Troubleshoot EIGRP Common Issues

Contents

- Introduction
- Prerequisites
  - Requirements
  - Components Used
- Background Information
- Neighbor Flapping
  - Network Problems
    - SIA
    - Expired Holding Timer
    - Exceeded Retry Limit
    - Restarted Peer
    - Initial Update Before Hello
  - Additional Issues
    - Configuration Changes
    - Authentication
    - Mismatch on Primary and Secondary IP Addresses
    - DMVPN
    - Flags Explained
- SIA
  - Definition of SIA
  - Symptoms
  - Possible Causes
  - Troubleshoot Tips
- Missing Prefixes
  - Missing Prefixes in RIB
    - Prefix Installed by Routing Protocol with Lower Administrative Distance
    - Distribute-List Blocks the Prefix
  - Missing Prefixes in Topology Table
    - Mask Specification for Proper Command Output
    - Split-Horizon Blocks the Prefix
- Metrics
- Duplicate Router ID
- K-Values Mismatch/Graceful Shutdown
- Unequal Cost Load Balancing (Variance)
- Static Neighbors
- Static Route Redistribution
- Reliability and Load for Metric Calculation
- High CPU
- EIGRP in Frame Relay Networks (Broadcast Queue)
- Mismatched AS Numbers
- Auto-Summary

[EIGRP Event Log](#)

[Same Network Learned by Two EIGRP Autonomous Systems](#)

[Related Information](#)

Introduction

This document describes how to troubleshoot the most common Enhanced Interior Gateway Routing Protocol (EIGRP) issues.

Prerequisites

Requirements

There are no specific requirements for this document.

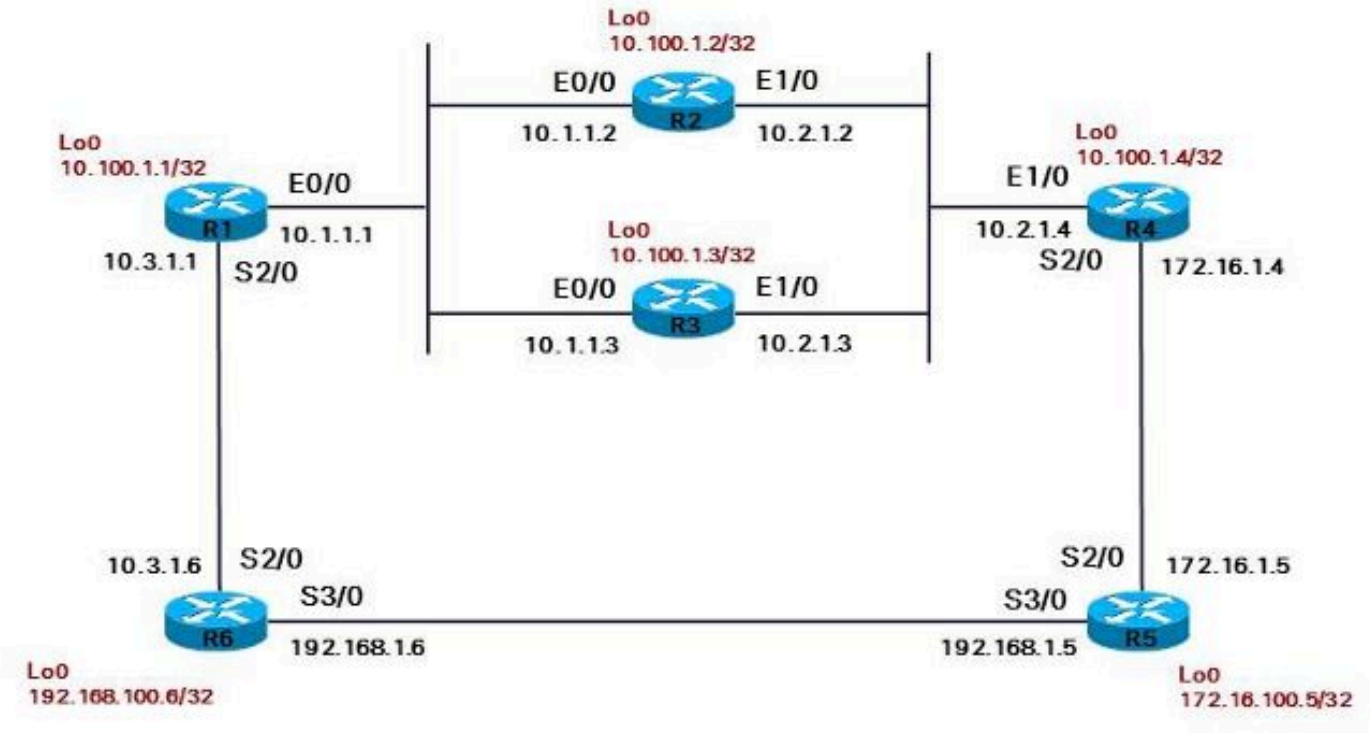
Components Used

The information in this document is based on Cisco IOS® to illustrate the various behaviors that can be encountered with this protocol.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Background Information

This is the topology that is used in this document:



The next sections describe some of the most common EIGRP issues and some tips about how to troubleshoot the issues.

Neighbor Flapping

The single most common issue that is encountered with the use of EIGRP is that it does not establish a neighborship properly. There are several possible causes for this:

- Maximum Transmission Unit (MTU) issue
- One-way communication (unidirectional links)
- There is a multicast problem on the link
- Unicast problems
- Link quality problems
- Authentication issues
- Misconfiguration issues

If you do not receive an EIGRP Hello message, you cannot see the neighbor in the neighbor list. Enter the **show ip eigrp neighbors** command in order to view the EIGRP neighbor information and identify the issue:

<#root>

R2#

show ip eigrp neighbors

```
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
3   10.1.1.1                Et0/0             12 00:00:48    1  5000
```

1

```
  0
2   10.1.1.3                Et0/0             12 02:47:13   22   200  0  339
1   10.2.1.4                Et1/0             12 02:47:13   24   200  0  318
0   10.2.1.3                Et1/0             12 02:47:13   20   200  0  338
```

If you think that the neighborship has been formed, but you do not have the prefixes that you must learn from that neighbor, check the output of the previous command: If the Q-count is always non-zero, it could be an indication that the same EIGRP packets are retransmitted continuously. Enter the **show ip eigrp neighbors detail** command in order to verify whether the same packet is always sent. If the sequence number of the first packet is always the same, then the same packet is retransmitted indefinitely:

<#root>

R2#

show ip eigrp neighbors detail

```
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
3   10.1.1.1                Et0/0             11
```

00:00:08

```
    1  4500

1

  0
   Version 12.4/1.2, Retrans: 2, Retries: 2, Waiting for Init, Waiting for Init Ack
    UPDATE seq 350 ser 0-0 Sent 8040 Init Sequenced
2   10.1.1.3                Et0/0             11 02:47:56   22   200  0  339
   Version 12.4/1.2, Retrans: 11, Retries: 0, Prefixes: 10
1   10.2.1.4                Et1/0             10 02:47:56   24   200  0  318
   Version 12.4/1.2, Retrans: 10, Retries: 0, Prefixes: 8
0   10.2.1.3                Et1/0             11 02:47:56   20   200  0  338
   Version 12.4/1.2, Retrans: 11, Retries: 0, Prefixes: 2
```

You can see in the output that the first neighbor has a problem, and the Uptime is reset.

It is important that you verify whether the process router EIGRP has the **eigrp log-neighbor-changes** command. However, this command is included by default since Cisco bug ID CSCdx67706, so it does not show up in the configuration in that case. Check the entry in the logs for both of the EIGRP neighbors on each side of the link. In at least one of the logs, there must be a meaningful entry.

Here are all of the possible reasons for an EIGRP neighborship change and their log entries:

- No EIGRP packet was received during the hold time:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is down:
 holding time expired
```

- An EIGRP reliable packet was not acknowledged within the retry limit:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is down:
 retry limit exceeded
```

- The EIGRP sees the interface in a down state:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.3.1.6 (Serial2/0) is down:
 interface down
```

- The router received an initial update packet and restarted the neighborship:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is down:
 peer restarted
```

- The router received an initial update packet and formed a new adjacency:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is up:
 new adjacency
```

- The **clear ip eigrp neighbor** command was entered, which resulted in a manual clear:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 172.16.1.4 (Serial2/0) is down:
 manually cleared
```

- The IP address on the interface was changed:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 192.168.1.5 (Serial3/0) is down:
 address changed
```

- There was a delay/bandwidth change on the interface:



Note: This only occurs in older code versions. There is no neighbor flap since Cisco bug ID CSCdp08764.

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.3.1.6 (Serial2/0) is down:
 metric changed
```

- The K-values are misconfigured or a graceful shutdown occurs:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.4.1.5 (Ethernet1/0) is down:
 K-value mismatch
```

- A graceful shutdown occurs:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is down:
 Interface Goodbye received
```

- The **ip authentication mode eigrp 1 md5** command was configured on the interface:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.3 (Ethernet0/0) is down:
 authentication mode changed
```

- A graceful restart/Non-Stop Forwarding (NSF) occurred:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.2 (FastEthernet1) is resync:
```

```
peer graceful-restart
```

- The neighbors to which there are queries sent without a reply received are cleared:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 192.168.1.16 (Serial3/0) is down:
 stuck in active
```

Network Problems

These five issues indicate a network problem:

- A Stuck-In-Active (SIA) state
- An expired holding timer
- An exceeded retry limit
- A restarted peer
- An Initial Update is sent before the Hello packet

SIA

Refer to the SIA section of this document.

Expired Holding Timer

An expired holding timer indicates that the router did not receive any EIGRP packet (that is, an EIGRP Hello or any other EIGRP packet) during the hold-time interval. There is more than likely a problem on the link in this case.

Check that the router receives the EIGRP Hello packets on this link and that the other side sends them. In order to verify this, enter the **debug eigrp packet hello** command. As an alternative to the use of the debug command, you can ping IP address 224.0.0.10 and verify whether that neighbor replies. Possible causes for the multicast problem on the link are due to interface problems, such as if an intermediate switch blocks the EIGRP Hello packets.

Another quick test that you can perform is to try another protocol that uses another multicast IP address. For example, you can configure Routing Information Protocol (RIP) Version 2 that uses the multicast IP address 224.0.0.9.

Exceeded Retry Limit

An exceeded retry limit indicates that an EIGRP reliable packet was not acknowledged multiple times. An EIGRP reliable packet is one of these five types of packets:

- Update
- Query
- Reply
- SIA-Query
- SIA-Reply

The reliable EIGRP packet was retransmitted at least 16 times. A packet is retransmitted every Retransmit Time Out (RTO). The minimum RTO is 200 ms and the maximum is 5,000 ms. The RTO increases or

decreases dynamically via observation of the time difference between the time that the reliable EIGRP packet is sent and the time that the acknowledgement is received. When the reliable packet is not acknowledged, the RTO increases. If this persists, then the RTO increases up to five seconds quickly, so the retry limit can reach 16 x 5 seconds = 80 seconds. However, if the EIGRP hold time is larger than 80 seconds, the neighborship does not go down until the hold time has expired. This can occur on slow WAN links where, for example, the default hold time is 180 seconds.

For links with hold times lower than 80 seconds, this effectively means that if the hold time does not expire, it is kept up by the EIGRP Hello packets. The retry limit can then be exceeded. This indicates that there is either an MTU problem or a unicast problem. The EIGRP Hello packets are small; the (first) EIGRP Update packet can be up to full MTU. It can be full MTU size if there are enough prefixes to fill the update. The neighbor can be learned via the reception of the EIGRP Hello packets, but full adjacency cannot succeed if the EIGRP Update packet is not acknowledged.

Typically, this is the output that appears:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is down:
 retry limit exceeded
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is up:
 new adjacency
```



Note: As of Cisco bug ID CSCsc72090, the EIGRP also uses the IP MTU settings of the interface. Before this fix was applied, the EIGRP packets would become fragmented if the IP MTU was configured with a value that was lower than 1500. This issue can typically occur in Dynamic Multipoint VPN (DMVPN) networks.

A second possibility is that the EIGRP Hello packets make it because they are multicast to IP address 224.0.0.10. Some EIGRP Update packets can make it, as they can be multicast. However, retransmitted EIGRP reliable packets are always unicast. If the unicast data path to the neighbor is broken, the retransmitted reliable packet does not process properly. Ping the EIGRP neighbor unicast IP address (with the size of the ping set to the full MTU size of the link, and with the Do Not Fragment bit (DF-bit) set) in order to verify.

A one-way link can cause this problem as well. The EIGRP router can receive the EIGRP Hello packets, but the packets that are sent from this neighbor do not make it across the link. If the Hello packets do not make it, the router is unaware because the Hello packets are unreliably sent. The EIGRP Update packets that are sent cannot be acknowledged.

The EIGRP reliable packets or the acknowledgement can become corrupted. A quick test is to send pings with reply validation enabled:

```
<#root>

R1#

ping


Protocol [ip]:
Target IP address: 10.1.1.2
Repeat count [5]: 10
Datagram size [100]:
```

```
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface:
Type of service [0]:
Set DF bit in IP header? [no]:
```

Validate reply data? [no]: yes

```
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 10, 100-byte ICMP Echos to 10.1.1.2, timeout is 2 seconds:
Reply data will be validated
!!!!!!!!!!
Success rate is 100 percent (10/10), round-trip min/avg/max = 1/24/152 ms
```

Enable the **debug eigrp packets** command in order to verify the transmission and reception of the EIGRP Hello packets and the EIGRP Update packets at a minimum:

<#root>

R1#

debug eigrp packets ?

```
  SIAquery  EIGRP SIA-Query packets
  SIAreply  EIGRP SIA-Reply packets
  ack       EIGRP ack packets
  hello     EIGRP hello packets
  ipxsap    EIGRP ipxsap packets
  probe     EIGRP probe packets
  query     EIGRP query packets
  reply     EIGRP reply packets
  request   EIGRP request packets
  retry     EIGRP retransmissions
  stub      EIGRP stub packets
  terse     Display all EIGRP packets except Hellos
  update    EIGRP update packets
  verbose   Display all EIGRP packets
```

Here is a typical example of the **retry limit exceeded** issue:

<#root>

R2#

show ip eigrp neighbors

```
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
```

```
3   10.1.1.1                Et0/0             12 00:00:48    1  5000
```

1

```
  0
2   10.1.1.3                Et0/0             12 02:47:13   22   200  0  339
1   10.2.1.4                Et1/0             12 02:47:13   24   200  0  318
0   10.2.1.3                Et1/0             12 02:47:13   20   200  0  338
```

Note: There is always one or more packets in the queue (Q Cnt).

<#root>

```
R2#
```

show ip eigrp neighbors detail

```
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
3   10.1.1.1                Et0/0             10 00:00:59    1
```

5000

1

```
  0
   Version 12.4/1.2,
```

Retrans: 12

```
, Retries: 12,
```

Waiting for Init, Waiting for Init Ack

UPDATE seq 349

```
 ser 0-0 Sent 59472 Init Sequenced
2   10.1.1.3                Et0/0             11 02:47:23   22   200  0  339
   Version 12.4/1.2, Retrans: 11, Retries: 0, Prefixes: 10
1   10.2.1.4                Et1/0             11 02:47:23   24   200  0  318
   Version 12.4/1.2, Retrans: 10, Retries: 0, Prefixes: 8
0   10.2.1.3                Et1/0             10 02:47:23   20   200  0  338
   Version 12.4/1.2, Retrans: 11, Retries: 0, Prefixes: 2
```

As shown in the output, R2 awaits the first Update packet (init bit set) from the neighbor at IP address 10.1.1.1.

In this next output, R2 awaits the acknowledgement of the first Update packet (init bit set) from the neighbor at IP address 10.1.1.1.

Note: The RTO is at its maximum of 5,000 ms, which indicates that the EIGRP reliable packets are

not acknowledged within the five seconds.

<#root>

R2#

show ip eigrp neighbors detail

```
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
3   10.1.1.1                Et0/0             11 00:01:17    1
```

5000

1

```
  0
   Version 12.4/1.2,
```

Retrans: 16

```
, Retries: 16,
```

Waiting for Init, Waiting for Init Ack

UPDATE seq 349

```
 ser 0-0 Sent 77844 Init Sequenced
2   10.1.1.3                Et0/0             12 02:47:42   22   200  0  339
   Version 12.4/1.2, Retrans: 11, Retries: 0, Prefixes: 10
1   10.2.1.4                Et1/0             10 02:47:42   24   200  0  318
   Version 12.4/1.2, Retrans: 10, Retries: 0, Prefixes: 8
0   10.2.1.3                Et1/0             11 02:47:42   20   200  0  338
   Version 12.4/1.2, Retrans: 11, Retries: 0, Prefixes: 2
```

The number of retransmissions rises steadily. It is always the same packet in the queue (seq 349). After R2 has sent this same packet 16 times, the neighborship goes down:

```
R2#
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is down:
 retry limit exceeded
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is up:
 new adjacency
```

The process begins once again:

<#root>

```
R2#
```

show ip eigrp neighbors detail

```
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
3   10.1.1.1                Et0/0             11 00:00:08    1  4500  1  0
   Version 12.4/1.2, Retrans: 2, Retries: 2, Waiting for Init, Waiting for Init Ack
    UPDATE seq 350 ser 0-0 Sent 8040 Init Sequenced
2   10.1.1.3                Et0/0             11 02:47:56   22   200  0  339
   Version 12.4/1.2, Retrans: 11, Retries: 0, Prefixes: 10
1   10.2.1.4                Et1/0             10 02:47:56   24   200  0  318
   Version 12.4/1.2, Retrans: 10, Retries: 0, Prefixes: 8
0   10.2.1.3                Et1/0             11 02:47:56   20   200  0  338
   Version 12.4/1.2, Retrans: 11, Retries: 0, Prefixes: 2
```

The output of the **debug eigrp packets terse** command shows that R2 sends the same packet over and over again:

Note: The retry value increases, the Flags value is 0x1 , and the Init bit is set.

<#root>

```
R2#
```

debug eigrp packets terse

```
EIGRP Packets debugging is on
    (UPDATE, REQUEST, QUERY, REPLY, IPXSAP, PROBE, ACK, STUB, SIAQUERY, SIAREPLY)

R2#
EIGRP: Sending UPDATE on Ethernet0/0 nbr 10.1.1.1,
```

retry 14

```
,
```

RTO 5000

```
 AS 1,
```

Flags 0x1

```
, Seq 350/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
EIGRP: Sending UPDATE on Ethernet0/0 nbr 10.1.1.1,
```

retry 15

```
,
```

RTO 5000

```
 AS 1,
```

Flags 0x1

```
, Seq 350/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
```

The hold time does not expire because the Hello packets are sent and received properly:

```
<#root>

R2#

debug eigrp packets hello


EIGRP Packets debugging is on
    (HELLO)

EIGRP: Received HELLO on Ethernet0/0 nbr 10.1.1.1
  AS 1, Flags 0x0, Seq 0/0 idbQ 0/0
```

Restarted Peer

If you observe a peer restarted repeatedly on one router, it indicates that the router receives the initial Update packets from its neighbor. Be aware of the Flag 1 in the received Update packets.

```
<#root>

R2#

debug eigrp packets terse



EIGRP Packets debugging is on
    (UPDATE, REQUEST, QUERY, REPLY, IPXSAP, PROBE, ACK, STUB, SIAQUERY, SIAREPLY)

R2#
EIGRP: Received Sequence TLV from 10.1.1.1
       10.1.1.2
       address matched
      clearing CR-mode
EIGRP: Received CR sequence TLV from 10.1.1.1, sequence 479
EIGRP: Received UPDATE on Ethernet0/0 nbr 10.1.1.1
  AS 1, Flags 0xA, Seq 479/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0,
   not in CR-mode, packet discarded
EIGRP: Received UPDATE on Ethernet0/0 nbr 10.1.1.1
  AS 1,

Flags 0x1

, Seq 478/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0

%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is down:


peer restarted


%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is up:
 new adjacency
```

```
EIGRP: Enqueueing UPDATE on Ethernet0/0 nbr 10.1.1.1 iidbQ un/rely 0/1
 peerQ un/rely 0/0
```

Initial Update Before Hello

Here is an example where the initial Update packet is received before the Hello packet:

```
EIGRP: Received UPDATE on Ethernet0/0 nbr 10.1.1.2
  AS 1, Flags 0x1, Seq 3/0 idbQ 0/0
EIGRP: Neighbor(10.1.1.2) not yet found
```

If this occurs once after a neighbor flap, then this situation is not a problem. However, if you experience it often, it indicates that the unicast on the link is operational, but the multicast on the link is broken. In other words, the router receives the unicast Update packet, but not the Hello packets.

Additional Issues

Some other types of problems include:

- Configuration changes
- Authentication issues
- Mismatches on primary and secondary IP addresses
- DMVPN issues

These issues are explained in greater detail in these next sections.

Configuration Changes

Note: The results of the commands that are used throughout this section are the same if you configure the negation instead (the **no** command).

When you configure the summary statement (or the auto-summary) on the interface, you observe this message on the router:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.3 (Ethernet0/0) is resync:
 summary configured
```

Here is an example that shows the configuration of a global distribute-list for the EIGRP process:

<#root>

```
R1(config-router)#
```

distribute-list 1 out

```
R1(config-router)#
```

This message is observed on the router:

Note: The same occurs when you configure a distribute-list <> in as well.

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.3 (Ethernet0/0) is resync:
 route configuration changed
```

All of the EIGRP neighbors then go down when you configure an interface distribute-list for the EIGRP process:

```
R1(config-router)#distribute-list 1 out ethernet 0/0
```

In this case, only the EIGRP neighborships on this interface are reset.

Note: After Cisco bug ID CSCdy20284, the neighborships are not reset for manual changes such as summarization and filters.

Authentication

Authentication can be misconfigured or missing. This can cause the EIGRP neighborship to go down because of the exceeded retry-limit. Enable the **debug eigrp packets** command in order to confirm that it is the Message Digest 5 (MD5) authentication that causes the issue:

<#root>

```
R1#
```

```
debug eigrp packets
```

```
EIGRP Packets debugging is on
    (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB, SIAQUERY,
     SIAREPLY)

EIGRP: Ethernet0/0: ignored packet from 10.1.1.3, opcode = 1 (missing
 authentication or key-chain missing)
```

Mismatch on Primary and Secondary IP Addresses

The EIGRP sends out the Hello and all other packets from the primary IP address. The packets are accepted from the other router if the source IP addresses falls into the primary IP address range or one of the

secondary IP address ranges on the interface. If not, this error message (when **eigrp log-neighbor-warnings** is enabled) is observed:

```
IP-EIGRP(Default-IP-Routing-Table:1): Neighbor 10.1.1.2 not on common subnet
 for Ethernet0/0
```

DMVPN

Check for IPSec problems in the DMVPN networks. The IPSec can cause the EIGRP to flap if the encryption is not clean:

<#root>

```
 show crypto ipsec sa
```

```
   protected vrf:
   local  ident (addr/mask/prot/port):  (10.10.110.1/255.255.255.255/47/0)
   remote ident (addr/mask/prot/port):  (10.10.101.1/255.255.255.255/47/0)
   current_peer: 144.23.252.1:500
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 190840467, #pkts encrypt: 190840467, #pkts digest 190840467
    #pkts decaps: 158102457, #pkts decrypt: 158102457, #pkts verify 158102457
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0
    #pkts not decompressed: 0, #pkts decompress failed: 0

#send errors 5523, #recv errors 42
```

Flags Explained

There is a 32-bit Flags field in the EIGRP packet header, and it is useful to understand the indications of the various Flag values.

- Flag 0x1 Init bit

  This Flag is set in the initial Update packet.

  <#root>

  ```
  EIGRP: Received UPDATE on Ethernet0/0 nbr 10.1.1.1
   AS 1,
  Flags 0x1
  , Seq 478/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
  ```

- Flag 0x2

This Flag indicates Conditional Receive mode (CR-mode). This is a part of the reliable EIGRP multicast process and is used in order to allow the neighbors who have not acknowledged a previous reliable packet to catch up on a shared link. The addresses in the sequence Type Length Value (TLV) are the peers who must ignore the multicast packets until they catch up via unicast packets.

<#root>

```
EIGRP: Received UPDATE on Ethernet0/0 nbr 10.1.1.2
  AS 1,
```

Flags 0x2

```
, Seq 21/0 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/1,
  not in CR-mode, packet discarded
```

- Flag 0x4

This Flag is the Restart bit (RS bit). It is set in the Hello packets and the Update packets when NSF is signaled. An NSF-aware router views this bit in order to detect if the neighbor router restarts. The neighbor that detects then knows to keep up the EIGRP adjacency. The router that restarts views this Flag in order to determine whether the peer helps with the restart.

<#root>

```
EIGRP: Received HELLO on Ethernet0/0 nbr 10.1.1.2
 AS 1,
```

Flags 0x4

```
, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
```

- Flag 0x8

This is the End-of-Table (EOT) bit. This bit indicates that the complete routing table has been sent to the neighbor. An NSF-capable router views this bit in order to determine whether the neighbor router has completed its restart. An NSF-capable router waits for this bit before it removes stale routes from the router that restarts.

<#root>

```
EIGRP: Received UPDATE on Ethernet0/0 nbr 10.1.1.2
  AS 1,
```

Flags 0x8

```
, Seq 4/33 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
EIGRP: NSF: AS1. Receive EOT from 10.1.1.2
```

The Flags are printed in one HEX number. Thus, Flag 0x5 means that Flags 4 and 1 are set; Flag 0x9 means that Flags 8 and 1 are set; Flag 0xA means that Flags 8 and 2 are set.

You can use these commands in order to troubleshoot flapping neighbors:

- **show eigrp interface detail**
- **show ip eigrp neighbor detail**
- **ping unicast**
- **ping with size full MTU**
- **ping with verify reply data**
- **ping multicast**
- **debug eigrp packet (hello)**
- **show ip eigrp traffic**
- **show ip traffic | begin EIGRP**

SIA

This section provides an overview of the SIA state, some possible symptoms and causes, and how to troubleshoot it.

Definition of SIA

The SIA state means that an EIGRP router has not received a reply to a query from one or more neighbors within the allotted time (approximately three minutes). When this occurs, the EIGRP clears the neighbors that do not send a reply and logs a "DUAL-3-SIA" error message for the route that went active.

Symptoms

These messages can be seen on one or many routers:

```
%DUAL-3-SIA: Route 10.100.1.1/32 stuck-in-active state in IP-EIGRP(0) 1.  Cleaning up

%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 192.168.1.6 (Serial3/0) is down:
 stuck in active
```

If this only occurs sporadically, it can be ignored. If it occurs frequently, it indicates a persistent network problem.

Possible Causes

Here are some possible causes for an SIA state:

- Flapping links
- Bad links
- Flapping routes
- Congested links
- Large network diameter (large query range)
- Memory shortage
- High CPU
- Misconfiguration (wrong bandwidth value)

Troubleshoot Tips

When an SIA situation occurs, there is a problem somewhere in the network. The exact cause can be difficult to discover. There are two approaches:

1. View the prefixes that are consistently reported as SIA and determine the commonalities.
2. Locate the router that consistently fails to answer queries for these routes.

Determine whether all of the prefixes for which SIA is reported have commonalities. For example, they all can be /32 routes from the edge of the network (such as in dial-up networks). If so, it can indicate the problem location in the network (namely, where these prefixes originated).

Ultimately, you must discover the location where one or more routers sends queries and does not receive replies, while the downstream router is not in this state. For example, the router could send queries and they are acknowledged, but the reply from the downstream router is not received.

You can use the **show ip eigrp topology active** command in order to help troubleshoot the SIA issue. Look for the small **r** in the command output. This means that the router awaits a reply to a query for that prefix from that neighbor.

Here is an example. View the topology. The links R1-R6 and R1-R5 are shut down. When the loopback interface of the router R1 is shut down, R1 sends a query for the prefix 10.100.1.1/32 to R2 and R3. The router R1 is now active for this prefix. The routers R2 and R3 go active and query in turn the router R4, which goes active and sends a query to R5. The router R5 finally goes active and sends a query to R6. The router R6 must return a reply to R5. The router R5 goes passive and replies to R4, which in turn goes passive and sends a reply to R2 and R3. Finally, R2 and R3 go passive and send a reply to R1, which goes passive again.

If a problem is encountered, then a router can stay active for an extended time, as it must wait for a reply. In order to prevent the router waiting for a reply that can never be received, the router can declare SIA and kill the neighborship through which it awaits the reply. In order to troubleshoot the problem, view the **show ip eigrp topology active** command output and follow the trail of the **r**.

Here is the output for the router R1:

<#root>

```
R1#
```

```
show ip eigrp topology active
```

```
IP-EIGRP Topology Table for AS 1)/ID(10.100.1.1)

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

A 10.100.1.1/32, 1 successors, FD is Inaccessible
    1 replies, active 00:01:11, query-origin: Local origin
        via Connected (Infinity/Infinity), Loopback0
      Remaining replies:
         via 10.1.1.2,
```

```
r
```

```
, Ethernet0/0
```

The router R1 is active and awaits a reply from R2. Here is the output for the router R2:

<#root>

```
R2#
show ip eigrp topology active


IP-EIGRP Topology Table for AS(1)/ID(10.100.1.2)

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

A 10.100.1.1/32, 1 successors, FD is Inaccessible
    1 replies, active 00:01:01, query-origin: Successor Origin
        via 10.1.1.1 (Infinity/Infinity), Ethernet0/0
        via 10.2.1.4 (Infinity/Infinity),
r
, Ethernet1/0, serno 524
        via 10.2.1.3 (Infinity/Infinity), Ethernet1/0, serno 523
```

The router R2 is active and awaits a reply from R4. Here is the output for the router R4:

```
<#root>
R4#
show ip eigrp topology active


IP-EIGRP Topology Table for AS(1)/ID(10.100.1.4)

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

A 10.100.1.1/32, 1 successors, FD is Inaccessible
    1 replies, active 00:00:56, query-origin: Successor Origin
        via 10.2.1.2 (Infinity/Infinity), Ethernet1/0
        via 172.16.1.5 (Infinity/Infinity),
r
, Serial2/0, serno 562
        via 10.2.1.3 (Infinity/Infinity), Ethernet1/0, serno 560
```

The router R4 is active and awaits a reply from R5. Here is the output for the router R5:

```
<#root>
R5#
show ip eigrp topology active


IP-EIGRP Topology Table for AS(1)/ID(172.16.1.5)

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


r - reply Status
```

```
, s - sia Status

A 10.100.1.1/32, 1 successors, FD is Inaccessible, Q
    1 replies, active 00:00:53, query-origin: Successor Origin
        via 172.16.1.4 (Infinity/Infinity), Serial2/0
      Remaining replies:
         via 192.168.1.6,

r

, Serial3/0
```

The router R5 is active and awaits a reply from R6. Here is the output for the router R6:

<#root>

```
R6#

show ip eigrp topology active


IP-EIGRP Topology Table for AS(1)/ID(192.168.1.6)
R6#
```

As shown, the router R6 is not active for the prefix, so the problem must be between routers R5 and R6. After some time, you see that R5 kills the neighborship to R6 and declares an SIA state:

```
R5#
%DUAL-3-SIA: Route 10.100.1.1/32 stuck-in-active state in IP-EIGRP(0) 1.
  Cleaning up
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 192.168.1.6 (Serial3/0) is down:
 stuck in active
```

When you view the output for router R5, you can see that there are problems on the link towards R6.

This is new SIA code, and as such, the SIA occurred on a router that was next to the problem. In this example, this is the link between the routers R5 and R6. In older code versions, the SIA could be declared on any router along the path (such as on R2), which can be distant from the problem. The SIA timer was three minutes. Any router along the path could be the first to go SIA and kill the neighborship(s). With the newer code, the router awaits a reply, intermediately sends an SIA query to its neighbor, and the neighbor immediately answers with an SIA reply. For example, while in the active state, the router R4 sends an SIA query to R5, and R5 replies with an SIA reply.

```
R5#
EIGRP: Received SIAQUERY on Serial2/0 nbr 172.16.1.4
  AS 1, Flags 0x0, Seq 456/447 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Enqueueing SIAREPLY on Serial2/0 nbr 172.16.1.4 iidbQ un/rely 0/1
  peerQ un/rely 0/0 serno 374-374
EIGRP: Sending SIAREPLY on Serial2/0 nbr 172.16.1.4
  AS 1, Flags 0x0, Seq 448/456 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
  serno 374-374
```

The router R5 also sends SIA queries to R6, but it does not receive a SIA reply from R6.

```
R5#
EIGRP: Enqueueing SIAQUERY on Serial3/0 nbr 192.168.1.6 iidbQ un/rely 0/2
 peerQ un/rely 5/0 serno 60-60
```

Once the router sends an SIA query but does not receive an SIA reply, the **s** appears for that neighbor:

<#root>

```
R5#
```

show ip eigrp topology active

```
IP-EIGRP Topology Table for AS(1)/ID(172.16.1.5)

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

s - sia Status

```
A 10.100.1.1/32, 1 successors, FD is Inaccessible, Qqr
    1 replies, active 00:02:36, query-origin: Successor Origin, retries(1)
        via 1172.16.1.4 (Infinity/Infinity), Serial2/0, serno 61
        via 192.168.1.6 (Infinity/Infinity), r
```

s

```
, q, Serial3/0, serno 60, anchored
```

With the new SIA code, the SIA must be declared on the router R5 when it does not receive an SIA reply. You must then enable the debugging for these two EIGRP SIA packets:

<#root>

```
R2#
```

debug eigrp packets SIAquery SIAreply

```
EIGRP Packets debugging is on
    (SIAQUERY, SIAREPLY)

R2#
```

show debug

```
EIGRP:
  EIGRP Packets debugging is on
    (SIAQUERY, SIAREPLY)
```

In summary, you can use these commands in order to troubleshoot the SIA issue:

- **show ip eigrp topology active**
- **show ip eigrp event** (possibly increase the event log size)
- **show ip eigrp traffic** (search for many SIA queries and SIA replies)
- **show proc mem**
- **show mem sum**

Here are some possible solutions for the SIA issue:

- Fix the link problem.
- Apply summarization (manual or automatic) in networks with many prefixes or a deep query range.
- Use distribute-lists in order to decrease the query range.
- Define remote routers as stubs.

Missing Prefixes

There are two types of missing prefixes: those that are missing in the routing table (or Routing Information Base (RIB)), and those that are missing in the topology table.

Missing Prefixes in RIB

There can be several reasons that a prefix is not included in the RIB:

- The prefix is installed in the routing table by another routing protocol with a lower administrative distance.
- A distribute-list blocks the prefix.
- A split-horizon blocks the prefix.

Prefix Installed by Routing Protocol with Lower Administrative Distance

In this example, the prefix is installed in the routing table by a static route or a routing protocol with a lower administrative distance.

Typically when this occurs, the prefix is in the topology table but has no successor. You can view all of these entries with the **show ip eigrp topology zero-successors** command. The Feasible Distance (FD) must have an infinite value.

Enter the **show ip route <prefix>** command and verify the routing protocols that own the route in the RIB:

<#root>

```
R1#
show ip eigrp topology 192.168.100.6 255.255.255.255


IP-EIGRP (AS 1): Topology entry for 192.168.100.6/32
  State is Passive, Query origin flag is 1,
0 Successor(s), FD is 4294967295


  Routing Descriptor Blocks:
  10.3.1.6 (Serial2/0), from 10.3.1.6, Send flag is 0x0
```

```
      Composite metric is (2297856/128256), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 25000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1



<#root>

R1#

show ip eigrp topology zero-successors


IP-EIGRP Topology Table for AS(1)/ID(10.100.1.1)

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

P 192.168.1.0/24, 0 successors, FD is Inaccessible
        via 10.3.1.6 (2681856/2169856), Serial2/0
P 192.168.100.6/32, 0 successors, FD is Inaccessible
        via 10.3.1.6 (2297856/128256), Serial2/0
```

Distribute-List Blocks the Prefix

The EIGRP is a distance-vector routing protocol. You can use a distribute-list on any router in order to block prefixes. You can use it on an interface in order to stop the prefixes transmission or reception, or you can configure the distribute-list globally under the router EIGRP process in order to apply the routing filter on all of the EIGRP-enabled interfaces.

Here is an example:

```
<#root>

R1#

show running-config | begin router eigrp



router eigrp 1
 network 10.0.0.0


distribute-list 1 in


 no auto-summary
!
access-list 1 deny   192.168.100.6
access-list 1 permit any
```

Missing Prefixes in Topology Table

This section describes some of the reasons that a prefix can be missing from the topology table.

Mask Specification for Proper Command Output

Do not make the typical mistake; when you verify a prefix in the topology table, always specify the mask. This occurs if you do not use the mask:

```
<#root>

R1#

show ip eigrp topology 192.168.100.6


% IP-EIGRP (AS 1): Route not in topology table
```

Here is the **show ip eigrp topology** command output when the mask is specified:

```
<#root>

R1#

show ip eigrp topology 192.168.100.6 255.255.255.255


IP-EIGRP (AS 1): Topology entry for 192.168.100.6/32
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 2297856
  Routing Descriptor Blocks:
  10.3.1.6 (Serial2/0), from 10.3.1.6, Send flag is 0x0
      Composite metric is (2297856/128256), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 25000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
  10.4.1.5 (Ethernet1/0), from 10.4.1.5, Send flag is 0x
      Composite metric is (2323456/2297856), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 26000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
```

As shown, the prefix is present in the topology table.

Split-Horizon Blocks the Prefix

This section describes another common mistake. EIGRP is not a link-state routing protocol, but rather it is a

distance-vector routing protocol. The topology table must be used for the correct operation of Diffuse Update Algorithm (DUAL), not because the EIGRP is a link-state routing protocol; hence, it requires a database. The topology table is required because only the best routes are installed in the routing table, while the DUAL demands that the feasible routes are monitored as well. These are stored in the topology table.

You must always have the successor route and the feasible routes in the topology table. If not, there is a bug. However, there could also be non-feasible routes in the topology table, as long as they are received. If they are not received from a neighbor, there could be a split-horizon that blocks the prefix.

The output of the **show ip eigrp topology** command shows only the prefix entries that point to successors and feasible successors. If you want to view the prefixes that are received over all of the paths (also non-feasible paths), then enter the **show ip eigrp topology all-links** command instead.

Here is an example:

<#root>

```
R1#
show ip eigrp topology


IP-EIGRP Topology Table for AS(1)/ID(10.100.1.1)

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

P 10.3.1.0/24, 1 successors, FD is 2169856
        via Connected, Serial2/0
P 10.2.1.0/24, 2 successors, FD is 307200
        via 10.1.1.2 (307200/281600), Ethernet0/0
        via 10.1.1.3 (307200/281600), Ethernet0/0
P 10.1.1.0/24, 1 successors, FD is 281600
        via Connected, Ethernet0/0
P 172.16.1.0/24, 1 successors, FD is 2195456
        via 10.4.1.5 (2195456/2169856), Ethernet1/0
P 192.168.1.0/24, 1 successors, FD is 2195456
        via 10.4.1.5 (2195456/2169856), Ethernet1/0
        via 10.3.1.6 (2681856/2169856), Serial2/0
P 10.4.1.0/24, 1 successors, FD is 281600
        via Connected, Ethernet1/0
P 172.16.100.5/32, 1 successors, FD is 409600
        via 10.4.1.5 (409600/128256), Ethernet1/0
P 10.100.1.4/32, 2 successors, FD is 435200
        via 10.1.1.2 (435200/409600), Ethernet0/0
        via 10.1.1.3 (435200/409600), Ethernet0/0
P 10.100.1.3/32, 1 successors, FD is 409600
        via 10.1.1.3 (409600/128256), Ethernet0/0
P 10.100.1.2/32, 1 successors, FD is 409600
        via 10.1.1.2 (409600/128256), Ethernet0/0
P 10.100.1.1/32, 1 successors, FD is 128256
        via Connected, Loopback0
P 192.168.100.6/32, 1 successors, FD is 2297856
        via 10.3.1.6 (2297856/128256), Serial2/0
```

In this output you can see that the **all-links** portion of the command includes more paths:

<#root>

R1#

show ip eigrp topology all-links

IP-EIGRP Topology Table for AS(1)/ID(10.100.1.1)

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

P 10.3.1.0/24, 1 successors, FD is 2169856, serno 43
        via Connected, Serial2/0
P 10.2.1.0/24, 2 successors, FD is 307200, serno 127
        via 10.1.1.2 (307200/281600), Ethernet0/0
        via 10.1.1.3 (307200/281600), Ethernet0/0
P 10.1.1.0/24, 1 successors, FD is 281600, serno 80
        via Connected, Ethernet0/0
P 172.16.1.0/24, 1 successors, FD is 2195456, serno 116
        via 10.4.1.5 (2195456/2169856), Ethernet1/0

via 10.3.1.6 (3193856/2681856), Serial2/0

via 10.1.1.2 (2221056/2195456), Ethernet0/0

via 10.1.1.3 (2221056/2195456), Ethernet0/0

P 192.168.1.0/24, 1 successors, FD is 2195456, serno 118
        via 10.4.1.5 (2195456/2169856), Ethernet1/0
        via 10.3.1.6 (2681856/2169856), Serial2/0
P 10.4.1.0/24, 1 successors, FD is 281600, serno 70
        via Connected, Ethernet1/0
P 172.16.100.5/32, 1 successors, FD is 409600, serno 117
        via 10.4.1.5 (409600/128256), Ethernet1/0

via 10.3.1.6 (2809856/2297856), Serial2/0

P 10.100.1.4/32, 2 successors, FD is 435200, serno 128
        via 10.1.1.2 (435200/409600), Ethernet0/0
        via 10.1.1.3 (435200/409600), Ethernet0/0
P 10.100.1.3/32, 1 successors, FD is 409600, serno 115
        via 10.1.1.3 (409600/128256), Ethernet0/0
P 10.100.1.2/32, 1 successors, FD is 409600, serno 109
        via 10.1.1.2 (409600/128256), Ethernet0/0
P 10.100.1.1/32, 1 successors, FD is 128256, serno 4
        via Connected, Loopback0
P 192.168.100.6/32, 1 successors,

FD is 2297856

, serno 135
        via 10.3.1.6 (2297856/128256), Serial2/0

via 10.4.1.5 (2323456/2297856), Ethernet1/0

Consider the last prefix in the previous output; the path via 10.4.1.5 has (2323456/2297856). The reported distance (advertised metric) is 2297856, which is not smaller than the FD of 2297856, so the path is not feasible.

<#root>

```
P 192.168.100.6/32, 1 successors, FD is 2297856, serno 135
        via 10.3.1.6 (2297856/128256), Serial2/0
```

via 10.4.1.5 (2323456/2297856), Ethernet1/0

Here is an example where a split-horizon causes a path to be excluded from the topology table for one route. When you view the topology, you can see that the router R1 has the prefix 192.168.100.6/32 via R6 and R5 in the topology table, but not via R2 or R3:

<#root>

```
R1#
```

show ip eigrp topology 192.168.100.6 255.255.255.255

```
IP-EIGRP (AS 1): Topology entry for 192.168.100.6/32
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 2297856
  Routing Descriptor Blocks:
  10.3.1.6 (Serial2/0), from 10.3.1.6, Send flag is 0x0
      Composite metric is (2297856/128256), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 25000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
  10.4.1.5 (Ethernet1/0), from 10.4.1.5, Send flag is 0x0
      Composite metric is (2323456/2297856), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 26000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
```

This is because the router R1 never received the prefix 192.168.100.6/32 via R2 or R3, as it has the prefix 192.168.100.6/32 via R1 in the routing table.

<#root>

```
R2#
```

show ip route 192.168.100.6 255.255.255.255

```
Routing entry for 192.168.100.6/32
  Known via "eigrp 1", distance 90, metric 2323456, type internal
  Redistributing via eigrp 1
  Last update from 10.1.1.1 on Ethernet0/0, 00:02:07 ago
  Routing Descriptor Blocks:
  * 10.1.1.1, from 10.1.1.1, 00:02:07 ago, via Ethernet0/0
      Route metric is 2323456, traffic share count is 1
      Total delay is 26000 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2

R3#
```

```
show ip route 192.168.100.6 255.255.255.255
```

```
Routing entry for 192.168.100.6/32
  Known via "eigrp 1", distance 90, metric 2323456, type internal
  Redistributing via eigrp 1
  Last update from 10.1.1.1 on Ethernet0/0, 00:01:58 ago
  Routing Descriptor Blocks:
  * 10.1.1.1, from 10.1.1.1, 00:01:58 ago, via Ethernet0/0
      Route metric is 2323456, traffic share count is 1
      Total delay is 26000 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

In order to verify this, use the **all-links** keyword on R1 when you view the topology table. This shows all of the paths for all of the prefixes, which includes the non-feasible paths. You can then see that the prefix 192.168.100.6/32 has not been learned by the router R1 from R2 or R3.

Metrics

 **Note**: The MTU and hop count are not included in the metric calculation.

These are the formulas that are used in order to calculate the path metric of a route:

- If K5 is a non-zero value:

  EIGRP Metric = 256*(((K1*Bw) + (K2*Bw)/(256-Load) + (K3*Delay))*(K5/(Reliability + K4)))

- If K5 is equal to zero:

  EIGRP Metric = 256*((K1*Bw) + (K2*Bw)/(256-Load) + (K3*Delay))

The K-values are weights that are used in order to weight the four components of the EIGRP metric: delay, bandwidth, reliability, and load. These are the default K-values:

- K1 = 1
- K2 = 0
- K3 = 1
- K4 = 0

- K5 = 0

With the default K-values (only with bandwidth and delay), the formula becomes:

EIGRP Metric = 256 * (Bw + Delay)

Bw= (10^7/minimum Bw in kilobits per second)

Note: The delay is measured in tens of microseconds; however, on the interface, it is measured in microseconds.

All of the four components can be verified with the **show interface** command:

<#root>

R1#

show interface et 0/0

```
  Ethernet0/0 is up, line protocol is up
  Hardware is AmdP2, address is aabb.cc00.0100 (bia aabb.cc00.0100)
  Internet address is 10.1.1.1/24
  MTU 1500 bytes,
```

BW 10000 Kbit

,

DLY 1000 usec

,

reliability 255/255

,

txload 1/255

,

rxload 1/255

```
  Encapsulation ARPA, loopback not set  Keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 04:00:00  Last input 00:00:02, output 00:00:02,
   output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     789 packets input, 76700 bytes, 0 no buffer
     Received 707 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 input packets with dribble condition detected
     548 packets output, 49206 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
```

```
     0 unknown protocol drops
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
```

The delay is cumulative, which means that you add the delay of each link along the path. The bandwidth is not cumulative, so the bandwidth that is used in the formula is the smallest bandwidth of any link along the path.

Duplicate Router ID

In order to view the router ID that the EIGRP uses, enter the **show ip eigrp topology** command on the router and view the first line of the output:

```
<#root>

R1#

show ip eigrp topology


IP-EIGRP Topology Table for AS(1)/

ID(10.100.1.1)



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

P 10.3.1.0/24, 1 successors, FD is 2169856
        via Connected, Serial2/0
```

The EIGRP router ID is not used at all for internal routes in older Cisco IOS versions. A duplicate router ID for the EIGRP must not cause any problems if only internal routes are used. In newer Cisco IOS software, the EIGRP internal routes do carry the EIGRP router ID.

The router ID for external routes can be viewed in this output:

```
<#root>

R1#

show ip eigrp topology 192.168.1.4 255.255.255.255


IP-EIGRP (AS 1): Topology entry for 192.168.1.4/32
  State is Passive, Query origin flag is 1, 2 Successor(s), FD is 435200
  Routing Descriptor Blocks:
  10.1.1.2 (Ethernet0/0), from 10.1.1.2, Send flag is 0x0
      Composite metric is (435200/409600), Route is External
      Vector metric:
        Minimum bandwidth is 10000 Kbit
        Total delay is 7000 microseconds
        Reliability is 255/255
```

```
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
      External data&colon;
```

Originating router is 10.100.1.4

```
        AS number of route is 0
        External protocol is Connected, external metric is 0
        Administrator tag is 0 (0x00000000)
```

If an (external) EIGRP route with the same EIGRP router ID as the router is received, it does not generate a log entry. However, the EIGRP event log does capture this. When you check for the (external) EIGRP route, it does not show up in the topology table.

Check the EIGRP event log for possible duplicate router ID messages:

<#root>

R1#

show ip eigrp events

```
Event information for AS 1:
1    08:36:35.303 Ignored route, metric: 10.33.33.33 3347456
2    08:36:35.303 Ignored route, neighbor info: 10.3.1.6 Serial2/1
3    08:36:35.303
```

Ignored route, dup router: 10.100.1.1

```
4    08:36:35.303 Rcv EOT update src/seq: 10.3.1.6 143
5    08:36:35.227 Change queue emptied, entries: 2
6    08:36:35.227 Route OBE net/refcount: 10.100.1.4/32 3
7    08:36:35.227 Route OBE net/refcount: 10.2.1.0/24 3
8    08:36:35.227 Metric set: 10.100.1.4/32 435200
9    08:36:35.227 Update reason, delay: nexthop changed 179200
10   08:36:35.227 Update sent, RD: 10.100.1.4/32 435200
11   08:36:35.227 Route install: 10.100.1.4/32 10.1.1.3
12   08:36:35.227 Route install: 10.100.1.4/32 10.1.1.2
13   08:36:35.227 RDB delete: 10.100.1.4/32 10.3.1.6
```

K-Values Mismatch/Graceful Shutdown

When the K-values are not the same on neighbor routers, this message is observed:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.4.1.5 (Ethernet1/0) is down:
 K-value mismatch
```

The K-values are configured with this command (with the possible values of K between 0 and 255):

<#root>

```
metric weights

 tos k1 k2 k3 k4 k5

!
router eigrp 1
 network 10.0.0.0
 metric weights 0 1 2 3 4 5
!
```

The message indicates that the EIGRP neighborship is not established because of a mismatch in K-values. The K-values must be the same on all of the EIGRP routers in one autonomous system in order to prevent routing problems when different routers use different metric calculations.

Check whether the K-values are the same on the neighbor routers. If the K-values are the same, the issue can be caused by the EIGRP graceful shutdown feature. In that case, a router sends an EIGRP Hello packet with the K-values set to 255 so that the K-values mismatch occurs intentionally. This is to indicate to the neighbor EIGRP router that goes down. On the neighbor router, you would see this goodbye message received:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.1.1 (Ethernet0/0) is down:
 Interface Goodbye received
```

However, if the neighbor router runs an older code version (prior to Cisco bug ID CSCdr96531), it does not recognize this as a graceful shutdown message, but as a mismatch in K-values:

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.4.1.5 (Ethernet1/0) is down:
 K-value mismatch
```

This is the same message as in the case of a true K-values mismatch on the neighbor routers.

These are the triggers for a graceful shutdown:

- The **no router eigrp** command is entered.
- The **no network** command is entered.
- The **clear ip eigrp neighbor** command is entered.
- The router is reloaded.

A graceful shutdown is used in order to speed up the detection of a neighbor down state. Without a graceful shutdown, a neighbor must wait until the hold time expires before it declares the neighbor to be down.

Unequal Cost Load Balancing (Variance)

Unequal cost load balancing is possible in EIGRP with the **variance** command, but both the variance and feasibility conditions must be met.

The variance condition means that the metric of the route is not larger than the best metric multiplied by the variance. In order for a route to be considered feasible, the route must have been advertised with a reported distance that is lower than the Feasible Distance (FD). Here is an example:

<#root>

```
!
router eigrp 1

variance 2

 network 10.0.0.0
 no auto-summary
!
```

The router R1 has a variance 2 configured. This means that if the router has a another path for the route with a metric that is not larger than twice the best metric for that route, there must be unequal cost load balancing for that route.

<#root>

```
R1#

show ip eigrp topology 172.16.100.5 255.255.255.255

IP-EIGRP (AS 1): Topology entry for 172.16.100.5/32
  State is Passive, Query origin flag is 1, 1 Successor(s),

FD is 409600

  Routing Descriptor Blocks:
  10.4.1.5 (Ethernet1/0), from 10.4.1.5, Send flag is 0x0
      Composite metric is (

409600

/128256), Route is Internal
      Vector metric:
        Minimum bandwidth is 10000 Kbit
        Total delay is 6000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
  10.3.1.6 (Serial2/0), from 10.3.1.6, Send flag is 0x0
      Composite metric is (

435200/409600

), Route is Internal <<< RD = 409600
      Vector metric:
        Minimum bandwidth is 10000 Kbit
        Total delay is 7000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
```

```
        Hop count is 2
```

If the second topology entry is installed in the routing table, the metric of the second topology entry is 435200. Since twice the best metric is 2 x 409600 = 819200, and 435200 < 819200, the second topology entry is within the variance range. The reported distance of the second topology entry is 409600, which is not smaller than the FD = 409600. The second condition (feasibility) is not met, and the second entry cannot be installed in the RIB.

<#root>

R1#

show ip route 172.16.100.5

```
Routing entry for 172.16.100.5/32
  Known via "eigrp 1", distance 90, metric 409600, type internal
  Redistributing via eigrp 1
  Last update from 10.4.1.5 on Ethernet1/0, 00:00:16 ago
  Routing Descriptor Blocks:
  * 10.4.1.5, from 10.4.1.5, 00:00:16 ago, via Ethernet1/0
      Route metric is 409600, traffic share count is 1
      Total delay is 6000 microseconds, minimum bandwidth is 10000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

If the RD of the second topology entry is smaller then the FD, as in the next example, there would be unequal cost load balancing.

<#root>

R1#

show ip eigrp topology 172.16.100.5 255.255.255.255

```
IP-EIGRP (AS 1): Topology entry for 172.16.100.5/32
  State is Passive, Query origin flag is 1, 1 Successor(s),
```

FD is 409600

```
  Routing Descriptor Blocks:
  10.4.1.5 (Ethernet1/0), from 10.4.1.5, Send flag is 0x0
      Composite metric is (409600/128256), Route is Internal
      Vector metric:
        Minimum bandwidth is 10000 Kbit
        Total delay is 6000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
  10.3.1.6 (Serial2/0), from 10.3.1.6, Send flag is 0x0
      Composite metric is (434944/
```

409344

```
), Route is Internal <<< RD = 409344
      Vector metric:
        Minimum bandwidth is 10000 Kbit
        Total delay is 6990 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
```

Both topology entries are now in the routing table:

<#root>

```
R1#
```

show ip route 172.16.100.5

```
Routing entry for 172.16.100.5/32
  Known via "eigrp 1", distance 90, metric 409600, type internal
  Redistributing via eigrp 1
  Last update from 10.3.1.6 on Serial2/0, 00:00:26 ago
  Routing Descriptor Blocks:
  * 10.4.1.5, from 10.4.1.5, 00:00:26 ago, via Ethernet1/0
      Route metric is 409600, traffic share count is 120
      Total delay is 6000 microseconds, minimum bandwidth is 10000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
    10.3.1.6, from 10.3.1.6, 00:00:26 ago, via Serial2/0
      Route metric is 434944, traffic share count is 113
      Total delay is 6990 microseconds, minimum bandwidth is 10000 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

Static Neighbors

The EIGRP supports configurations with one or more static neighbors on the same interface. As soon as you configure one static EIGRP neighbor on the interface, the router no longer sends the EIGRP packets as multicast on that interface or processes the received multicast EIGRP packets. This means that the Hello, Update, and Query packets are now unicasted. No additional neighborships can be formed unless the **static neighbor** command is explicitly configured for those neighbors on that interface.

This is how to configure a static EIGRP neighbor:

<#root>

```
router eigrp 1
 passive-interface Loopback0
 network 10.0.0.0
 no auto-summary
```

neighbor 10.1.1.1 Ethernet0/0

```
!
```

When the routers on both sides of the link have the **static neighbor** command, the neighborship is formed:

<#root>

```
R1#
```

show ip eigrp neighbors detail

```
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
1   10.1.1.2                Et0/0             14 00:00:23   27   200  0  230
```

Static neighbor

```
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 1
0   10.3.1.6                Se2/0              14 1d02h      26   200  0  169
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 12
3   10.4.1.5                Et1/0              10 1d02h      16   200  0  234
   Version 12.4/1.2, Retrans: 0, Retries: 0, Prefixes: 7
```

If only one router has the **static neighbor** command configured, you can observe that the router ignores the multicast EIGRP packets and the other router ignores the unicasted EIGRP packets:

```
R1#
EIGRP: Received HELLO on Ethernet0/0 nbr 10.1.1.2
  AS 1, Flags 0x0, Seq 0/0 idbQ 0/0
EIGRP: Ignore multicast Hello Ethernet0/0 10.1.1.2
```

```
R2#
EIGRP: Received HELLO on Ethernet0/0 nbr 10.1.1.1
  AS 1, Flags 0x0, Seq 0/0 idbQ 0/0
EIGRP: Ignore unicast Hello from Ethernet0/0 10.1.1.1
```

There is a special debug command for EIGRP static neighbors:

<#root>

```
R2#
```

debug eigrp neighbors static

```
EIGRP Static Neighbors debugging is on
```

```
R2#
```

```
conf t


Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)#router eigrp 1
R2(config-router)#neighbor 10.1.1.1 et 0/0
R2(config-router)#end
R2#

EIGRP: Multicast Hello is disabled on Ethernet0/0!
EIGRP: Add new static nbr 10.1.1.1 to AS 1 Ethernet0/0
```

Here are some reasons that static EIGRP neighbors can be configured:

- You want to limit or avoid broadcasts on Non-Broadcast Multi-Access (NBMA) networks.
- You want to limit or avoid multicasts on broadcast media (Ethernet).
- Use to troubleshoot (with unicast instead of multicast).

⚠ **Caution**: Do not configure the **passive-interface** command together with the **static EIGRP neighbor** command.

Static Route Redistribution

When you configure a static route that points to an interface, and the route is covered by a network statement under the router EIGRP, then the static route is advertised by the EIGRP as if it were a connected route. The **redistribute static** command or a default metric is not required in this case.

```
router eigrp 1
 network 10.0.0.0
 network 172.16.0.0
 no auto-summary
!
ip route 172.16.0.0 255.255.0.0 Serial2/0
!
```

<#root>

```
R1#

show ip eigrp top 172.16.0.0 255.255.0.0


IP-EIGRP (AS 1): Topology entry for 172.16.0.0/16
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 2169856
  Routing Descriptor Blocks:
  0.0.0.0, from Rstatic, Send flag is 0x0
      Composite metric is (2169856/0),

Route is Internal


      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 20000 microseconds
```

```
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 0
```

Reliability and Load for Metric Calculation

⚠ **Caution**: Do not use reliability and/or load in order to calculate metrics.

The reliability and load parameters appear in the **show interface** command output. There are no dynamic updates for these parameters when the load and reliability change. If the load and reliability change, it does not trigger an immediate change in the metric. Only if the EIGRP decides to send updates to its neighbors because of topology changes change in load and reliability be propagated can happen. Furthermore, use of load and reliability in order to calculate the metric can introduce instability, as adaptive routing is then performed. If you desire to change the routing in accordance with the traffic load, then you must consider the use of Multiprotocol Label Switching (MPLS) traffic engineering or Performance Routing (PfR).

High CPU

There are three EIGRP processes that run simultaneously:

1. Router – This process holds the shared memory pools.
2. Hello – This process sends and receives the Hello packets and maintains the peer connections.
3. Protocol Dependent Module (PDM) – The EIGRP supports four protocol suites: IP, IPv6, IPX, and AppleTalk. Each suite has its own PDM. Here are the primary functions of the PDM:

    - Maintains the neighbor and topology tables of the EIGRP routers that belong to that protocol suite.
    - Builds and translates protocol-specific packets for DUAL (transmission and reception of EIGRP packets).
    - Interfaces DUAL to the protocol-specific routing table.
    - Computes the metric and pass the information to DUAL (DUAL only picks the successors and feasible successors).
    - Implements filtering and access-lists.
    - Performs redistribution functions to and from the other routing protocols.

Here is an example output that shows these three processes:

<#root>

```
R1#
```

show process cpu | include EIGRP

```
  89           4        24        166  0.00%  0.00%  0.00%   0 IP-EIGRP
```

Router

```
  90        1016      4406        230  0.00%  0.03%  0.00%   0 IP-EIGRP:
```

PDM

```
  91        2472      6881        359  0.00%  0.07%  0.08%   0 IP-EIGRP:
HELLO
```

High CPU in the EIGRP is not normal. If this occurs, either the EIGRP has too much to do or there is a bug in EIGRP. In the first case, check the number of prefixes in the topology table and the number of peers. Check for instability among the EIGRP routes and neighbors.

EIGRP in Frame Relay Networks (Broadcast Queue)

In frame relay networks where there are multiple neighbor routers on one point-to-multipoint interface, there can be many broadcast or multicast packets that must be transmitted. For this reason, there is a separate broadcast queue with its own buffers. The broadcast queue has priority when it transmits at a rate under the configured maximum and has a guaranteed minimum bandwidth allocation.

Here is the command that is used in this scenario:

<#root>

```
frame-relay broadcast-queue size byte-rate packet-rate
```

As a general rule, begin with twenty packets per Data Link Connection Identifier (DLCI). The byte rate must be less than both of these:

- N/4 times the minimum remote access rate (measured in bytes per second), where N is the number of DLCIs to which the broadcast must be replicated.
- One quarter of the local access rate (measured in bytes per second).

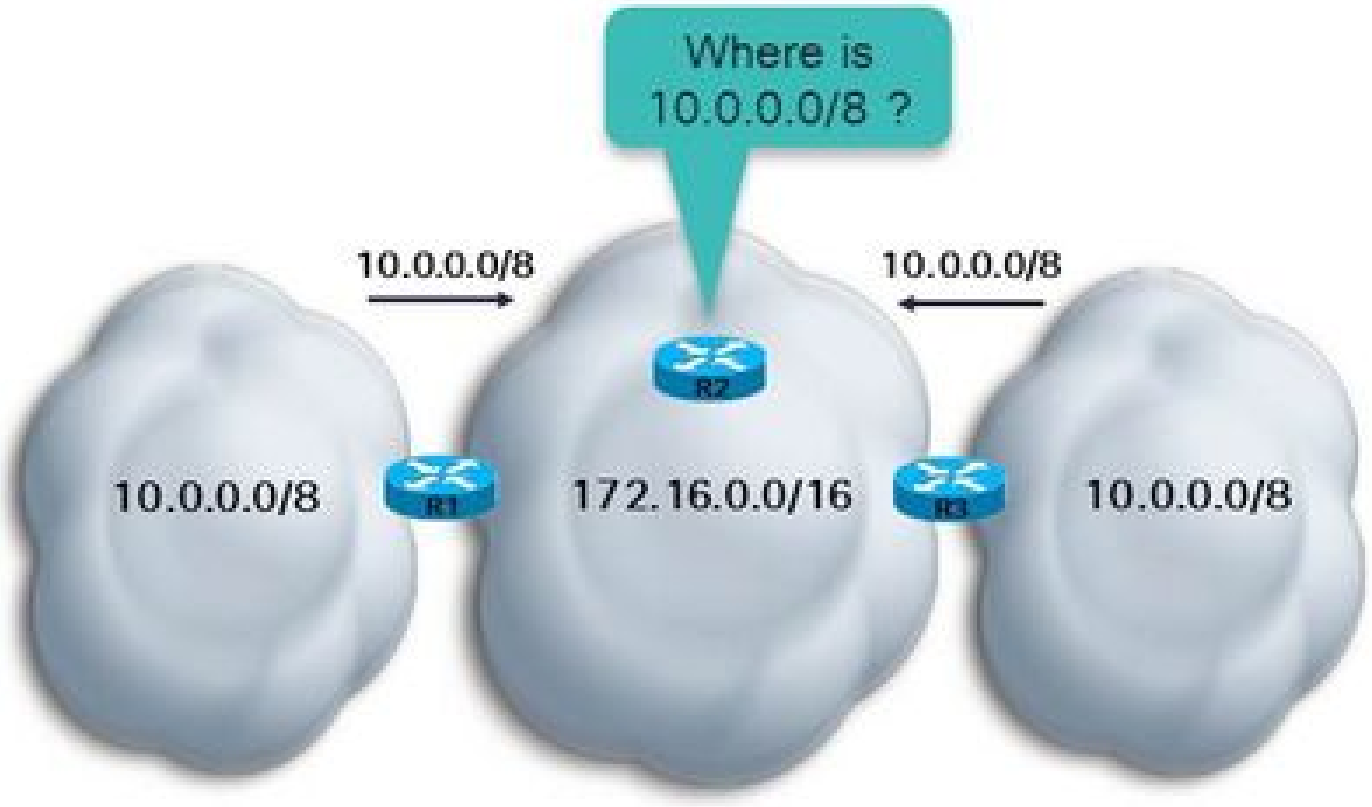
If you observe a large number of EIGRP neighbors flapping, increase the frame-relay broadcast-queue size. This issue is not present if there are frame relay subinterfaces because each neighbor router is on one subinterface with a different IP subnet. Consider this as a workaround when there is a large, fully-meshed frame relay network.

Mismatched AS Numbers

When you enter the **debug eigrp packets hello** command, it reveals that the router does not receive the Hello packets.

Auto-Summary

The EIGRP used to perform summarization at the major network (networks A, B, and C) boundaries by default. This means that more specific routes than the /8 prefixes for the major network type A, more specific routes than the /16 prefixes for major network type B, and more specific routes than the /24 prefixes for major networks type C, are lost when they cross their boundaries. Here is an example where auto-summary causes a problem:



As shown, the routers R1 and R3 have auto-summary under router EIGRP. The router R2 receives 10.0.0.0/8 from both routers R2 and R3 because both R2 and R3 are boundary routers between major class A network 10.0.0.0/8 and 172.16.0.0/16. The router R2 can have the route 10.0.0.0/8 via R1 and R3 if the metric happens to be the same. Otherwise, R2 has the route 10.0.0.0/8 either via R1 or via R3, dependent upon the path that produces the least cost. In either case, if R2 must send traffic to certain subnets of 10.0.0.0/8, it cannot be completely sure that the traffic reaches its destination, as one subnet of 10.0.0.0/8 can be only on either the left or the right network cloud.

In order to alleviate this problem, simply type **no auto-summary** under the router EIGRP process. The router then propagates subnets of the major networks across the boundary. In newer Cisco IOS versions, the no auto-summary setting is the default behavior.

EIGRP Event Log

The EIGRP event log captures the EIGRP events. It is similar to when debugs are enabled for EIGRP. However, it is less disruptive and runs by default. It can be used in order to capture events that are more difficult to troubleshoot or more intermittent events. This log is by default only 500 lines. In order to increase it, enter the **eigrp event-log-size <0 – 209878>** command. You can increase the log size as much as desired, but keep in mind the amount of memory that the router has to spare for this log. In order to clear the EIGRP event log, enter the **clear ip eigrp events** command.

Here is an example:

<#root>

```
R1#
```

```
show ip eigrp events
```

```
Event information for AS 1:
```

```
1    09:01:36.107 Poison squashed: 10.100.1.3/32 reverse
2    09:01:35.991 Update ACK: 10.100.1.4/32 Serial2/0
3    09:01:35.967 Update ACK: 10.100.1.4/32 Ethernet0/0
4    09:01:35.967 Update ACK: 10.100.1.4/32 Ethernet1/0
5    09:01:35.943 Update delay/poison: 179200 FALSE
6    09:01:35.943 Update transmitted: 10.100.1.4/32 Serial2/0
7    09:01:35.943 Update delay/poison: 179200 TRUE
8    09:01:35.943 Update transmitted: 10.100.1.4/32 Ethernet0/0
9    09:01:35.943 Update delay/poison: 179200 FALSE
10   09:01:35.943 Update transmitted: 10.100.1.4/32 Ethernet1/0
11   09:01:35.923 Update packetized: 10.100.1.4/32 Ethernet0/0
12   09:01:35.923 Update packetized: 10.100.1.4/32 Ethernet1/0
13   09:01:35.923 Update packetized: 10.100.1.4/32 Serial2/0
14   09:01:35.903 Change queue emptied, entries: 1
15   09:01:35.903 Route OBE net/refcount: 10.100.1.4/32 3
16   09:01:35.903 Metric set: 172.16.1.0/24 2195456
17   09:01:35.903 Route install: 172.16.1.0/24 10.4.1.5
18   09:01:35.903 FC sat rdbmet/succmet: 2195456 2169856
19   09:01:35.903 FC sat nh/ndbmet: 10.4.1.5 2195456
20   09:01:35.903 Find FS: 172.16.1.0/24 2195456
```

The most recent events appear at the top of the log. You can filter certain types of EIGRP events, such as DUAL, Xmit, and transport:

```
eigrp log-event-type {dual | xmit | transport}
```

Additionally, you can enable logging for one of these three types, a combination of two types, or for all three. Here is an example where two types of logging are enabled:

<#root>

```
router eigrp 1
 redistribute connected
 network 10.0.0.0
 no auto-summary
```

eigrp log-event-type dual xmit

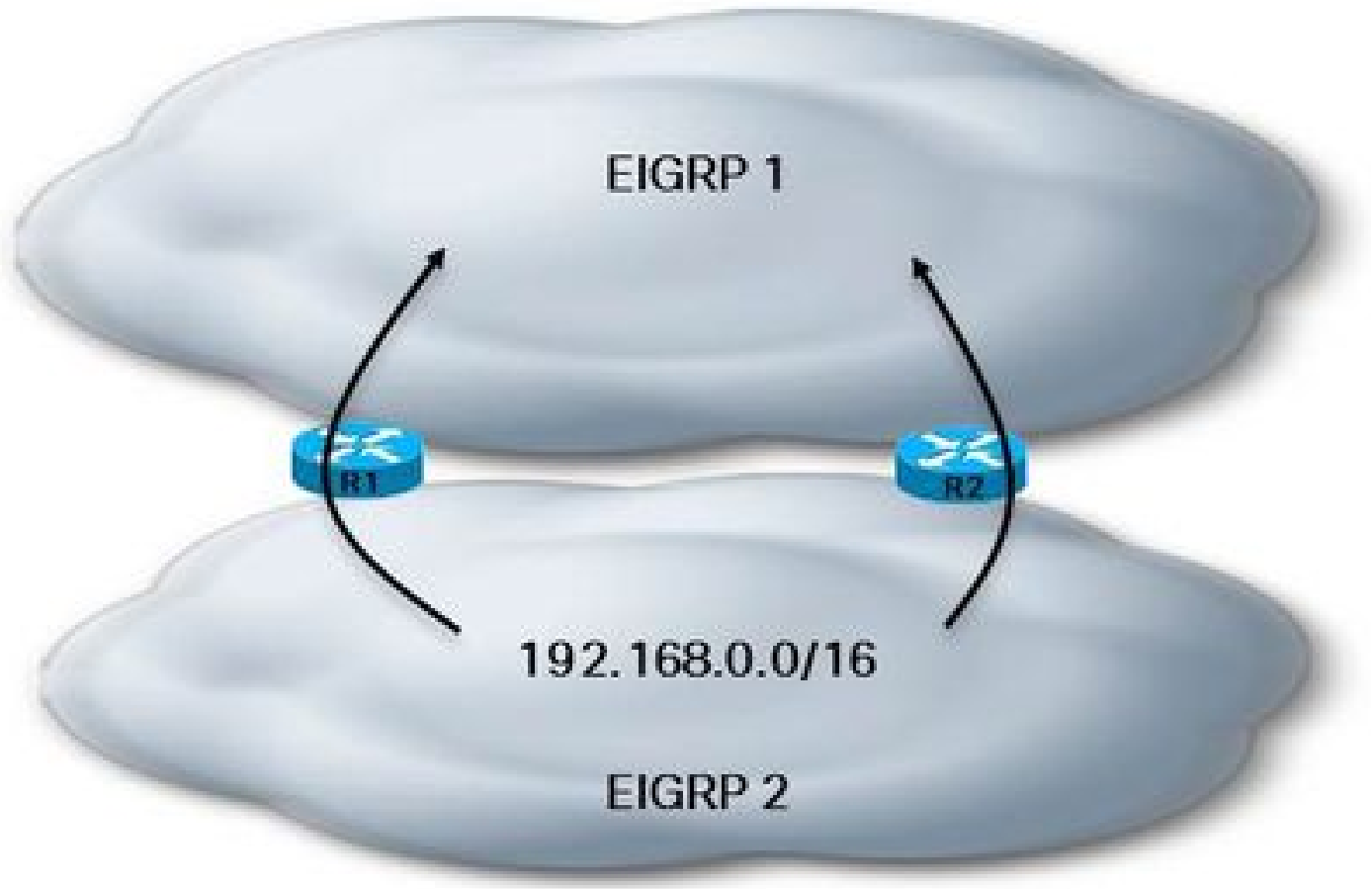
```
 eigrp event-logging
 eigrp event-log-size 100000
!
```

⚠ **Caution**: When you enable **eigrp event-logging**, it prints the event logging and stores it in the event table. This can lead to a large amount of printed output on the console, similar to when heavy EIGRP debugging is enabled.

Same Network Learned by Two EIGRP Autonomous Systems

If a route is learned via two EIGRP processes, then only one of the EIGRP processes can install the route in the RIB. The process with the lowest administrative distance installs the route. If the administrative distance is the same, then the process with the lowest metric installs the route. If the metric is the same as well, then the EIGRP process with the lowest EIGRP process ID install the route in the RIB. The topology table of the other EIGRP process can have the route installed with zero successors and an infinite FD value.

Here is an example:



<#root>

R1#

show ip eigrp topology 192.168.1.0 255.255.255.0

IP-EIGRP (

AS 1

): Topology entry for 192.168.1.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 2681856
  Routing Descriptor Blocks:
  10.3.1.6 (Serial2/0), from 10.3.1.6, Send flag is 0x0
      Composite metric is (2681856/2169856), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 40000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1

```
IP-EIGRP (
```

AS 2

```
): Topology entry for 192.168.1.0/24
  State is Passive, Query origin flag is 1,
```

0 Successor(s)

```
,
```

FD is 4294967295

```
  Routing Descriptor Blocks:
  10.4.1.5 (Ethernet1/0), from 10.4.1.5, Send flag is 0x0
      Composite metric is (2681856/2169856), Route is Internal
      Vector metric:
        Minimum bandwidth is 1544 Kbit
        Total delay is 40000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
```

<#root>

```
R1#
```

show ip route 192.168.1.0 255.255.255.0

```
Routing entry for 192.168.1.0/24
  Known via "eigrp 1", distance 90, metric 2681856, type internal
  Redistributing via eigrp 1
  Last update from 10.3.1.6 on Serial2/0, 00:04:16 ago
  Routing Descriptor Blocks:
  * 10.3.1.6, from 10.3.1.6, 00:04:16 ago, via Serial2/0
      Route metric is 2681856, traffic share count is 1
      Total delay is 40000 microseconds, minimum bandwidth is 1544 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 1
```

Related Information

- Cisco Technical Support & Downloads