

Redistributing Routing Protocols

Table of Contents

<u>Redistributing Routing Protocols</u>	1
<u>Introduction</u>	1
<u>Metrics</u>	1
<u>Administrative Distance</u>	2
<u>Redistribution Configuration Syntax and Examples</u>	3
<u>IGRP and EIGRP</u>	3
<u>OSPF</u>	3
<u>RIP</u>	4
<u>ISIS</u>	4
<u>Connected Routes</u>	5
<u>Avoiding Problems Due to Redistribution</u>	6
<u>Tools Information</u>	11
<u>Related Information</u>	11

Redistributing Routing Protocols

Introduction

Metrics

Administrative Distance

Redistribution Configuration Syntax and Examples

IGRP and EIGRP

OSPF

RIP

ISIS

Connected Routes

Avoiding Problems Due to Redistribution

Tools Information

Related Information

Introduction

Using a routing protocol to advertise routes that are learned by some other means, such as by another routing protocol, static routes, or directly connected routes, is called redistribution. While running a single routing protocol throughout your entire IP internetwork is desirable, multi-protocol routing is common for a number of reasons, including company mergers, multiple departments managed by multiple network administrators, and multi-vendor environments. Often, running different routing protocols is part of a network design. In any case, having a multiple protocol environment makes redistribution a necessity.

Differences in routing protocol characteristics, such as metrics, administrative distance, classful and classless capabilities can effect redistribution. Consideration must be given to these differences in order for redistribution to be successful.

Metrics

When you redistribute one protocol into another, remember that the metrics of each protocol play an important role in redistribution. Each protocol uses different metrics. For example, the RIP metric is based on hop count while the IGRP/EIGRP metric is based on bandwidth and delay. When routes are redistributed, you must define a metric that is understandable to the receiving protocol. There are two ways of defining metrics when redistributing routes.



You can define the metric for that specific redistribution only:

```
router rip
 redistribute static metric 1
 redistribute ospf 1 metric 1
```

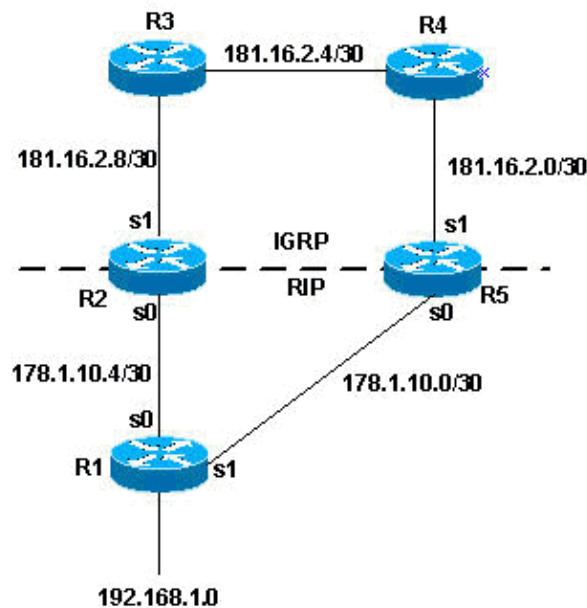
Or you can use the same metric as a default for all redistribution (Using the **default-metric** command saves work since it eliminates the need for defining the metric separately for each redistribution.):

```
router rip
 redistribute static
 redistribute ospf 1
 default-metric 1
```

Administrative Distance

If a router is running more than one routing protocol and learns a route to the same destination using both routing protocols, then which route should be selected as the best route? Each protocol uses its own metric type to determine the best route. Comparing routes with different metric types cannot be done. Administrative distances take care of this problem. Administrative distances are assigned to route sources so that the route from the most preferred source will be chosen as the best path. For more information about administrative distances and route selection, see *Route Selection in Cisco Routers*.

Administrative distances help with route selection among different routing protocols, but they can cause problems for redistribution. These problems can be in the form of routing loops, convergence problems, or inefficient routing. See below for a topology and description of a possible problem.



In the above topology, if R1 is running RIP, and R2 and R5 are running both RIP and IGRP and redistributing RIP into IGRP, then there is a potential problem. For example, R2 and R5 are both learning about network 192.168.1.0 from R1 using RIP. This knowledge is redistributed into IGRP. R2 learns about network 192.168.1.0 through R3, and R5 learns about it from R4 using IGRP. IGRP has a lower administrative distance than RIP (100 versus 120); therefore, the IGRP route is what is used in the routing table. Now there is a potential routing loop. Even if split horizon, or any other feature meant to help prevent routing loops, comes into play, there is still a convergence problem.

If R2 and R5 are also redistributing IGRP into RIP (otherwise known as mutual redistribution) and the network, 192.168.1.0, is not directly connected to R1 (R1 is learning from another router upstream from it), then there is a potential problem that R1 will learn the network from R2 or R5 with a better metric than from the original source.

Tips on avoiding this problem are mention below in the Avoiding Problems Due to Redistribution section.

Redistribution Configuration Syntax and Examples

The following examples illustrate the syntax of redistribution. The examples show many protocols being redistributed between each other. The fewer routing protocol processes running on a router the better, since it increases the risk of creating a routing loop or simply exhausting the router resources and CPU. However, there are some situations where a network administrator has to route more than one protocol and redistribute between them. The examples below demonstrate the syntax for handling this situation. Avoiding problems due to redistribution is discussed at the end of this document.

IGRP and EIGRP

The following output shows an IGRP/EIGRP router redistributing static, OSPF, RIP, and ISIS routes.

```
router igrp/eigrp 1
network 131.108.0.0
redistribute static
redistribute ospf 1
redistribute rip
redistribute isis
default-metric 10000 100 255 1 1500
```

IGRP and EIGRP need five metrics when redistributing other protocols: bandwidth, delay, reliability, load and maximum transmission unit (MTU) respectively. An example of IGRP metrics follows:

Metric	Value
bandwidth	10000 for Ethernet. Minimum bandwidth of the route may be 9.6.
delay	100 x 10 microseconds = 1 ms.
reliability	255 for 100% reliability.
load	Effective bandwidth of the route expressed as a number from 0 to 255 (255 is 100 percent loading).
MTU	Minimum MTU of the router, usually equals the Ethernet bandwidth.

Multiple IGRP and EIGRP processes can run on the same router, with redistribution between them. For example, IGRP1 and IGRP2 can run on the same router. However, running two processes of the same protocol on the same router is rarely necessary, and it can consume the router's memory and CPU.

The redistribution of IGRP/EIGRP into another IGRP/EIGRP process doesn't require any metric conversion, so there is no need to define metrics or use the **default-metric** command during redistribution.

OSPF

The following output shows an OSPF router redistributing static, RIP, IGRP, EIGRP, and ISIS routes.

```
router ospf 1
network 131.108.0.0 0.0.255.255 area 0
redistribute static metric 200 subnets
redistribute rip metric 200 subnets
redistribute igrp 1 metric 100 subnets
redistribute eigrp 1 metric 100 subnets
```

```
redistribute isis metric 10 subnets
```

The OSPF metric is based on 10^8 /bandwidth of the link. For example, the OSPF cost of Ethernet is 10: $10^8/10^7 = 10$

Note: If no metric is specified, OSPF puts a default value of 20 when redistributing routes from all protocols except Border Gateway Protocol (BGP) routes, which get a metric of 1.

Whenever there is a major net that is subnetted, you need to use the keyword *subnet* to redistribute protocols into OSPF. Without this keyword, OSPF only redistributes major nets that aren't subnetted.

It's possible to run more than one OSPF process on the same router, but as mentioned before, running more than one process of the same protocol is rarely needed, and it consumes the router's memory and CPU.

You don't need to define metric or use the **default-metric** command when redistributing one OSPF process into another.

RIP

Note: The principles in this document apply to RIP versions I and II.

The following output shows a RIP router redistributing static, IGRP, EIGRP, OSPF, and ISIS routes.

```
router rip
network 131.108.0.0
redistribute static
redistribute igrp 1
redistribute eigrp 1
redistribute ospf 1
redistribute isis
default-metric 1
```

The RIP metric is composed of hop count, and the maximum valid metric is 15. Anything above 15 is considered infinite; you can use 16 to describe an infinite metric in RIP. When redistributing a protocol into RIP, we recommend using a low metric, such as 1. A high metric, such as 10, limits RIP even further: If we define a metric of 10 for redistributed routes, these routes can only be advertised to routers up to 5 hops away, at which point the metric (hop count) exceeds 15. By defining a metric of 1, you enable a route to travel the maximum number of hops in a RIP domain.

ISIS

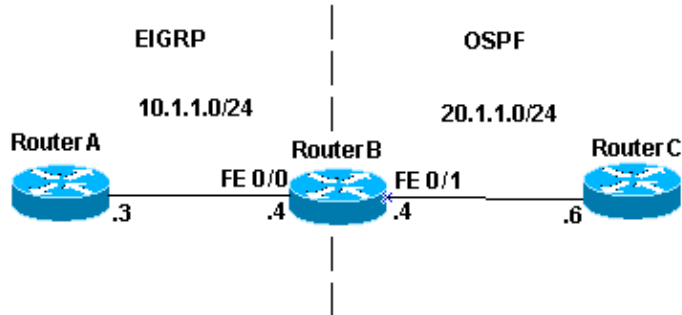
The following output shows an ISIS router redistributing static, RIP, IGRP, EIGRP, and OSPF routes.

```
router isis
network 49.1234.1111.1111.1111.00
redistribute static
redistribute rip metric 20
redistribute igrp 1 metric 20
redistribute eigrp 1 metric 20
redistribute ospf 1 metric 20
```

The ISIS metric needs to be between 1 and 63. There is no default-metric option in ISIS, so you should define a metric for each protocol, as shown in the example above.

Connected Routes

Redistributing directly connected networks into routing protocols is not a common practice and isn't shown in any of the examples above for this reason. However, it's important to note that it can be done, both directly and indirectly. To directly redistribute connected routes, use the **redistribute connected** sub-router configuration command. You should also define a metric in this case. You can also indirectly redistribute connected routes into routing protocols as shown in the following example.



Router B above has two Fast Ethernet interfaces. FastEthernet 0/0 is in network 10.1.1.0/24 and FastEthernet 0/1 is in network 20.1.1.0/24. Router B is running EIGRP with Router A, and OSPF with Router C. Router B is mutually redistributing between the EIGRP and OSPF processes. Below is the pertinent configuration information for Router B.

```
Router B
interface FastEthernet0/0
 ip address 10.1.1.4 255.255.255.0

interface FastEthernet0/1
 ip address 20.1.1.4 255.255.255.0

router eigrp 7
 redistribute ospf 7 metric 10000 100 255 1 1500
 network 10.1.1.0 0.0.0.255
 auto-summary
 no eigrp log-neighbor-changes
!
router ospf 7
 log-adjacency-changes
 redistribute eigrp 7 subnets
 network 20.1.1.0 0.0.0.255 area 0
```

If we look at the routing table for Router B we see the following:

```
routerB#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route
```

Gateway of last resort is not set

```
      20.0.0.0/24 is subnetted, 1 subnets
C       20.1.1.0 is directly connected, FastEthernet0/1
      10.0.0.0/24 is subnetted, 1 subnets
```

Redistributing Routing Protocols

```
C      10.1.1.0 is directly connected, FastEthernet0/0
```

From the configuration and the routing table above there are three things to notice:

- The networks in question are in Router B's routing table as directly connected networks.
- Network 10.1.1.0/24 is part of the EIGRP process and network 20.1.1.0/24 is part of the OSPF process.
- Router B is mutually redistributing between EIGRP and OSPF.

Now, let's take a look at the routing tables for Routers A and C.

```
routerA#show ip route
```

```
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
```

```
Gateway of last resort is not set
```

```
      10.0.0.0/24 is subnetted, 1 subnets
C      10.1.1.0 is directly connected, FastEthernet0
      20.0.0.0/24 is subnetted, 1 subnets
D EX   20.1.1.0 [170/2560025856] via 10.1.1.4, 00:07:26, FastEthernet0
```

```
routerC#show ip route
```

```
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route
```

```
Gateway of last resort is not set
```

```
      20.0.0.0/24 is subnetted, 1 subnets
C      20.1.1.0 is directly connected, FastEthernet1
O E2   10.1.1.0 [110/20] via 20.1.1.4, 00:07:32, FastEthernet1
```

We see that Router A has learned about network 20.1.1.0/24 via EIGRP and that Router C has learned about network 10.1.1.0/24 via OSPF. Although Router B is not redistributing connected networks, it does advertise the network 10.1.1.0/24, which is part of the EIGRP process redistributed into OSPF. Similarly, Router B advertises network 20.1.1.0/24, which is part of the OSPF process redistributed into EIGRP.

For more information about connected routes being redistributed into OSPF, see [Redistributing Connected Networks into OSPF](#).

Avoiding Problems Due to Redistribution

In the above section on administrative distance you saw how redistribution can potentially cause problems such as below optimal routing, routing loops, or slow convergence. Avoiding these types of problems is really quite simple: never announce the information originally received from routing process X back into routing


```

redistribute rip metric 1 1 1 1 1
distribute-list 1 in s1

router rip
network 178.1.10.0
redistribute igrp 7 metric 2

access-list 1 deny 192.168.1.0
access-list 1 permit any

```

R5:

```

router igrp 7
network 181.16.2.0

redistribute rip metric 1 1 1 1 1
distribute-list 1 in s1

router rip
network 178.1.10.0
redistribute igrp 7 metric 2

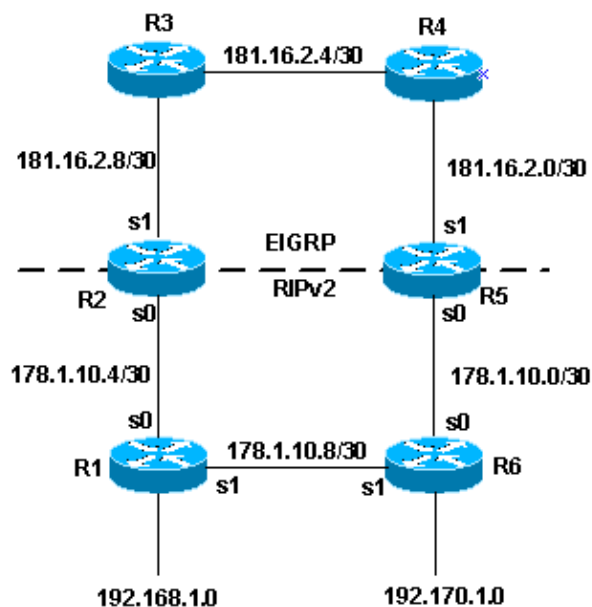
access-list 1 deny 192.168.1.0
access-list 1 permit any

```

The distribute lists added to the configurations, as shown above, filter any IGRP updates coming into the serial 1 interface of the routers. If the routes in the updates are permitted by access list 1, then the router will accept them in the update; otherwise it won't. In this example, the routers are being told that they should not learn network 192.168.1.0 through the IGRP updates they receive on their serial 1 interface; therefore, the only knowledge these routers will have for network 192.168.1.0 will be through RIP from R1.

Also keep in mind that in this case it is not necessary to use the same filter strategy for the RIP process because RIP has a higher administrative distance than IGRP. If routes that originate in the IGRP domain were fed back to R2 and R5 through RIP, then the IGRP routes would still take precedence.

Example 2



Using the topology as above, we can demonstrate another method (which is sometimes more preferable) for avoiding redistribution problems. This technique utilizes route-maps to set tags for various routes. Routing processes can then redistribute based on the tags. Note that redistribution based on tags will not work with RIP version 1 or IGRP.

One of the problems you can run into in the above topology is as follows:

R1 advertises network 192.168.1.0 to R2. R2 then redistributes to EIGRP. R5 learns the network via EIGRP and redistributes it to RIPv2. Depending on the metric R5 sets for the RIPv2 route, R6 may prefer the less desirable route through R5 instead of through R1 to reach the network. The following configuration helps to prevent this by setting tags and then redistributing based on the tags.

R2:

```
router eigrp 7
network 181.16.2.0
redistribute rip route-map rip_to_eigrp metric 1 1 1 1 1
!--- Redistributes RIP routes that are
!--- permitted by the route-map rip_to_eigrp

router rip
version 2
network 178.1.10.0
redistribute eigrp 7 route-map eigrp_to_rip metric 2
!--- Redistributes EIGRP routes and set the tags
!--- according to the eigrp_to_rip route-map

route-map rip_to_eigrp deny 10
match tag 88
!--- Route-map statement to deny any routes that have a tag of "88"
!--- from being redistributed into EIGRP
!--- Notice the routes tagged with "88" should be the EIGRP
!--- routes that are redistributed into RIPv2

route-map rip_to_eigrp permit 20
set tag 77
!--- Route-map statement to set the tag
!--- on RIPv2 routes redistributed into EIGRP to "77"

route-map eigrp_to_rip deny 10
match tag 77
!--- Route-map statement to deny any routes that have a
!--- tag of "77" from being redistributed into RIPv2
!--- Notice the routes tagged with "77" should be the RIPv2
!--- routes that are redistributed into EIGRP

route-map eigrp_to_rip permit 20
set tag 88
!--- Route-map statement to set the tag on RIPv2
!--- routes redistributed into EIGRP to "88"
```

R5:

```
router eigrp 7
network 181.16.2.0
redistribute rip route-map rip_to_eigrp metric 1 1 1 1 1
!--- Redistributes RIPv2 routes that are permitted
```

Redistributing Routing Protocols

```

!--- by the route-map rip_to_eigrp

router rip
version 2
network 178.1.10.0
redistribute eigrp 7 route-map eigrp_to_rip metric 2
!--- Redistributes EIGRP routes and sets the tags
!--- according to the eigrp_to_rip route-map

route-map rip_to_eigrp deny 10
match tag 88
!--- Route-map statement to deny any routes that have a tag
!--- of "88" from being redistributed into EIGRP
!--- Notice the routes tagged with "88" should be the EIGRP routes
!--- that are redistributed into RIPv2

route-map rip_to_eigrp permit 20
set tag 77
!--- Route-map statement to set the tag on rip routes
!--- redistributed into EIGRP to "77"

route-map eigrp_to_rip deny 10
match tag 77
!--- Route-map statement to deny any routes that have a tag
!--- of "77" from being redistributed into RIPv2
!--- Notice the routes tagged with "77" should be the RIPv2 routes
!--- that are redistributed into EIGRP

route-map eigrp_to_rip permit 20
set tag 88
!--- Route-map statement to set the tag on RIPv2 routes
!--- redistributed into EIGRP to "88"

```

With the above configuration configured, we can look at some specific routes in the routing table to see that the tags have been set. Below shows output from the **show ip route** command for specific routes on R3 and R1:

```

R3#show ip route 178.1.10.8
Routing entry for 178.1.10.8/30
  Known via "eigrp 7", distance 170, metric 2560025856
  Tag 77, type external
  Redistributing via eigrp 7
  Last update from 181.16.2.10 on Ethernet0/0, 00:14:30 ago
  Routing Descriptor Blocks:
  * 181.16.2.10, from 181.16.2.10, 00:14:30 ago, via Ethernet0/0
    Route metric is 2560025856, traffic share count is 1
    Total delay is 1010 microseconds, minimum bandwidth is 1 Kbit
    Reliability 1/255, minimum MTU 1 bytes
    Loading 1/255, Hops 1

```

```

R1#show ip route 181.16.2.4
Routing entry for 181.16.2.4/30
  Known via "rip", distance 120, metric 2
  Tag 88
  Redistributing via rip
  Last update from 178.1.10.5 on Ethernet1/0, 00:00:10 ago
  Routing Descriptor Blocks:
  * 178.1.10.5, from 178.1.10.5, 00:00:10 ago, via Ethernet1/0
    Route metric is 2, traffic share count is 1

```

Example 3

Redistribution can also take place among different process of the same routing protocol. The following configuration is an example of a redistribution policy used for redistributing two EIGRP process running on the same router or on multiple routers:

```
router eigrp 3
 redistribute eigrp 5 route-map to_eigrp_3
 default-metric 10000 100 255 1 1500
 !--- Redistributes EIGRP 5 into EIGRP 3, setting the tags
 !--- according to the route map "to_eigrp_3"

router eigrp 5
 redistribute eigrp 3 route-map to_eigrp_5
 default-metric 10000 100 255 1 1500
 !--- Redistributes EIGRP 3 into EIGRP 5
 !--- Routes with tag 51 will not be redistributed
 !--- due to route map "to_eigrp_5"

route-map to_eigrp_3 deny 10
 match tag 55
 !--- Route-map statement used to deny any routes that have a tag
 !--- of "55" from being redistributed into EIGRP 3
 !--- Notice the routes tagged with "55" should be the EIGRP 3 routes
 !--- that are redistributed into EIGRP 5

route-map to_eigrp_3 permit 20
 set tag 33
 !--- Route-map statement used to set the tag on routes
 !--- redistributed from EIGRP 5 to EIGRP 3 to "33"

route-map to_eigrp_5 deny 10
 match tag 33
 !--- Route-map statement used to deny any routes that have a tag
 !--- of "33" from being redistributed into EIGRP 5
 !--- Notice the routes tagged with "33" should be the EIGRP 5 routes
 !--- that are redistributed into EIGRP 3

route-map to_eigrp_5 permit 20
 set tag 55
 !--- Route-map statement used to set the tag on routes
 !--- redistributed from EIGRP 3 to EIGRP 5 to "55"
```

These are just a few examples of filtering strategies used for the intent of this document; however, there may be other valid strategies that can also be used. For more information see the section on Filtering Routing Information in Configuring IP Routing Protocol–Independent Features.

Tools Information

For additional resources, refer to Cisco TAC Tools for Routing Protocol Technologies.

Related Information

- **RIP and OSPF Redistribution**
- **Redistribution between Enhanced IGRP and RIP**
- **Redistribution Between EIGRP and IGRP in the Same Autonomous System**

- **Redistributing Between Classful and Classless Protocols: EIGRP or OSPF into RIP or IGRP**
 - **Redistributing in BGP**
 - **redistribute Command Reference**
 - **More Routing Protocol Technical Tips**
-

All contents are Copyright © 1992—2002 Cisco Systems Inc. All rights reserved. Important Notices and Privacy Statement.

Updated: Apr 23, 2002

Document ID: 8606
