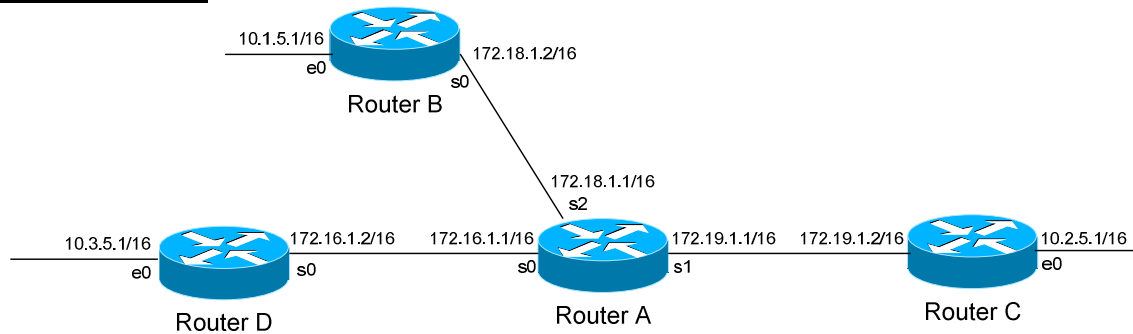
The *maximum-paths* command adjusts the number of links EIGRP can load-balance across.

(Reference: <http://www.cisco.com/warp/public/103/19.html>)

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP Stubs



Consider the above hub-and-spoke environment. If Router C were to fail, Router A (the *hub* router) would mark the 10.2.0.0/16 route as **Active**, and send out **Query** packets to the *spoke* routers for an alternate path.

However, it is obvious that no other route exists to the 10.2.0.0/16 network. Thus, the *querying* process is a waste of bandwidth and resources.

To prevent unnecessary querying, “spoke” routers in a hub-and-spoke environment can be configured as **Stub** routers. A stub router builds a neighbor adjacency with its hub router(s), and will inform neighbors of its stub status. The stub router will still build the *full* topology table.

However, the stub router will immediately respond to any Query packets with an **Inaccessible** message. Neighbors will eventually stop querying the stub router, which helps EIGRP converge quicker and conserves bandwidth.

Configuration of an EIGRP *stub* is always performed on the **spoke** router:

```
RouterB(config)# router eigrp 10
RouterB(config-router)# eigrp stub connected
```

The *eigrp stub* command configures this router as Stub, and supports four possible parameters:

- **Receive-only** – router will not share updates with neighbors
- **Connected** – router will only advertise connected networks
- **Static** – router will only advertise static networks
- **Summary** – router will only advertise summary routes

The *connected* and *static* parameters will only advertise those networks if they have been injected into the EIGRP process, either using *network* statements or using route redistribution. By default, EIGRP stubs will only send **connected** and **summary** routes to neighbors.

(Reference: <http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120limit/120s/120s15/eigrpstb.htm>)

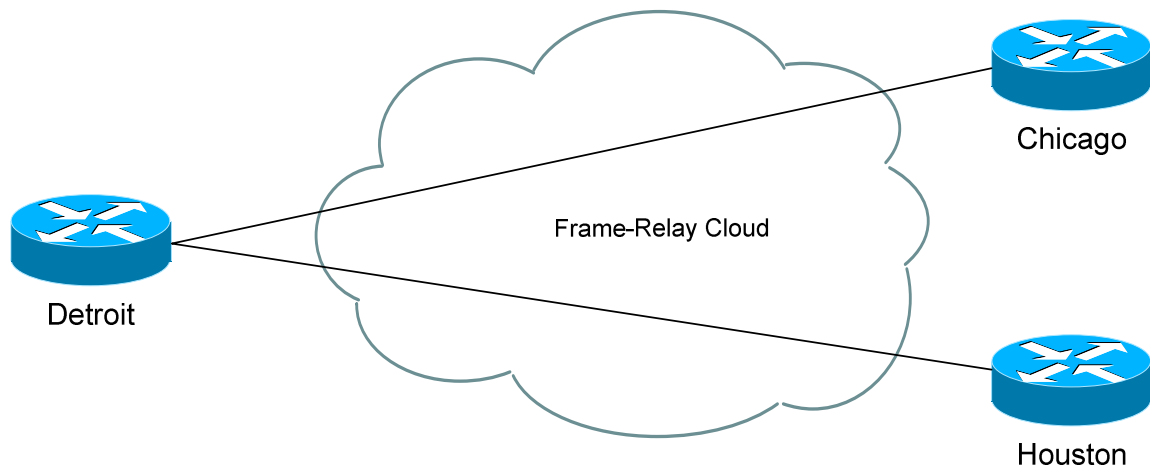
All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP, Frame-Relay, and Bandwidth

Recall that EIGRP's default bandwidth on serial links is set to 1.544 Mbps (specifically, 1544). Bandwidth on LAN interfaces (such as Ethernet) is set to the actual physical speed of the link. For point-to-point PPP or HDLC links, bandwidth should be manually adjusted to the line's physical speed.

Additional considerations exist when using Frame-Relay. Observe the following diagram. Assume the Detroit router's connection into the Frame-Relay cloud is 256 Kbps (shared between the Chicago and Houston PVCs).



Cisco specifies three rules regarding EIGRP over Frame-Relay:

- The configured bandwidth (and the percentage of bandwidth EIGRP can use) for a PVC cannot exceed the bandwidth of the PVC (CIR).
- The bandwidth for EIGRP across *all* PVCs on an interface cannot exceed the physical bandwidth of the interface
- The bandwidth for EIGRP must be identical on both ends of a PVC.

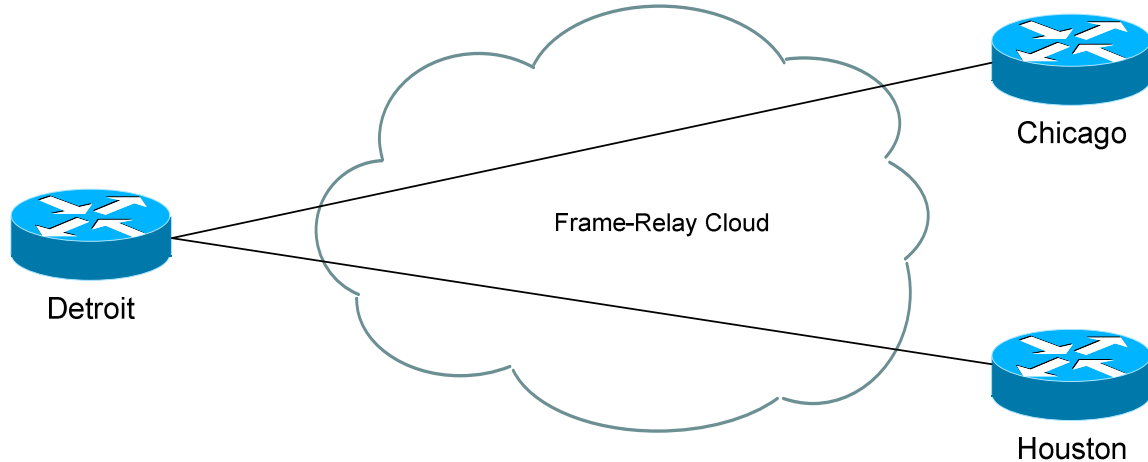
Consider if router Detroit was configured using Frame-Relay point-to-multipoint, using no sub-interfaces. Assume also that no bandwidth command is configured on the physical interface. EIGRP will assume that the bandwidth is evenly split between all PVCs. In the above scenario, EIGRP would assume that each PVC was allocated 128 Kbps.

If the CIR for the PVCs were not equal – say, Detroit to Chicago is 56Kbps, and Detroit to Houston is 256Kbps – the bandwidth should be calculated by multiplying the bandwidth of the *slowest* PVC with the total number of PVCs. In this scenario, the bandwidth should be set to 128Kbps.

(Reference: <http://www.cisco.com/warp/public/103/12.html>)

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

EIGRP, Frame-Relay, and Split-Horizon

Observe the above Frame-Relay network. We have two possible configuration options for Detroit:

- Configure frame-relay map statements on the physical interface
- Create separate subinterfaces for each link, treating them as separate point-to-points.

If choosing the latter, EIGRP will treat each subinterface as a separate link, and routing will occur with no issue.

If choosing the former, EIGRP will be faced with a split-horizon issue. Updates from Houston will not be forwarded to Chicago, and visa versa, as split horizon prevents an update from being sent out the link it was received on.

Thus, we must disable split horizon for EIGRP:

```
Detroit(config)# interface s0/0
Detroit(config-router)# no ip split-horizon eigrp 10
```

* * *

Troubleshooting EIGRP

To view the EIGRP Neighbor Table:

Router# *show ip eigrp neighbor*

IP-EIGRP neighbors for process 10

H	Address	Interface	Hold	Uptime	SRTT (sec)	RTO (ms)	Q Cnt	Seq Num	Type
0	172.16.1.2	S0	13	00:00:53	32	200	0	2	
0	172.18.1.2	S2	11	00:00:59	32	200	0	3	

To view the EIGRP Topology Table, containing all EIGRP route information:

Router# *show ip eigrp topology*

IP-EIGRP Topology Table for AS(10)/ID(172.19.1.1)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
r - reply Status, s - sia Status

```
P 10.3.0.0/16, 1 successors, FD is 2297856
   via 172.16.1.2 (2297856/128256), Serial0
P 172.19.0.0/16, 1 successors, FD is 281600
   via Connected, Serial 1
P 172.18.0.0/16, 1 successors, FD is 128256
   via Connected, Serial 2
P 172.16.0.0/16, 1 successors, FD is 2169856
   via Connected, Serial0
```

To view information on EIGRP traffic sent and received on a router:

Router# *show ip eigrp traffic*

IP-EIGRP Traffic Statistics for process 10

```
Hellos sent/received: 685/429
Updates sent/received: 4/3
Queries sent/received: 0/0
Replies sent/received: 0/0
Acks sent/received: 1/2
Input queue high water mark 1, 0 drops
SIA-Queries sent/received: 0/0
SIA-Replies sent/received: 0/0
```

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com),
unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

Troubleshooting EIGRP (continued)

To view the bandwidth, delay, load, reliability and MTU values of an interface:

Router# *show interface s0*

```
Serial0 is up, line protocol is up
  Hardware is HD64570
  Internet address is 172.16.1.1/16
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
    reliability 255/255, txload 1/255, rxload 1/255
```

<irrelevant output removed>

To view information specific to the EIGRP protocol:

Router# *show ip protocols*

```
Routing Protocol is "eigrp 10"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP metric weight K1=1, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 1
  Redistributing: eigrp 10
  Automatic network summarization is not in effect
  Maximum path: 4
  Routing for Networks:
    172.16.0.0
    172.18.0.0
    172.19.0.0
  Routing Information Sources:
    Gateway         Distance      Last Update
    (this router)           90          00:26:11
    172.16.1.2             90          00:23:49
  Distance: internal 90 external 170
```

This command provides us with information on EIGRP timers, EIGRP metrics, summarization, and the specific networks RIP is advertising.

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com), unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.

Troubleshooting EIGRP (continued)

To view the IP routing table:

Router# *show ip route*

Gateway of last resort is not set

```
C    172.16.0.0 is directly connected, Serial0
C    172.19.0.0 is directly connected, Serial1
D    10.3.0.0 [90/2297856] via 172.16.1.2, 00:00:15, Serial0
```

To view a specific route within the IP routing table:

Router# *show ip route 10.3.0.0*

```
Routing entry for 10.3.0.0/16
  Known via "eigrp 10", distance 90, metric 2297856 type internal
  Last update from 172.16.1.2 on Serial 0, 00:00:15 ago
```

To debug EIGRP in realtime:

Router# *debug eigrp neighbors*

Router# *debug eigrp packet*

Router# *debug eigrp route*

Router# *debug eigrp summary*

All original material copyright © 2007 by Aaron Balchunas (aaron@routeralley.com),
unless otherwise noted. All other material copyright © of their respective owners.

This material may be copied and used freely, but may not be altered or sold without the expressed written consent of the owner of the above copyright. Updated material may be found at <http://www.routeralley.com>.