

CHAPTER 7

Border Gateway Protocol 4 (BGP-4)

In this chapter:

- Background
- Getting BGP Running
- How BGP Works
- Load Balancing
- Route-Filtering
- Connecting to the Internet
- Choosing an ISP
- Troubleshooting BGP
- Summing Up

My first passport, issued by the Government of India in 1981, bore a curious stamp in bold red ink on one of its first pages: “NOT VALID FOR TRAVEL IN THE REPUBLIC OF SOUTH AFRICA.”

Toward the end of the 1990s, I was traveling through Europe with a U.S. passport. Upon landing at Schiphol Airport in Amsterdam, I was stopped by an officer and led to a small room off to the side. There, I was made to remove my left shoe and sock. The officer checked my sock carefully before letting me go. Once I had cleared Dutch customs, I freely roamed up and down that beautiful country, walking barefoot over grass and flowers, past windmills.

Nations have policies concerning who can pass through their borders. So, India did not permit its citizens to grace the Republic of South Africa during its reign of apartheid. And the Dutch bar entry into their country if your left sock is not fresh and clean.

What does all this have to do with internetworking, or BGP-4, for that matter? Each network is an autonomous system (AS) managed by a single technical entity and under one political administration. ASs are akin to nation states. Much as nation states apply their immigration policies at international airports, seaports, and land border points, the Internet is composed of ASs that use Border Gateway Protocol 4 (BGP-4) to implement inter-AS IP routing policies.

What is a routing policy? Consider the topology in Figure 7-1. TraderMary has two links to the Internet: to ISP-A and ISP-B. TraderMary may implement a policy that all traffic should exit and enter their network via ISP-A and that the link to ISP-B should

be used only for backup. ISP-Global connects to both ISP-US and ISP-Finland. ISP-US and ISP-Finland may enter into an arrangement with ISP-Global to carry transit traffic for them across the Atlantic. All three ISPs would need to reflect this arrangement in their routing policies.

The interior gateway protocols (IGPs) we discussed in earlier chapters—RIP, IGRP, EIGRP, and OSPF—relied on a single composite metric to choose the best route to a destination. BGP-4 implements routing policies based on a new paradigm—a set of *attributes* accompanying each route are used to pick the “shortest” path across multiple ASs, while also satisfying one or more routing policies.

RIP, IGRP, and OSPF are examples of IGPs. IGPs are designed for intra-AS routing. BGP-4 is an exterior gateway protocol (EGP), designed for inter-AS routing.

BGP-4 may be used to set up routing between any two ASs. However, the most interesting and complex use of BGP-4 is in the Internet: to connect client networks (such as TraderMary) to their ISPs and ISPs to other ISPs. This chapter will focus on the use of BGP-4 in connecting clients, such as TraderMary, to their ISPs.

BGP was first defined in RFC 1105 (1989) and was updated to BGP-2 in RFC 1163 (1990), to BGP-3 in RFC 1267 (1991), and then to BGP-4 in RFC 1771 (1995). BGP-4 is the first version that handles aggregation of prefixes along Classless Inter-Domain Routing (CIDR) lines, as described in the next section. BGP-4 may live longer than its forerunners because it is capable of supporting new attributes to keep up with an evolving Internet.

Background

An AS in the Internet must be identifiable via a unique, registered AS number and one or more unique, registered IP addresses. IP addresses in the Internet are not carved along the classful A, B, and C boundaries, but instead use the concepts of Classless Inter-Domain Routing (CIDR).

This section describes how an AS number and an IP block may be acquired and the concepts behind CIDR. I begin with a discussion of the various types of ASs.

AS Types

An AS is a network managed by a single technical entity and under one political administration. Figure 7-1 shows nine different ASs connected to each other in a small mesh that may be seen as a microcosm of the Internet. Architecturally, these ASs may be quite similar. Functionally, however, all ASs are not equal. TraderMary, BrotherX, and SisterY are clients of the ISPs to which they connect. These ASs do not carry *transit* traffic, implying that if you pick any IP packet from these networks, its source *or* destination IP address will be internal to the AS. These ASs are referred to as *stub* ASs.

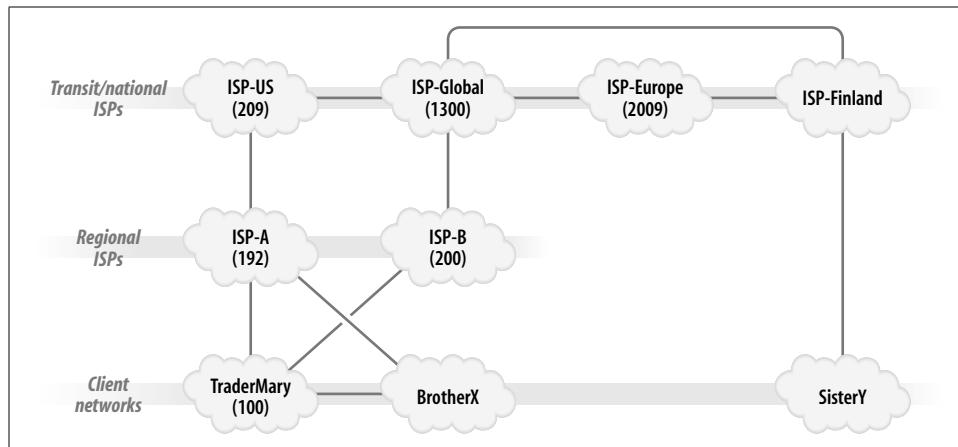


Figure 7-1. An attempt to define structure in the Internet

Local ISPs or regional ISPs provide transit service to stub ASs, implying that local/regional ISPs carry transit traffic for the stub AS to other networks. The local and regional ISPs, in turn, are clients of national ISPs and transit ISPs, which provide transit service over wider geographies.

The moment I describe these rules, I must admit that the rules are meant to be broken. For example, it is common for larger stub ASs to bypass local/regional ISPs and connect directly to national/transit ISPs.



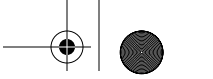
Stub ASs may be further classified to describe their connectivity to the Internet, as follows. BrotherX's network has a single connection to ISP-A. BrotherX's network may be described as *singly-homed*. TraderMary may be described as *multi-homed*, since it connects to ISP-A and ISP-B.

Multi-homing to the same ISP guards against the failure of a single link but not against failures in the ISP's network. *Multi-homing to different ISPs* also guards against failures in an ISP's network.

Gluing AS to AS... Physical Connectivity in the Internet

At a physical level, stub ASs typically connect to an ISP via a serial link. TraderMary may lease a T-3 line to ISP-A and a T-1 line to ISP-B; BrotherX may lease a 56-kbps circuit to ISP-A.

ISPs establish connections with each other at higher speeds. The NSF originally helped establish *network access points* (NAPs) that provided the infrastructure over which various ASs could exchange routes and traffic. The original NAPs were run by Sprint (Pennauken, NJ), PacBell (San Francisco), Ameritech (Chicago), and MFS (Washington, D.C. and San Jose). Now, there are dozens of "Internet Exchanges" where ISPs and other ASs may connect. Several exchange points maintain excellent



web sites with details on their services. For example, the Ameritech (Chicago) exchange (<http://www.aads.net/main.html>) uses ATM to provide high-speed connectivity. PAIX (<http://www.paix.net>) is located in Palo Alto, CA. The Commercial Internet Exchange (<http://www.cix.net>) is a not-for-profit organization that provides various interconnection options at different prices. In addition to the Exchanges, ISPs establish *private* peering arrangements in which two ASs set up a private network to exchange routes and traffic.

Internet Registries: IP Addresses and AS Numbers

Each device in the Internet must be identified by a unique IP address registered with one of the Internet Registries. ARIN (<http://www.arin.net>) registers IP addresses for the Americas, the Caribbean, and parts of Africa; RIPE NCC (<http://www.ripe.net>) takes care of the same tasks for Europe, the Middle East, and other parts of Africa; and APNIC (<http://www.apnic.net>) is responsible for Asia and the Pacific Region.

In the early to mid 1990s, it was recognized that the Internet was facing two critical issues. First, the available IP address space, especially that of Class B numbers, was rapidly depleting. The Class A address space was too big for most users and the Class C address space was too small. Second, just as more and more addresses were being allocated to users and organizations, the size of the Internet routing table was growing more rapidly than router processing power.

RFCs 1517, 1518, 1519, and 1520 proposed a solution to these twin problems: Classless Inter-Domain Routing (CIDR).

Classless Inter-Domain Routing (CIDR)

CIDR was revolutionary. To begin with, CIDR did away with fixed Class A, B, and C addresses. This takes a little retraining for us old-timers who grew up on classful addressing. We were taught that for a network of 1,000 hosts you would need a Class B address, Class A being too big and Class C being too small. However, since 1,000 hosts can be addressed with 10 bits, any network number with 10 bits in the host field and $32 - 10 = 22$ bits in the network field would suffice. In CIDR parlance, such a network is described as “/22,” implying that there are 22 bits in the network field. These 22 bits can be derived from any of the address ranges that were classically described as Class A, B, and C. Thus, $20.1.4.0/22$, $150.100.252.0/22$, and $192.168.68.0/22$ are all valid CIDR blocks with 22 bits in the network field. The network administrator may then subnet the 10 bits in the host field as appropriate, just as she would if assigned a classful IP address. By allocating addresses in blocks that match user requirements, CIDR reduces the rate at which the available IP address pool is depleting.

CIDR goes a step further to reduce the size of IP routing tables. The CIDR schema proposes that clients derive IP addresses from their connected ISP rather than directly

from an Internet Registry. In other words, ISPs derive address blocks from Internet Registries and carve them for their clients. This address-allocation schema is described as *topological* since clients derive IP addresses based on their physical connectivity. The advantage of this schema is that each ISP needs to advertise only one *aggregate* route for all its connected clients, rather than individual prefixes for each client.

All this deserves an example. Consider ISP-X. Let's say that ISP-X owns the IP address block 180.180.0.0/16. *Uncle-P* connects to ISP-X and requires 8 bits to address his hosts. In other words, the client requires a "/24". ISP-X will assign, say, 180.180.1.0/24 to *Uncle-P*. Then, the next day, *Uncle-Q* connects to ISP-X and also requires 8 bits to address his hosts. ISP-X will assign, say, 180.180.2.0/24 to *Uncle-Q*. ISP-X's own routing tables hold detailed routes for all subnets in the 180.180.0.0/16 block, but ISP-X advertises only one prefix—180.180.0.0/16—to all other ASs. In other words, ISP-X issues an aggregate 180.180.0.0/16 to other ISPs. This is illustrated in Figure 7-2.

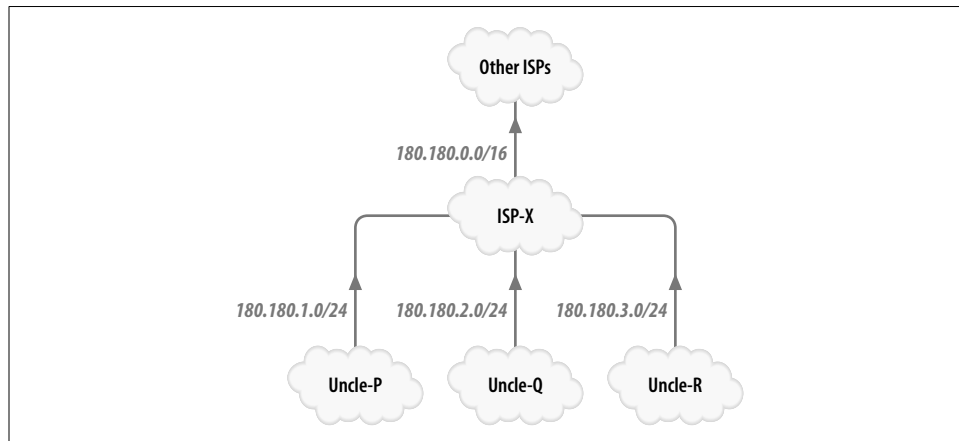


Figure 7-2. Route aggregation using CIDR

One problem with this method of address assignment is that when *Uncle-P* decides to use a different ISP—say, ISP-Y—*Uncle-P* has to return the 180.180.1.0 block to ISP-X and readdress all his devices from a new block of addresses procured from ISP-Y. This plan forces *Uncle-P* to renumber his IP addresses when he changes his ISP.

Multi-homing to different ISPs also creates problems with this schema. *Uncle-Q* has the address block 180.180.1.0/24 from ISP-X, but he also connects to ISP-Z. ISP-Z would have to carry BrotherX's specific route, 180.180.1.0/24. In other words, since ISP-Z advertises BrotherX's prefix, the routing tables in the attached ASs will see both the aggregate 180.180.0.0/16 from ISP-X and 180.180.1.0 from ISP-A. However, addresses may still be aggregated at a higher level in the network, based on address hierarchy.

Route aggregation using CIDR is hierarchical, since address blocks have been allocated by geography. Ideally, an ISP in Europe should see an aggregate for all CIDR-derived routes in Japan and another aggregate for all CIDR-derived routes in Australia.

Prior to IOS Release 12.0, a router assumed that all routing information it received was classful. This implies that the router assumed that all networks had IP addresses that were assigned along classful lines. The following command turns off classful behavior in IOS releases earlier than 12.0. Releases 12.0 and later assume classless behavior by default.

```
ip classless
```

Further, in classful behavior, the use of subnet zero creates confusion in the IP routing table. If 180.180.0.0 were to be subnetted with a 24-bit mask, the subnet zero would be 180.180.0.0/24. This subnet is easily confused with the entire address space 180.180.0.0/16 if there are no masks in routing updates. Hence, with classful routing protocols, it was common for the IOS to prevent the configuration of the subnet zero on any user interface. The following command in global configuration mode allows the configuration of subnet zero:

```
ip subnet-zero
```

Acquiring an IP Address

Every organization attached to the Internet must have a *unique* IP address in order for it to have an unambiguous path in the Internet. A stub AS may derive an IP address block from its ISP using the CIDR schema (as *Uncle-P* did from ISP-X) or it may apply for an IP address block directly from an Internet Registry.

CIDR-derived addresses reduce routing-table overhead in the Internet. Internet Registry-derived addresses have the advantage of being *portable*: if the stub AS moves to a different ISP, there is no need to renumber IP devices.

All devices in a network typically do not access the Internet directly. User workstations and servers employ a proxy device (such as a firewall) that has one interface on an internal network and another on the Internet. The proxy device originates a TCP session on behalf of the user. The proxy must have a registered IP address on its Internet interface. The internal addresses are not visible on the Internet and so do not need to be registered. The IP addresses we have seen thus far on TraderMary's network have come from the pool of private addresses reserved by the Internet Assigned Numbers Authority (IANA). This pool of private addresses is defined by the following ranges (see RFC 1918 for further details):

```
10.0.0.0 - 10.255.255.255 (10/8 prefix)
172.16.0.0 - 172.31.255.255 (172.16/12 prefix)
192.168.0.0 - 192.168.255.255 (192.168/16 prefix)
```

RFC 1918 sets these addresses aside for use by *any* organization for numbering its devices. These prefixes cannot be advertised into the (public) Internet.

In the example we'll look at later, TraderMary uses the registered Class C addresses 192.200.200.0/24 and 160.160.0.0/16 to connect to ISP-A.

Acquiring an AS Number

Each AS running BGP-4 is associated with an AS number. This AS number must be unique and unambiguous for BGP-4 to operate correctly.

Each AS in the Internet should be identified by an AS number that is registered with one of the Internet Registries. ARIN registers AS numbers for the Americas, the Caribbean, and parts of Africa; RIPE NCC takes care of the same tasks for Europe, the Middle East, and other parts of Africa; and APNIC does the same for Asia and the Pacific Region.

Every AS in the Internet must have a unique AS number. However, a stub AS that is singly-homed to an ISP may "borrow" its ISP's AS number or use one of the AS numbers reserved for private use by IANA. The range of private AS numbers is 64,512 through 65,535.

Getting BGP Running

Starting BGP on a router is similar to starting any other routing process, such as RIP or IGRP. The command to start BGP is:

```
router bgp AutonomousSystemNumber
```

where *AutonomousSystemNumber* is the AS number of the local router.

This is where the similarity with other routing protocols stops. When configured under BGP, the following network statement:

```
network IPAddress [mask A.B.C.D]
```

specifies the prefix to announce to BGP peers. Compare this with the configuration of IGP's, where the network-number statement has very different semantics: it specifies the attached networks on which to discover neighbors or peers.

Speaking of peers, there are no mechanisms in BGP-4 to automatically discover neighbors. BGP-4 requires that peers *must* be specified by IP address. The command to specify a peer is:

```
neighbor IPAddress remote-as AutonomousSystemNumber
```

where *IPAddress* specifies the peer with an AS number of *AutonomousSystemNumber*.

Let's look at TraderMary's configuration for its connection to ISP-A, as shown in Figure 7-3. Line 1 in the following code block starts BGP with a local AS number of 100. Line 3 specifies that the prefix 192.200.200.0/26 be announced to *TrdrMary-1*'s BGP peers. Line 4 specifies that the network number 30.0.0.0 be announced as well, with an 8-bit mask (the natural classful mask is used when a mask is not specified).

Line 5 specifies that all static routes should also be announced. (There are two static routes known to *TrdrMary-1*, as shown in lines 7 and 8). Line 6 specifies that *TrdrMary-1*'s BGP peer (ISP-A) has an IP address of 192.100.100.254 and an AS number of 192. This is the only neighbor statement, so in this example *TrdrMary-1* has only one peer: ISP-A.

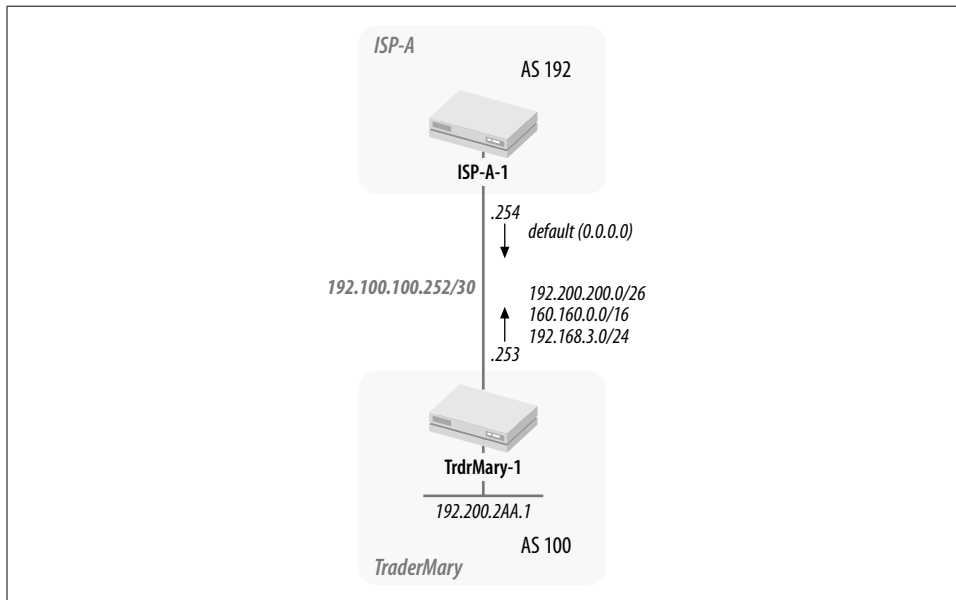


Figure 7-3. TraderMary's connection to ISP-A

Here's what the configuration looks like:

```

hostname TrdrMary-1
!
interface Loopback0
 ip address 192.168.1.10 255.255.255.255
!
interface Ethernet0
 description * External Network *
 ip address 192.200.200.1 255.255.255.192
!
interface Ethernet1
 ip address 172.16.1.3 255.255.255.0
!
interface Serial1
 description * to ISP-A *
 ip address 192.100.100.253 255.255.255.252
...
1 router bgp 100
2 no synchronization
3 network 192.200.200.0 mask 255.255.255.192
    
```



```

4 network 30.0.0.0
5 redistribute static
6 neighbor 192.100.100.254 remote-as 192
  !
  ip classless
7 ip route 160.160.1.0 255.255.255.0 Ethernet1
8 ip route 192.168.3.0 255.255.255.0 Ethernet1

```

The configuration on *ISP-A-1* is very similar. Line 9 starts BGP on *ISP-A-1* with the AS number 192. Line 10 specifies the default route being announced to TraderMary. Line 11 specifies that *ISP-A-1*'s BGP peer (*TrdrMary-1*) has an IP address of 192.100.100.253 and an AS number of 100:

```

hostname ISP-A-1
!
interface Loopback9
 ip address 98.2.0.1 255.255.0.0
!
interface Serial0
 ip address 192.100.100.254 255.255.255.252
!
9 router bgp 192
10 network 0.0.0.0
11 neighbor 192.100.100.253 remote-as 100
  !
  no ip classless
12 ip route 0.0.0.0 0.0.0.0 Null0

```

The first question to ask after both peers have been configured is whether the peers *see* each other. The following command checks the status of the neighbor relationship between *TrdrMary-1* and *ISP-A-1*:

```

TrdrMary-1#sh ip bgp neighbor 192.100.100.254
13 BGP neighbor is 192.100.100.254, remote AS 192, external link
  Index 1, Offset 0, Mask 0x2
14 BGP version 4, remote router ID 98.2.0.1
15 BGP state = Established, table version = 5, up for 00:00:42
  ...

```

Line 15 indicates that the BGP state is *Established*, which implies that the peers see each other.

The configuration in lines 3, 4, and 5 suggests that *TrdrMary-1* intends to announce the following prefixes to *ISP-A-1*:

```

192.200.200.0/26
30.0.0.0/8
160.160.0.0/16
192.168.3.0/24

```

and the configuration in line 10 suggests that *ISP-A-1* intends to announce the following prefix to *TrdrMary-1*:

```

0.0.0.0/0

```

Let's check the routing tables for these prefixes:

```
TrdrMary-1#sh 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
```

```
Gateway of last resort is 192.100.100.254 to network 0.0.0.0
```

```
192.200.200.0/26 is subnetted, 1 subnets
C    192.200.200.0 is directly connected, Ethernet0
172.16.0.0/24 is subnetted, 1 subnets
C    172.16.1.0 is directly connected, Ethernet1
192.168.1.0/32 is subnetted, 1 subnets
C    192.168.1.10 is directly connected, Loopback0
S    192.168.3.0/24 is directly connected, Null0
192.100.100.0/30 is subnetted, 1 subnets
C    192.100.100.252 is directly connected, Serial1
160.160.0.0/24 is subnetted, 1 subnets
S    160.160.1.0 is directly connected, Ethernet1
B*  0.0.0.0/0 [20/0] via 192.100.100.254, 00:00:46
```

TrdrMary-1 does receive the default route (as expected) and modifies its gateway of last resort to 192.100.100.254 (the IP address of *ISP-A-1*).

ISP-A-1 receives three routes from *TrdrMary-1*, as shown in lines 16, 17, and 18:



```
ISP-A-1#sh ip route
...
Gateway of last resort is 0.0.0.0 to network 0.0.0.0

192.200.200.0/26 is subnetted, 1 subnets
16 B    192.200.200.0 [20/0] via 192.100.100.253, 00:00:23
17 B    160.160.0.0/16 [20/0] via 192.100.100.253, 00:00:23
S*  0.0.0.0/0 is directly connected, Null0
18 B    192.168.3.0/24 [20/0] via 192.100.100.253, 00:20:51
...
```

However, if you look carefully at *ISP-A-1*'s routing table, you'll see that 30.0.0.0/8 (which *TrdrMary-1* attempted to announce on line 4) is missing. Why is 30.0.0.0/8 not in *ISP-A-1*'s routing table? Think on this. We will get back to this question in the next section.

How BGP Works

BGP's underlying algorithm is the simple DV protocol—when a BGP speaker hears a prefix via multiple paths, it selects the “best” path for insertion in the routing table and announces this “best” path to other peers.





We are already familiar with the DV protocol via RIP and IGRP. However, unlike RIP and IGRP, BGP's purpose is inter-AS routing, which is a different beast from intra-AS routing. The architects of BGP created several new structures to support inter-AS routing. This section gives an overview of these new structures.

Let's start at the beginning. A unique, registered AS number is required for the BGP process to connect to the Internet. Then, unlike IGPs, BGP does not contain any mechanism for automatic neighbor discovery. The network administrator must manually define BGP neighbors. This is appropriate given that the neighbor may be in another AS.

Protocols such as RIP and OSPF are generous in exchanging updates; the network statement permits all known subnets to be announced in updates. BGP operates under a different paradigm—updates should be tightly controlled. Cisco's implementation of BGP gives several methods to control not only the prefixes that are announced but also the associated attributes.

As the Internet routing protocol, BGP must support a very large routing table: the current size of the Internet routing table is roughly 70,000 prefixes. Given this size, periodic refreshing (such as every 30 s in a RIP network) of this table (with the associated attributes) would be very costly. Hence, the BGP protocol specification calls for the prefix table to be exchanged only once, when BGP neighbors first see each other. Thereafter, BGP updates only announce new prefixes or withdraw previously announced prefixes.



This incremental or *quiet* approach to announcing prefixes reduces the routing protocol overhead but creates a new twist. Suppose that a BGP router X lost a link to a neighbor Y. All paths known via Y would be deleted. Now, let's suppose that router X had a second-best route via another neighbor, Z. Since there are no periodic updates in BGP, X would never discover the new path via Z. BGP gets around this problem by storing all prefixes learned via all neighbors in a table called the *BGP table*. The second-best route can now be learned from the BGP table. The BGP table can be quite huge and can require a considerable chunk of router memory, especially when there are multiple neighbors.

Small, homogenous networks can get away with a single metric to describe the best path to a destination—for example, RIP uses hop count to choose the “shortest” path. Given the complexity of inter-AS routing, no single metric can describe the best path across various ASs running multiple IGPs on heterogeneous media. BGP defines a rich set of attributes that describe each path to a destination. These attributes describe the path in various ways, allowing network administrators to implement various routing policies. As an example, the AS-PATH attribute is a list of AS numbers that describe the path to the destination. Other attributes describe the origin of the prefix, the IP address of the border router that should be used as the next hop,

the community to which the prefix belongs, a preference indication for the advertised route, etc.

The BGP route-selection algorithm uses the attribute list to select a single best path to each known destination. Understanding this route-selection algorithm is critical to manipulating BGP attributes to set routing policies.

The *route map* commands in Cisco IOS may be used to manipulate BGP attributes. The use of route maps is described later in this chapter.

Lastly, we must talk a little more about neighbor relationships. Neighbors come in two varieties: external and internal. If the neighbors are in different ASs, the BGP protocol between the neighbors is described as *External-BGP* (E-BGP). If the neighbors are in the same AS, the neighbor relationship is described as *Internal-BGP* (I-BGP).

Why do we need I-BGP? After all, isn't BGP's purpose inter-AS routing? Yes. Consider an AS with multiple E-BGP routers. In order for the AS to have a consistent routing policy, all BGP speakers in the AS must have identical BGP tables. I-BGP is used to propagate BGP tables through the AS to maintain a consistent routing policy.

Starting BGP

The command to start the BGP process on a Cisco router is:

```
router bgp AutonomousSystemNumber
```

where *AutonomousSystemNumber* is the local AS number. The AS number may be acquired as described earlier in the section "Acquiring an AS Number."

Neighbor Relationship

IGPs such as RIP broadcast or multicast updates, forming neighbor relationships with all directly connected routers. In contrast, neighbor relationships in BGP-4 are one-to-one between gateway and gateway.

After starting the BGP process, each gateway must specify its neighbor, or peer. The command to specify the peer is:

```
neighbor ip_address remote-as ASNumber
```

A gateway may have several neighbors, so multiple instances of the *neighbor* command may be listed under the BGP process. In the following example, TraderMary has added ISP-B (line 20) as a neighbor. The new topology is as shown in Figure 7-4.

```
hostname TraderMary-1
!
interface Loopback0
 ip address 192.168.1.10 255.255.255.255
!
interface Ethernet0
```

```
ip address 192.200.200.1 255.255.255.192
!  
interface Ethernet1  
ip address 172.16.1.3 255.255.255.0  
!  
interface Serial0  
description * to ISP-B *  
ip address 200.1.1.253 255.255.255.252  
!  
interface Serial1  
description * to ISP-A *  
ip address 192.100.100.253 255.255.255.252  
!  
router bgp 100  
no synchronization  
network 192.200.200.0 mask 255.255.255.192  
network 30.0.0.0  
redistribute static  
19 neighbor 192.100.100.254 remote-as 192  
20 neighbor 200.1.1.254 remote-as 200  
!  
ip classless  
ip route 160.160.1.0 255.255.255.0 Ethernet1  
ip route 192.168.3.0 255.255.255.0 Ethernet1
```

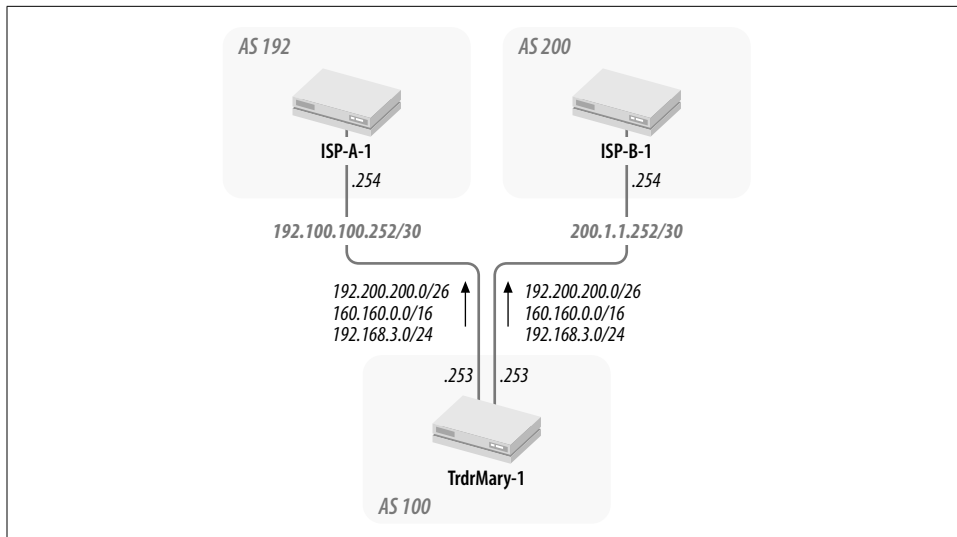


Figure 7-4. TraderMary's neighbors: ISP-A and ISP-B

ISP-A is configured with TraderMary as a neighbor:

```
hostname ISP-A-1  
...  
interface Serial0  
description * to TraderMary *
```

```

    ip address 192.100.100.254 255.255.255.252
    !
  router bgp 192
    neighbor 192.100.100.253 remote-as 100
    neighbor 192.100.100.253 default-originate
    !
  no ip classless
  ip route 0.0.0.0 0.0.0.0 Null0

```

And ISP-B is also configured with TraderMary as a neighbor:

```

hostname ISP-B-1
...
interface Serial1
  description * to TraderMary *
  ip address 200.1.1.254 255.255.255.252
  !
router bgp 200
  network 0.0.0.0
  neighbor 200.1.1.253 remote-as 100
  !
no ip classless
ip route 0.0.0.0 0.0.0.0 Null0

```

Since ISP-A and ISP-B have different AS numbers than TraderMary, they are E-BGP peers; when the BGP neighbors are in the same AS, they are I-BGP peers. The neighbor-building process is the same for E-BGP and I-BGP.

E-BGP peers have a restriction—they must be directly connected. This is usually not an issue because E-BGP neighbors are often on opposite ends of a serial link. (We will explore an exception to this restriction on E-BGP peers in the section “Load Balancing.”) When E-BGP peers issue an update, the NEXT-HOP IP address is modified to the IP address of the originating router’s outgoing interface.

I-BGP peers need not be directly connected. Further, I-BGP peers do not modify the NEXT-HOP IP address. When running I-BGP, it is useful to configure loopback addresses on the peering routers and use these addresses to source the I-BGP session. Since loopback interfaces are always up, the I-BGP session will stay up as long as there is any path between the I-BGP peers that can be discovered via an IGP.

Since all BGP exchanges are between a pair of BGP speakers, TCP port 179 can be used for reliability. (IGPs do not use TCP for reliability. Can you think of the reason for this?) Bear in mind that in order to establish a TCP session for the purpose of exchanging BGP messages, the BGP neighbors must be able to route IP datagrams to each other. In other words, the BGP neighbors must be able to reach each other via an IGP.

The following command specifies the source of a BGP TCP session, where *interface* may be the loopback interface:

```
neighbor ip_address update-source interface
```

In the absence of this command, the source interface is the closest interface to the neighbor, as known via an IGP. We will see a use for this command in the section “Load Balancing.”

The command `show ip bgp neighbor` shows the state of the neighbor relationship between BGP peers. Let’s take the BGP peers `TrdrMary-1` and `ISP-A-1` as an example:

```
TrdrMary-1#sh ip bgp neighbor 192.100.100.254
21 BGP neighbor is 192.100.100.254, remote AS 192, external link
    Index 1, Offset 0, Mask 0x2
22 BGP version 4, remote router ID 98.2.0.1
23 BGP state = Established, table version = 5, up for 00:00:42
24 Last read 00:00:42, hold time is 180, keepalive interval is 60 seconds
    Minimum time between advertisement runs is 30 seconds
    Received 54821 messages, 0 notifications, 0 in queue
    Sent 54826 messages, 0 notifications, 0 in queue
    Connections established 12; dropped 11
    Last reset 00:01:05, due to : User reset request
25 No. of prefix received 1
    Connection state is ESTAB, I/O status: 1, unread input bytes: 0
26 Local host: 192.100.100.253, Local port: 11070
27 Foreign host: 192.100.100.254, Foreign port: 179
...

```

Line 21 indicates the AS number of the remote BGP peer (ISP-A) and also shows that this is an *external link* (the local AS number is different from the remote AS number). If the BGP peers had the same AS numbers, line 21 would show *internal link*.

The BGP router ID (line 22) is the highest IP address configured on the router. However, if loopback interfaces are configured on the router, the router ID is the highest loopback address configured on the router.

Line 23 shows key information about the neighbor relationship. “BGP state = Established” indicates that the peers have successfully established a TCP connection. If the BGP peers had been unsuccessful in setting up a BGP session, line 23 would show a BGP state of *Idle*, *Active*, or *Connect*.

Initially, BGP peers exchange full BGP routing tables. After that exchange, only incremental updates are sent. A version number (line 23) tracks updates to the BGP routing table and can be used for troubleshooting. A rapidly increasing version number indicates that a route may be flapping.

After establishing a BGP session, the routers exchange *keepalives* every 60 seconds (line 24). The default hold-time is 180 seconds, following the standard rule in protocol design that if you miss them three times, you strike them off your list. Thus, if no keepalives are heard from a neighbor for 180 seconds, all routes learned from that neighbor are erased and the session is closed.

Line 25 shows the number of IP prefixes that have been received from the remote peer.

Lines 26 and 27 show the IP addresses of the BGP peers. Note that TCP port 179 was used on the foreign host; the local host used an ephemeral port number of 11070.

BGP Message Types

There are four message types in BGP. The *open message* allows BGP peers to identify their capabilities to each other, the *update message* is used to advertise/withdraw prefixes, the *notification message* is used to send errors/close the session, and the *keep-alive message* serves to keep the BGP session up. These four message types are described in further detail in the following sections.

Open

The purpose of the open message is for BGP peers to identify their capabilities to each other. This is the first message to be sent after BGP peers have established a TCP session.

Each open message specifies the following parameters defining the capabilities of the sender of the message:

BGP version number

Almost all implementations now use Version 4 (since it is the only version to support CIDR).

AS number

If the AS number sent does not match the AS number configured in the neighbor statement of the peer receiving the open message, the recipient sends a notification message indicating an error condition.

Hold timer

The duration of inactivity that will cause the sender of the open message to tear down the session. The hold timer is reset every time a keepalive or update message is received.

BGP identifier

This is the highest loopback address configured on the router and serves to uniquely identify the sender of the open message.

Optional parameters length

The length of the *optional parameters* field.

BGP peers may authenticate each other using the MD-5 algorithm, whose “message digest” may be placed in the open message as an optional parameter. A new optional parameter called *capability* permits BGP peers to evaluate each other’s capabilities for the support of new network-layer protocols such as IP multicast and IP Version 6. This new parameter—*capability*—is backward compatible, allowing a peer that does not support the parameter to maintain a session with a peer that does support the parameter.

Update

The update message is at the heart of BGP. Update messages are used to announce one or more prefixes to a BGP peer. The sender of the prefix must have a route to the prefix advertised, following the next-hop routing paradigm.

Sooner or later a network failure or change will cause the sender of the prefix to lose its route to the prefix it advertised. Hence, the update message must also include the ability to withdraw previously advertised prefixes. The update message specifies the following parameters:

Withdrawn routes length

The length of the *withdrawn routes* field.

Withdrawn routes

A list of IP prefixes that the sender had announced but now wishes to withdraw. This could be a result of a change in the network topology or configuration.

Total path attributes length

The length of the *attributes length* field.

Path attributes

A list of BGP attributes that apply to the prefixes described in the *network layer reachability information* field.

Network layer reachability information (NLRI)

A list of prefixes that the sender is advertising to its peer. Note that the path attributes listed earlier apply to all prefixes in the NLRI field.

Notification

Notification messages are used to indicate an error condition such as the expiry of the hold timer, the receipt of an unrecognized attribute type, an invalid AS number, etc. The underlying TCP session is closed after a notification message is sent.

An *error code* field in the notification message identifies the type of error.

Keepalive

The default interval between keepalive messages is 60 seconds (on Cisco routers). As per the specification, the hold timer is reset upon receipt of a keepalive or an update message.

Originating Routes

Now that we know how to start BGP and establish BGP neighbor relationships, we are ready to advertise prefixes between neighbors. There are three methods of transferring routes into the BGP table. Two of these methods were used by *TrdrMary-1* in the example in “Getting BGP Running.”

```

hostname TrdrMary-1
...
router bgp 100
no synchronization
28 network 192.200.200.0 mask 255.255.255.192
29 network 30.0.0.0
30 redistribute static
neighbor 192.100.100.254 remote-as 192

```

In lines 28 and 29, the network statement was used to insert routes into the BGP table. The syntax of the network statement is as follows:

```
network IPAddress [mask mask]
```

In line 28, 192.200.200.0 was advertised with a 26-bit mask, as specified by the network statement. When a mask is not specified, the natural classful mask is used in the BGP update. Hence, we would expect to see the prefix 30.0.0.0/8 in *ISP-A-1*'s routing table. However, as we saw earlier, there is no entry for 30.0.0.0/8 in *ISP-A-1*'s routing table. Did you think about the reason for this? You may want to take a moment before reading on...

TrdrMary-1 did not advertise 30.0.0.0/8 to *ISP-A-1* because *TrdrMary-1* has no IGP route for 30.0.0.0/8. This could be verified by checking *TrdrMary-1*'s BGP table: there is no entry for 30.0.0.0/8 in that table. A router should not advertise a route for which it does not have a path. Hence, the network-number statement alone is not enough to advertise a prefix; the router must also have a route for the prefix via an IGP. When the network statement is used to advertise a prefix, the ORIGIN attribute for the route is set to IGP.

Line 30 illustrates another mechanism for inserting prefixes into BGP updates: the *redistribute static* command.

TrdrMary-1 has two static routes, defined as follows:

```
ip route 160.160.1.0 255.255.255.0 Ethernet1
ip route 192.168.3.0 255.255.255.0 Ethernet1
```

Both routes are carried into *TrdrMary-1*'s BGP table, and henceforth to *ISP-A-1*, but not quite as *TraderMary* may have wanted. The static route for 160.160.1.0/24 gets copied as 160.160.0.0/16! This is because the default behavior is to use natural classful network numbers. To carry network numbers exactly as specified in the static route entries, use the BGP subcommand:

```
no auto-summary
```

When the *redistribute static* statement is used to advertise a prefix, BGP sets the ORIGIN attribute to *Incomplete*.

The last method for carrying routes into BGP is to redistribute an IGP into BGP. So, for example, the following command redistributes all routes known to the OSPF process into BGP:

```
redistribute ospf 10
```

When redistributing a dynamic protocol into BGP, there is the risk of not knowing what information is getting injected into BGP. Further, a flap in the IGP table will send ripples through all BGP tables of all routers that receive the prefixes that changed. This can be a big ripple when dealing with the Internet. In fact, ISPs will penalize routes that are repeatedly flapping, by not advertising the routes to other peers. (This is referred to as *BGP dampening*). Needless to say, redistributing a dynamic protocol into BGP is not the preferred method of transferring routes into BGP.

When an IGP is redistributed into BGP, the ORIGIN attribute is set to “?”.

In addition to these three methods of inserting prefixes into update packets, a BGP peer will also advertise any prefixes it hears from other BGP peers. This behavior is typical of DV routing protocols: pass the rumor along.

E-BGP Versus I-BGP

We have already seen a use of E-BGP in TraderMary’s network. TraderMary’s network has an E-BGP peer in ISP-A. The single default route received from ISP-A can be redistributed into TraderMary’s IGP, so there is no need for I-BGP in TraderMary’s network.

ISPs typically have multiple peers in other ASs. Each external router in the ISP receives routing information from all its E-BGP peers. For the AS to have a consistent routing policy, all external routers in the ISP’s network must share their BGP tables with each other. I-BGP peering between these routers allows them to share their BGP tables.

E-BGP peers modify the content of the routing information that is exchanged. As an example, E-BGP peers modify the NEXT-HOP attribute to point to their own IP addresses. I-BGP peers do not modify the content of the routing information that is exchanged. This ensures that all I-BGP speakers in the AS have a consistent BGP table and hence a consistent routing policy. The architects of the AS can then set up policies (such as “border routers *P* and *Q* will serve as exit points for destinations *X*, *Y*, and *Z*”) and propagate these policies throughout the AS via I-BGP.

When a BGP speaker receives an update from an E-BGP peer, it redistributes that update to all internal and external BGP peers. However, when a BGP speaker receives an update from an I-BGP peer, it redistributes that update to external, but not internal, peers. In Figure 7-5, *R* receives an update from an E-BGP neighbor. This update is redistributed to *K*, *S*, *T*, and *U*. But when *S* receives the update from *R*, it forwards it only to *L*, not to *U* or *T*.

In other words, when a BGP speaker receives an update from an I-BGP peer, it does not redistribute that update to other I-BGP peers. This restriction prevents routing updates from looping between I-BGP peers within the AS.

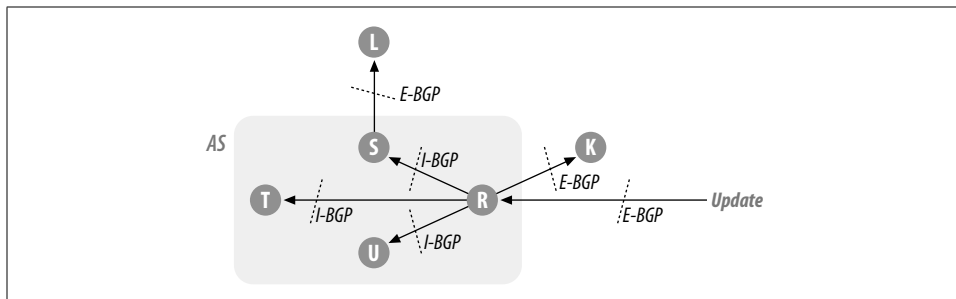


Figure 7-5. E-BGP versus I-BGP

Hence, in Figure 7-5, BGP speaker *R* must establish I-BGP relationships with *S*, *T*, and *U*. For the same reason, *S*, *T*, and *U* must establish I-BGP relationships with each other. In other words, a full I-BGP mesh is necessary for the AS to maintain a consistent BGP table.

A full I-BGP mesh ($n \times (n - 1) / 2$ sessions, where n is the number of I-BGP peers) is prohibitive even for mid-sized ASs. There are two common methods to get around implementing a full I-BGP mesh: *route reflectors* (RRs) and *confederations*. Both methods are based on the divide-and-conquer paradigm that also finds use in several computer algorithms.

When using RRs, the backbone is divided into clusters. Each cluster has an RR and several clients. The RRs must maintain a full I-BGP mesh, but a client in a cluster needs to maintain a session only with its RR. The job of the RRs is to forward updates to clients.

Confederations divide the backbone into sub-ASs. Each sub-AS must have a full I-BGP mesh. E-BGP is run between the confederations.

Synchronization

Let’s say that the IGP in use by ISP-A is slow compared to I-BGP (see Figure 7-7 later in this chapter). So, when *TrdrMary-1* advertises 192.200.200.0/24 to *ISP-A-1*, *ISP-A-3* may get possession of this route via I-BGP before *ISP-A-4* does via IGP. *ISP-A-3* modifies the NEXT-HOP attribute for 192.200.200.0/24 to its own IP address and begins to advertise the prefix to *ISP-US*. *ISP-US* begins forwarding traffic for 192.200.200.0/24 to *ISP-A-3*. *ISP-A-3*’s NEXT-HOP for 192.200.200.0/24 is 192.100.100.253, so it forwards the traffic to *ISP-A-4*. However, *ISP-A-4* promptly drops the traffic, since it has not yet learned this prefix via IGP. In other words, a “black hole” has occurred in the network due to the lack of synchronization between BGP and IGP. This black hole lasts until *ISP-A-4* learns the route to 192.200.200.0/24.

The moral of this story is that *ISP-A-3* should not advertise 192.200.200.0/24 to *ISP-US* until all the routers in the AS learn the prefix via IGP. In routing parlance, this is described as the *synchronization* of BGP with IGP.

Synchronization is on by default on Cisco routers, implying that a BGP speaker will not advertise a prefix to an E-BGP neighbor until it also learns that prefix via IGP. It is safe to turn off synchronization when the AS is not carrying any transit traffic. Nontransit ASs such as TraderMary should turn off synchronization to avoid the overhead of BGP/IGP interaction. The command to turn off synchronization is:

```
router bgp
no synchronization
```

The BGP Table

IGPs, such as RIP, maintain only the best route to any given destination. If the path to a destination becomes unavailable, RIP must wait for another update with a path to the same destination. However, BGP is a quiet protocol (a prefix is announced only once unless there is a change). If the best path is lost, how can BGP discover the second-best path?

The BGP process maintains a table that contains all known prefixes via all paths. The following output shows the BGP table for *TrdrMary-1* in the configuration described in Figure 7-4:

```
TrdrMary-1#sh ip bgp
BGP table version is 4, local router ID is 192.168.1.10
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop          Metric LocPrf Weight Path
31 * 0.0.0.0          200.1.1.254        0           0 200 i
32 *>                192.100.100.254    0           0 192 i
   *> 160.160.0.0     0.0.0.0            0           32768 ?
   *> 192.168.3.0      0.0.0.0            0           32768 ?
   *> 192.200.200.0   0.0.0.0            0           32768 i
```

The following is a column-by-column description of the entries in the BGP table.

“>” indicates the best route, which will be installed in the routing table. For example, network 0.0.0.0 is known via 200.1.1.254 (line 31) and 192.100.100.254 (line 32). The preferred path is via 192.100.100.254, as indicated by the “>” in line 32. Note that this table records the path via 200.1.1.254, even though that is not the preferred path.

Network describes the prefix in question. The BGP table lists locally generated prefixes as well as prefixes learned from other peers.

Next Hop describes the IP address to forward packets to for the prefix in the *Network* field. The next hop for prefixes learned from E-BGP peers is the IP address of the external router. The next hop for locally generated entries is 0.0.0.0.

Metric is the MED (Multi-Exit Discriminator) attribute associated with the prefix. The default value of this attribute is 0.

LocPrf is the value of the LOCAL-PREF attribute attached to the prefix. A higher LOCAL-PREF value indicates a more desirable route. The default value of the attribute is “?”.

Weight is a Cisco proprietary attribute. The default values of the Weight attribute are 32,768 for locally originated prefixes and 0 for prefixes advertised to a neighbor.

Path is the AS-PATH attribute and is represented in the table just before the Origin code. Thus, 0.0.0.0 is known via AS path 192 (which is the AS number of ISP-A) and AS path 200 (which is the AS number of ISP-B). The AS path is empty for locally generated prefixes such as 192.168.3.0.

The last column in the BGP table describes the ORIGIN attribute. The Origin codes of “i”, “e”, and “?” refer to the ORIGIN attribute. 0.0.0.0 and 192.200.200.0/26 were inserted into BGP with network statements, so the Origin code is “i”. 160.160.0.0 and 192.168.3.0 were inserted into BGP with the *redistribute static* command, so the Origin code is “?”.

Attributes

Akin to the description in my passport of my age, place of birth, and the border points I have crossed (there are stamps saying “Heathrow”, “Amsterdam”, “New Delhi”, and “New York”), which helps immigration officers decide whether to check my left or right shoe or to send me back to New York, each prefix in a BGP update message is accompanied by a set of *attributes*. Just as the information in a passport allows an immigration officer to implement her nation’s immigration policies, BGP’s attributes allow ASs to implement their own routing policies.

The number of attributes in a BGP update is variable, because some attributes are *mandatory* whereas others are *discretionary*. Every route update must be accompanied by all mandatory attributes, while discretionary attributes may or may not be sent in the update.

All BGP attributes fall into one of another two categories: *well-known* or *optional*. A well-known attribute must be recognized by all BGP implementations. An optional attribute need not be supported by all BGP implementations.

Lastly, all BGP attributes fall into one of another two categories: *transitive* or *non-transitive*. A nontransitive attribute is of significance only to the AS that receives the update—the attribute is not advertised to other ASs. A transitive attribute is of global significance and is forwarded in updates to other ASs.

It is common to use *route maps* on Cisco routers to control or modify routing information entering or leaving a routing process. Route maps have an intimate hook into BGP’s attributes. Each route map is a set of numbered clauses. Clauses are applied in order of their sequence number. Each clause contains a condition against which a route update is *matched*. If a route update matches the specified condition, the *set* option is used to specify an action.

Let's look at the syntax of a route map:

```
route-map map-tag [[permit | deny] | [sequence-number]]
match clauses
set actions
```

where *map-tag* is the name of the route map; the *permit/deny* keywords specify whether to accept or reject the prefixes that match the match clause; the *match clauses* can check for AS path, BGP community list, IP address, etc.; and the *set* command can modify one of several attributes associated with the route.

A route map may be applied on a peer using the *neighbor* command:

```
neighbor ip-address route-map route-map-name {in | out}
```

In the following example, route map *example1* (clause 10) sets the MED attribute to 200 for prefixes matching access list 1. The second clause (20) permits the advertisement of all other IP addresses with a MED attribute of 150:

```
route-map example1 permit 10
match ip address 1
set metric 200
!
route-map example1 permit 20
match ip address 2
set metric 150
!
access-list 1 permit 11.0.0.0 0.255.255.255
access-list 2 permit 0.0.0.0 255.255.255.255
```

This route map may be applied to neighbor 1.1.1.1 for all outgoing updates, as shown here:

```
router bgp 100
...
neighbor 1.1.1.1 route-map example1 out
```

Route map *example2* (clause 10) sets the MED to 200, the ORIGIN to IGP, and the Weight to 1,000 for prefixes matching access list 3. The second clause (20) permits the advertisement of all other IP addresses with a MED attribute of 150, ORIGIN of IGP, and Weight of 2,000.

```
route-map example2 permit 10
match ip address 3
set metric 200
set origin igp
set weight 1000
!
route-map example2 permit 20
match ip address 4
set metric 150
set origin igp
set weight 2000
!
access-list 3 permit 11.0.0.0 0.255.255.255
access-list 4 permit 0.0.0.0 255.255.255.255
```

This route map may be applied to all updates learned from RIP:

```
router bgp 100
...
redistribute rip route-map example2
```

The following sections review each BGP attribute with regard to its use in implementing routing policies.

ORIGIN (type code 1)

As we saw earlier, there are three methods for injecting a prefix into a BGP update message: the *network* statement, the *redistribute static* command, and the *redistribute dynamic routing protocol* command. The ORIGIN attribute describes which of these methods was used. The length of the attribute is 1 octet and is coded as follows:

0—IGP

The prefix is interior to the originating AS. This value is set when the *network* command is used to inject routes into BGP.

1—EGP

The prefix was learned via an EGP.

2—Incomplete

This most often means that the prefix was learned via static routes.

The Origin code is represented in the BGP table (*show ip bgp*) with the letters “i” for IGP, “e” for EGP, and “?” for Incomplete. Although it is well known and mandatory, the Origin code is not terribly useful in making routing decisions in today’s Internet.

AS-PATH (type code 2)

A prefix may travel from AS to AS in update messages. In the network in Figure 7-1, ISP-A receives the prefix 192.200.200.0 from AS 100. ISP-US sees 192.200.200.0 through ASs 192 and 100. ISP-Global sees the prefix through ASs 209, 192, and 100.

Each prefix in an update message is associated with the AS-PATH attribute. The AS-PATH attribute is the list of ASs that describes the path to the prefix.

BGP uses a rather straightforward algorithm to construct the AS-PATH attribute. The attribute is the *empty* list in the AS that originates the prefix. When sending the prefix to an E-BGP neighbor, the originating AS prepends its AS number to the AS-PATH list. The AS-PATH list is not modified between I-BGP peers.

The AS-PATH attribute finds two uses. First, given multiple paths to a destination, BGP-4 will prefer the path with the shortest AS-PATH length. Second, the AS-PATH attribute is effective against routing loops: when an AS receives an update, it discards any prefixes whose AS-PATH list includes its own AS number.

Sometimes an AS will aggregate prefixes it learns from multiple ASs. *Uncle-P* and *Uncle-Q* advertise 180.180.1.64/26 and 180.180.1.128/26, respectively. These prefixes

are aggregated by ISP-B into 180.180.1.0/24. The AS-PATH attribute can indicate that a prefix originated in multiple ASs using AS-SETs. So, when ISP-B advertises 180.180.1.0/24, it can indicate that this aggregate prefix came from ASs 1001 and 1002 using the AS-SET {1001, 1002}. Thus, ISP-Europe would see 180.180.1.0/24 with an AS-PATH of 200, {1001, 1002}, as shown in Figure 7-6.

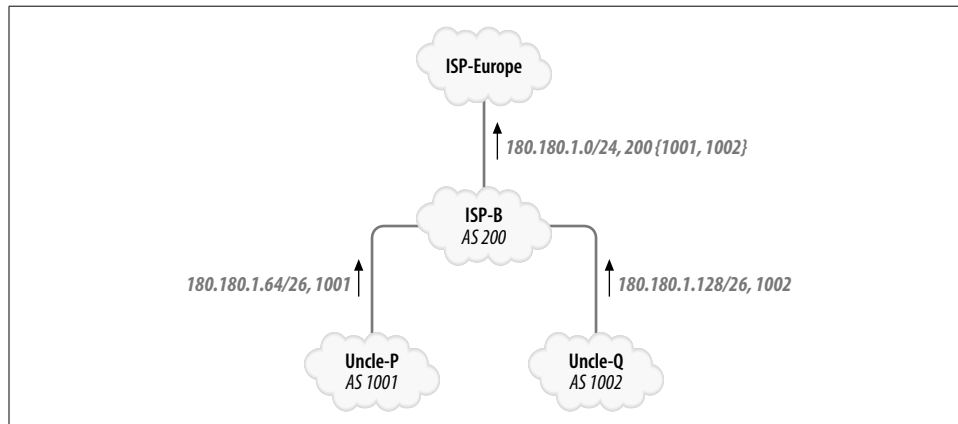


Figure 7-6. AS-PATH attribute

Since the AS-PATH attribute is used in making routing decisions, it is often manipulated to influence inbound routing policies. Say that TraderMary wants all inbound traffic to prefer its ISP-A link. The ISP-B link is to be used only when the ISP-A link is down. To implement its policy, *TrdrMary-1* can advertise 192.200.200.0/24 with a longer AS-PATH length out of ISP-B. We will see a detailed example of this later.

The AS-PATH attribute is often used to set policies such as “do not advertise any prefixes that originate in AS 556” or “prefer paths that traverse AS 905 over AS 111”. A pattern matcher is required that will match AS-PATH attributes to such criteria. Regular expressions (which are also used in Unix for pattern matching, such as in the Unix command *grep*) are used to construct AS-PATH access lists. The following is a brief tutorial on regular expressions.

Most characters and digits in regular expressions match themselves. Thus, the regular expression 10p will match only the string “10p” (but this is not too interesting an example). The following symbols have special significance in regular expressions that will help us construct more interesting examples.

- . (*period*)
Matches any single character.
- *
Matches 0 or more occurrences of the previous regular expression.
- +
Matches 1 or more occurrences of the previous regular expression.

Combining the first two symbols in this list, the regular expression `.*` will match any string. The expression `100.` will match any 4-character string beginning with 100, such as “100a”, “1000”, or “1009”. The expression `1*` will match the empty string, “1”, “11”, “111”, etc.; the expression `1+` will match the same strings, with the exception of the empty string. Here are some other special symbols:

`^` (*caret*)

Matches the beginning of a string.

`$` (*dollar*)

Matches the end of a string.

`_` (*underscore*)

Matches a space character, comma character, left and right braces (`{`, `}`), and left and right parenthesis (`(`, `)`).

Combining the first two symbols, the regular expression `^$` will match the empty string. `^100_` will match any AS-PATH list that begins with 100, in other words, lists such as “100, 200, 130”, “100, 130”, or “100”. Here are some other examples of the use of regular expressions in matching AS-PATH lists:

`^100_`

Matches any sequence starting with 100.

`_100_`

Matches any sequence with 100 somewhere in the path.

`_100$`

Matches any sequence that ends with 100.

`_100_200_`

Matches any sequence with 100 followed by 200 in the path.

`.*`

Matches any sequence starting with the local AS.

`^*`

Matches all ASs.

`^$`

Matches this AS exactly.

More BGP path selections are made on the basis of the AS-PATH attribute than any other attribute. We’ll use the AS-PATH attribute later, in the section “Connecting to the Internet.”

The AS-PATH attribute is well known and mandatory.

NEXT-HOP (type code 3)

Each prefix in an update message is associated with a NEXT-HOP attribute, which describes the IP address of the interface that should receive traffic for the prefix in question. The NEXT-HOP attribute for E-BGP peers is usually the IP address of the

BGP peer advertising the prefix. So, when *ISP-A-1* advertises a default route to *TrdrMary-1*, NEXT-HOP is set to 192.100.100.254.

Consider the network shown in Figure 7-7. There are four routers in the network, joined in a star. *ISP-A-4* is at the center and is not running BGP; *ISP-A-4* learns routes via IGP. *ISP-A-1*, *ISP-A-2*, and *ISP-A-3* have E-BGP peers as shown, have I-BGP peering relationships with each other, and run IGP with *ISP-A-4*.

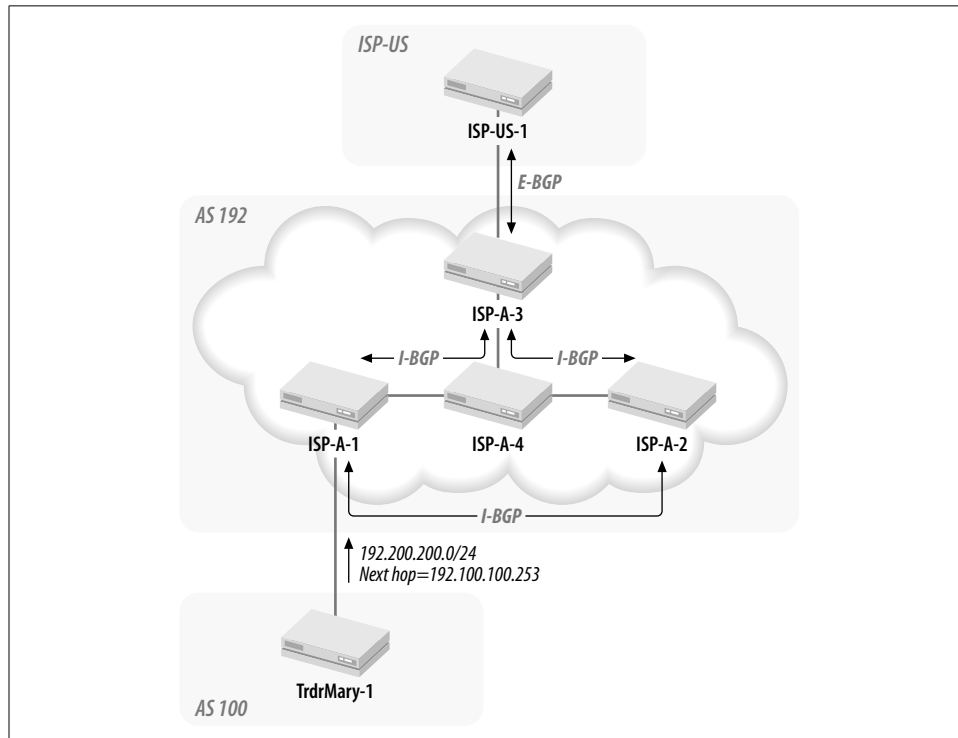


Figure 7-7. NEXT-HOP attribute

TrdrMary-1 advertises 192.200.200.0/24 to *ISP-A-1* via an E-BGP session and sets the NEXT-HOP attribute to itself (192.100.100.253).

ISP-A-1 advertises 192.200.200.0/24 to *ISP-A-2* and *ISP-A-3* via I-BGP sessions. *ISP-A-1* does not modify the NEXT-HOP attribute (which still reads 192.100.100.253). (We saw this in the section “E-BGP Versus I-BGP”—I-BGP neighbors carry the next hop unchanged, allowing the IGP to figure out the best route to the next hop).

ISP-A-4 is not running BGP, so *ISP-A-1* also redistributes the prefix into an IGP so that *ISP-A-4* can forward packets for this destination.

ISP-A-3 advertises the prefix to *ISP-US*, modifying the NEXT-HOP attribute to point to itself.

In other words, E-BGP peers modify the NEXT-HOP attribute to point to their own external IP addresses, whereas I-BGP peers do not modify the NEXT-HOP attribute.

Consider a more interesting use of the NEXT-HOP attribute, in the network shown in Figure 7-8. Routers A, B, and X are on a shared segment. A and B belong to AS 1 and X belongs to AS 2. A and X are E-BGP peers, but B is not a BGP speaker. When A advertises network 30.0.0.0, it would be most appropriate for X to send traffic for this prefix to B and not to A. This feature is referred to as *third-party next hop*.

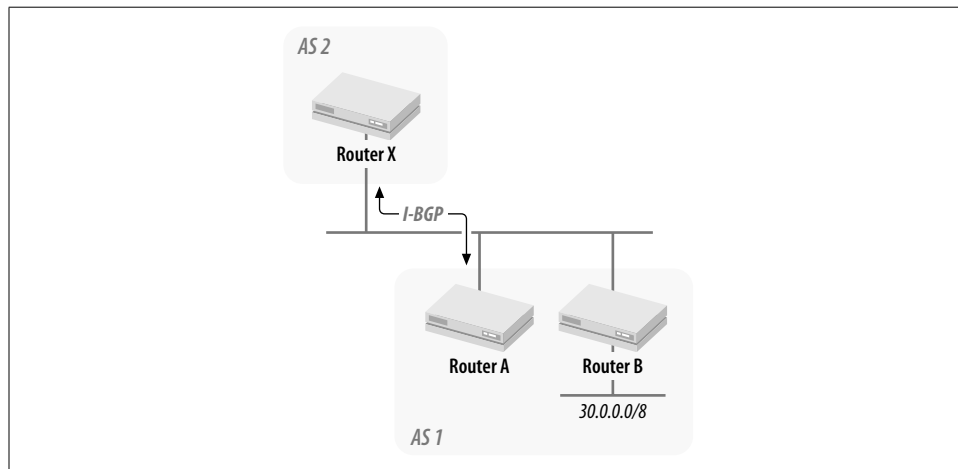


Figure 7-8. Third-party next hop

The NEXT-HOP attribute is well known and mandatory.

MED (type code 4)

Consider the network in Figure 7-9. There are two paths between AS 1 and AS 2. AS 1 prefers to receive traffic from AS 2 on link 1 (as opposed to link 2). AS 1 may use the Multi-Exit Discriminator (MED) attribute to signal its preference to AS 2. Router A advertises 11.0.0.0/8 to X with a MED value of 1; router B advertises 11.0.0.0/8 to Y with a MED value of 10. X and Y are I-BGP peers. A lower MED value indicates the preferred path. Both X and Y will prefer the X → A link to send traffic to AS 1.

The MED attribute is nontransitive, so it is of significance only between a pair of ASs. When AS 2 passes the prefix for network 11.0.0.0 to other ASs, it resets the MED value to 0. Since it is in the interest of ISPs to offload traffic at the closest exit point rather than at another gateway, ISPs usually ignore the MED attribute. The MED attribute may be of more use between two friendly ASs.

The MED attribute is optional and nontransitive. Remember that this attribute is significant only for inbound traffic.

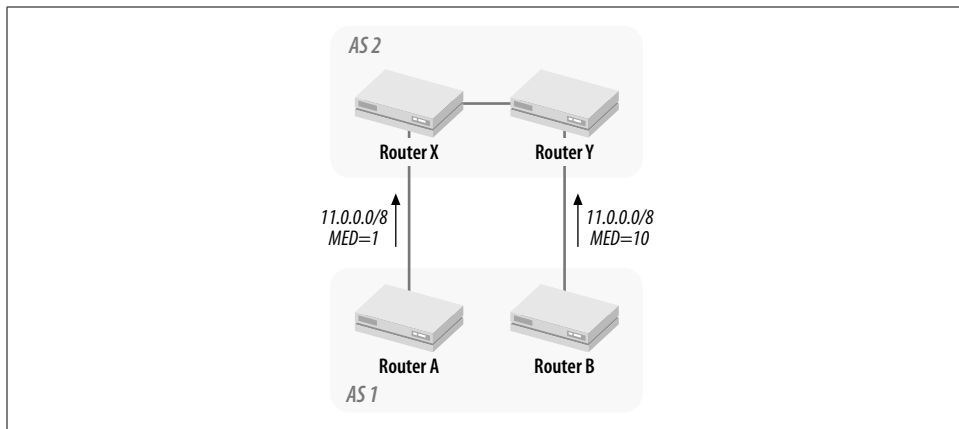


Figure 7-9. Multi-Exit Discriminator (MED) attribute

Weight

The Weight attribute is a Cisco proprietary attribute that is used locally by a router to select a path when multiple paths are available to a prefix. The Weight attribute is not exchanged in any BGP updates (I-BGP or E-BGP).

In the network in Figure 7-10, both P (AS 100) and Q (AS 300) announce 13.0.0.0/8 to R. Let's say that R prefers to route traffic for 13.0.0.0 via AS 100. R can use the Weight attribute to indicate this preference. The following configuration shows how this may be achieved.

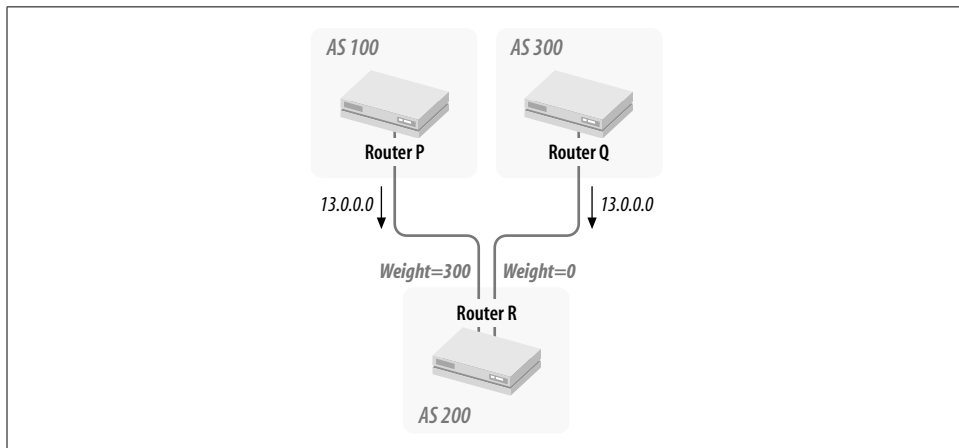


Figure 7-10. Weight attribute

The *route-map weight* is applied to prefixes received from P (line 33). Clause 10 of *route-map weight* matches any prefix that matches access list 101. Access list 101

(line 34) matches the prefix 13.0.0.0/8. Thus, the prefix 13.0.0.0/8 is assigned a Weight of 300 when received from *P*:

```

hostname R
!
interface Serial0
  description * to Q (AS 300) *
  ip address 3.1.1.7 255.255.255.0
!
interface Serial1
  description * to P (AS 100) *
  ip address 3.1.1.254 255.255.255.252
!
router bgp 200
  no synchronization
  neighbor 3.1.1.9 remote-as 300
  neighbor 3.1.1.253 remote-as 100
33  neighbor 3.1.1.253 route-map weight in
!
34  ip access-list 101 permit ip 13.0.0.0 0.255.255.255 255.0.0.0 0.0.0.0
!
    route-map weight permit 10

35  match ip address 101
    set weight 300
!
    route-map weight permit 20

```

The BGP table of *R* shows that 31.0.0.0/8 is received from both *P* (line 36) and *Q* (line 37). Line 36 shows a Weight of 300; line 37 shows a Weight of 0, which is the default value of the Weight attribute for prefixes received from other ASs. A higher value of the Weight attribute is preferable, so the next hop of *P* is installed in the routing table.

```

R#sh ip bgp
...

```

| | Network | Next Hop | Metric | LocPrf | Weight | Path |
|----|-------------|-----------|--------|--------|---------|------|
| 36 | *> 31.0.0.0 | 3.1.1.253 | 10 | | 300 100 | i |
| 37 | * 31.0.0.0 | 3.1.1.9 | 0 | | 0 300 | i |

The Weight attribute impacts only outgoing traffic. The default value of the Weight attribute is 32,768 for locally originated prefixes. This ensures that if a prefix is known via IGP as well E-BGP, the IGP route will be preferred.

LOCAL-PREF (type code 5)

The LOCAL-PREF attribute is similar to the Weight attribute, except LOCAL-PREF is exchanged between I-BGP peers. The LOCAL-PREF attribute is used to select an outgoing path when there are multiple exit points to another AS. A higher LOCAL-PREF value is preferred.

Consider the example of ISP-Europe in Figure 7-11. ISP-Europe may reach Trader-Mary via ISP-B or via ISP-Global and ISP-US. Let's say that ISP-Europe prefers the path via ISP-Global because it is more reliable.

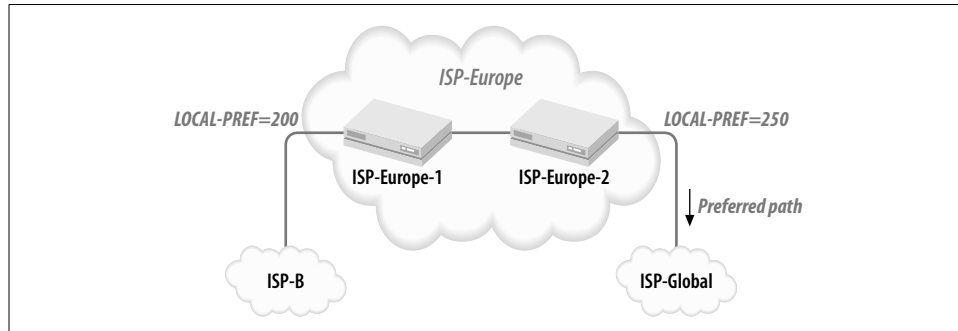


Figure 7-11. Local preference

In this example, *ISP-Europe-1* and *ISP-Europe-2* are I-BGP peers. *ISP-Europe-1* connects to ISP-B and assigns a LOCAL-PREF value of 200 to all prefixes (line 38):

```
hostname ISP-Europe-1
!
interface Ethernet0
 ip address 10.1.1.4 255.255.255.0
!
interface Serial0
 description * to ISP-B *
 ip address 194.1.100.254 255.255.255.252
!
router bgp 2009
 no synchronization
 neighbor 10.1.1.1 remote-as 2009
 neighbor 194.1.100.253 remote-as 200
38 neighbor 194.1.100.253 route-map local-pref in
!
 ip as-path access-list 1 permit _200_
!
 route-map local-pref
  match as-path 1
  set local-preference 200
```

ISP-Europe-2 connects to ISP-Global and assigns a LOCAL-PREF value of 250 to all prefixes (line 39):

```
hostname ISP-Europe-2
!
interface Ethernet0
 ip address 10.1.1.1 255.255.255.0
!
interface Serial0
 description * ISP-Global *
```

```

    ip address 2.1.1.7 255.255.255.0
    !
    router bgp 2009
    no synchronization
39  neighbor 2.1.1.9 remote-as 1300 route-map local-pref in
    neighbor 10.1.1.4 remote-as 2009
    !
    ip as-path access-list 1 permit _1300_
    !
    route-map local-pref
    match as-path 1
    set local-preference 250

```

The BGP tables of both routers reflect these preferences. So, for example, *ISP-Europe-1* installs both prefixes and indicates that the preferred path is via ISP-Global:

```

ISP-Europe-1#sh ip bgp
BGP table version is 9, local router ID is 98.2.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

```

| Network | Next Hop | Metric | LocPrf | Weight | Path |
|-----------------|---------------|--------|--------|--------|--------------------|
| * 192.200.200.0 | 194.1.100.253 | 10 | | 0 | 200 100 i |
| *>i 2.1.1.9 | 2.1.1.9 | 0 | 250 | 0 | 1300 209 192 100 i |
| ... | | | | | |

ISP-Europe-2 shows the following BGP table:

```

ISP-Europe-2#sh ip bgp
...
*> 192.200.200.0 2.1.1.9 10 250 0 1300 209 192 100 i

```

Note that since I-BGP peers do not modify the NEXT-HOP attribute, *ISP-Europe-1* must have an IGP route to 2.1.1.9.

The LOCAL-PREF attribute is similar to the MED and Weight attributes because both affect outbound traffic. However, there are significant differences between the three attributes. The Weight attribute is significant only locally and impacts only outbound traffic. The MED attribute is significant only between two ASs and is used by one AS to control inbound traffic by dictating routing policy to another AS (which that AS may choose to ignore). The LOCAL-PREF attribute is used by the local AS to control its own outbound routing policies.

The LOCAL-PREF attribute is nontransitive and affects only outbound traffic.

Atomic Aggregate (type code 6)

An AS may propagate an aggregate route that causes loss of routing information. Let's say that AS 10 receives the prefixes 11.0.0.0/8 and 11.1.0.0/16 (from different neighbors) with different AS-PATH attributes. If AS 10 propagates only 11.0.0.0/8 to AS 11, AS 11 will have incomplete information about the prefix. In such situations, AS 10 is required to set the Atomic Aggregate attribute to indicate this loss of

information. The Atomic Aggregate attribute should not be set if AS 10 uses the AS-SET attribute to indicate all of the originating ASs.

The Atomic Aggregate attribute is a well-known discretionary attribute.

Aggregator (type code 7)

ISP-X in Figure 7-2 aggregates prefixes received from *Uncle-P* and *Uncle-Q*. ISP-X may use the Aggregator attribute to specify the AS number and BGP router ID of the router performing the aggregation. The length of this attribute is always 6 octets: 4 for the AS number and 2 for the BGP router ID.

The Aggregator attribute is optional and transitive.

Community

Imagine the international flights descending upon London every few minutes from Nairobi, Naples, New Delhi, New York... The only way for an immigration officer to deal with this madness is to assign one or more categories to each arriving passenger—Political Refugee, Senior, Criminal, Musician-type, Brit, Au Pair, etc. Based on the assigned category, the immigration officer can make quick decisions regarding admission into the U.K.

Just like the immigration officer, an AS can assign a community to each prefix. Subsequent routing decisions can then be made based on the Community attribute.

In the following example, ISP-Finland attaches a Community attribute of 999 for prefixes learned from its clients (SisterY, in this example). ISP-Finland may attach a different Community attribute to prefixes learned from other ISPs.

```
hostname ISP-Finland-1
!
router bgp 1200
 neighbor 12.100.1.253 remote-as 108
 neighbor 12.100.1.253 route-map CLIENT in
!
ip as-path access-list 2 permit ^108$
!
route-map CLIENT permit 10
 match as-path 2
 set community 999
```

An examination of the details of the attributes attached to the received prefix will show the value of the Community attribute (line 40):

```
ISP-Finland-1#sh ip bgp 199.199.3.0
BGP routing table entry for 199.199.3.0/24, version 5
Paths: (1 available, best #1, advertised over I-BGP)
 108
   12.100.1.253 from 12.100.1.253 (192.168.1.10)
     Origin incomplete, metric 0, valid, external, best
40      Community: 999
```

Subsequent routing decisions may now be based on this Community attribute. So, all prefixes that match a Community attribute of 999 could be advertised to other ISPs, whereas prefixes learned from other ISPs may not be advertised to other ISPs. This will ensure that ISP-Finland is not misused as a transit ISP by other ISPs.

The Community attribute is optional and transitive.

Path Selection

Like RIP and IGRP, BGP is a DV protocol that uses the lowest metric to select the best path to a destination. Unlike RIP and IGRP, BGP's decision process is relatively complex. This complexity is due to the number of BGP attributes; each BGP attribute has a place in the decision process. Of course, if there is only one path to a prefix, the decision process described in this section is unnecessary: that single path wins. Unlike RIP and IGRP, BGP's decision process always yields a single best path: BGP does not install multiple paths to a destination (nor does it load-balance traffic over multiple paths).

Let's look at the BGP decision process. The input to this algorithm is a number of paths to the same prefix (with the same prefix length), known via BGP. Each path is accompanied by a set of attributes. The output of the algorithm is a single best path to the prefix. The best path is a candidate for advertisement to other BGP peers and to be placed into the routing table.

1. Choose the path with the highest Weight, a Cisco proprietary attribute. If the paths cannot be discriminated based on the Weight attribute, continue to the next criterion.
2. Choose the path with the highest LOCAL-PREF value. If the paths cannot be discriminated based on the LOCAL-PREF attribute, continue to the next criterion.
3. Choose the path that was locally originated via a *network* or *aggregate* command. If the paths cannot be discriminated based on this criterion, continue to the next criterion.
4. Choose the path with the shortest AS-PATH attribute. If the paths cannot be discriminated based on the AS-PATH attribute, continue to the next criterion. To disable the AS-PATH attribute as a factor in the selection of the best route, use the *bgp bestpath as-path ignore* command.
5. Choose the path with the lowest ORIGIN attribute. (IGP is lower than EGP, EGP is lower than Incomplete). If the paths cannot be discriminated based on the ORIGIN attribute, continue to the next criterion.
6. Choose the path with the lowest MED attribute. If the paths cannot be discriminated based on the MED attribute, continue to the next criterion. By default, the MED attribute is considered only when a prefix is received from neighbors in the same AS. To allow the comparison of the MED attribute when the prefix

is received from neighbors in different ASs, use the BGP *always-compare-med* command.

7. Choose an E-BGP path over an I-BGP path. If the paths cannot be discriminated based on this criterion, continue to the next criterion.
8. Choose the path with the lowest IGP metric to the next hop. If the paths cannot be discriminated based on IGP metric, continue to the next criterion.
9. Choose the path originated by the BGP router with the lowest router ID.

Load Balancing

As per RFC 1771, BGP installs only one best path to a destination network. This scenario leaves little room to load-balance over multiple paths. However, it is possible to use an IGP (such as IGRP) to achieve load balancing between ASs.

In the network in Figure 7-12, ISP-A and ISP-B set up two links between each other over which traffic is to be load-balanced.

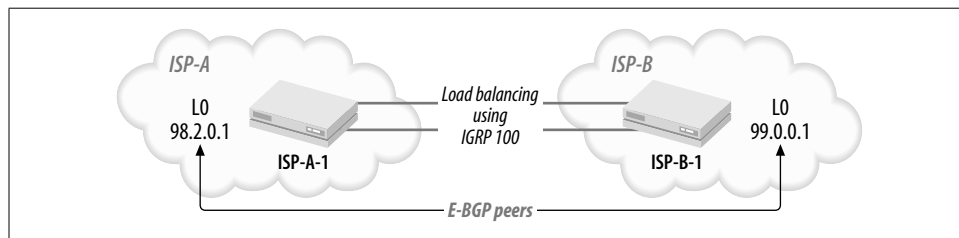


Figure 7-12. Load balancing

Both the peering routers set up loopback addresses (lines 41 and 46 in the following code blocks). BGP sessions between the peers are then established between the peers using these loopback addresses (lines 42 and 47 specify the loopback addresses of the neighbor; lines 43 and 48 say that the BGP TCP session should originate from the local loopback address). Since BGP normally expects its peers to be on a directly connected network, the *ebgp-multihop* command (lines 44 and 49) relaxes this restriction. None of this will work if each peer cannot route to the other's loopback IP address. Lines 45 and 50 set up IGRP between *ISP-A-1* and *ISP-B-1*, which permits the routers to share path information. Since IGRP will use both paths between the ASs to route to the peer's loopback address, all traffic between the ASs will use both paths.

```

hostname ISP-A-1
!
41 interface Loopback0
   ip address 98.2.0.1 255.255.0.0
!
interface Serial2
   description * to ISP-B *
```

```

        ip address 100.1.1.1 255.255.255.0
        !
    interface Serial3
        description * to ISP-B *
        ip address 100.1.2.1 255.255.255.0
        !
    router bgp 192

42 neighbor 99.0.0.1 remote-as 200
43 neighbor 99.0.0.1 update-source loopback0
44 neighbor 99.0.0.1 ebgp-multihop
    !
45 router igrp 100
    network 100.1.0.0
    network 98.0.0.0

```

The configuration of *ISP-B-1* mirrors that of *ISP-A-1*:

```

hostname ISP-B-1
!
46 interface Loopback0
    ip address 99.0.0.1 255.255.0.0
    !
interface Serial2
    description * to ISP-A *
    ip address 100.1.1.2 255.255.255.0
    !
interface Serial3
    description * to ISP-B *
    ip address 100.1.2.2 255.255.255.252
    !
router bgp 200
47 neighbor 98.2.0.1 remote-as 192
48 neighbor 99.0.0.1 update-source loopback0
49 neighbor 99.0.0.1 ebgp-multihop
    !
50 router igrp 100
    network 100.1.0.0
    network 99.0.0.0

```

Route Filtering

The filtering of routes between ASs is key to implementing routing policies. The following section shows several route-filtering techniques.

Filtering by Prefix (Address/Mask) Information

The following BGP sub-command may be used to filter updates from a neighbor based on the IP prefix in the update packet:

```

neighbor ip-address distribute-list {access-list-number | name/prefix-list
prefixlistname} {in | out}

```

ip-address is the address of the BGP peer. The IP prefixes to be filtered may be specified in an access list or a prefix list.

Consider TraderMary's network again. An access list would be appropriate to block *TrdrMary-1* from learning its own internal numbers from ISP-A:

```
hostname TrdrMary-1
!
interface Serial1
  description * to ISP-A *
  ip address 192.100.100.253 255.255.255.252
...
router bgp 100
neighbor 192.100.100.254 remote-as 192
neighbor 192.100.100.254 distribute-list 1 in
!
access-list 1 deny 160.160.0.0
access-list 1 deny 192.200.200.0
access-list 1 permit 0.0.0.0 255.255.255.255
```

Simple access lists do not allow control over the subnet mask field. So, ISP-X may advertise 192.156.0.0/16 to peer *a.b.c.d* as follows:

```
hostname ISP-X-1
!
router bgp 222
neighbor a.b.c.d
neighbor a.b.c.d distribute-list 10 out
access-list 10 permit 192.156.0.0
```

However, this access list will permit 192.156.0.0/16, 192.156.0.0/17, 192.156.0.0/18, and so on. To ensure that ISP-X advertises only 192.156.0.0/16, we need to configure an extended access list that has room to specify the mask portion of the update:

```
access-list 101 permit ip 192.156.0.0 0.0.255.255 255.255.0.0 0.0.0.0
```

The format of the extended access list is:

```
access-list <number> permit ip <ip-address> <don't care bits> <mask> <don't care bits>
```

ISPs may use extended access lists on inbound updates to filter out all advertisements with masks longer than a specific length. Here, access list 102 will filter out all updates with masks longer than 24 bits:

```
access-list 102 deny ip 0.0.0.0 255.255.255.255 255.255.255.0 0.0.0.255
```

You can also use an IP prefix list for this task. The format of the *ip prefix-list* command is:

```
ip prefix-list list-name [seq seq-value] deny | permit network/len [ge ge-value]
[le le-value]
```

Here, the prefix list *xyz* allows only 192.156.0.0/16:

```
ip prefix-list xyz permit 192.156.0.0/16.
```

Note that a modification to the filters on a BGP peer with an existing BGP session will not take effect until the connection is reset using the *clear ip bgp* command.

Filtering by AS-PATH Information

The *ip as-path access-list* command permits updates to be filtered based on the attached AS-PATH information:

```
ip as-path access-list access-list-number {permit | deny} as-regular-expression
```

The *access-list-number* should be in the range 1–199. The keyword *permit/deny* specifies the action to be taken if the AS-PATH information in the update matches the *as-regular-expression*.

Let's say that TraderMary wants only updates that transit AS 131. The following configuration creates a route map called *only131* that refers to the AS-PATH access list 1, which matches only AS-PATH strings that include 131:

```
hostname TrdrMary-1
...
ip as-path access-list 1 permit _131_
!
router bgp 100

    neighbor 192.100.100.254 remote-as 192
    neighbor 192.100.100.254 route-map only131 in
!
route-map only131 permit 10
match as-path 1
```

Connecting to the Internet

Several key design issues should be considered when connecting to the Internet. These issues impact the reliability, performance and cost of Internet connectivity. I will examine several design alternatives in the following sections, then look at a case study.

Design Alternatives

There are three alternatives regarding physical connectivity between the client and the ISP(s):

Singly-homed

A single circuit may be adequate for a small organization generating a trickle of traffic. The client organization must choose an ISP and decide on the speed of the access circuit. However, if that single circuit breaks, the entire organization will be without Internet service. Hence, if the organization's access to the Internet is critical, multi-homing is warranted (even when the traffic volume is small).

Multi-homed to the same ISP

An organization may decide to implement multiple circuits between itself and its ISP for reliability. An organization that decides to multi-home has a more complex task at hand. Should the links be of the same speed? What should the link speeds be? How should inbound traffic be distributed over the links? How should outbound traffic be distributed over the links?

Multi-homed to different ISPs

A multi-homed organization may consider using more than one ISP for additional reliability. This will guard against failures in a single ISP network.

There are several options regarding the routing of traffic between the client organization and the Internet. Note that these options apply only to multi-homed clients; singly-homed clients have only one path for inbound and outbound traffic.

In the following discussion, I will distinguish between *inbound* and *outbound* traffic. The traffic flow from the client's network to the Internet is referred to as outbound. The reverse flow from the Internet to the client's network is referred to as inbound. Inbound and outbound flows need not be symmetrical.

There are several options regarding outbound traffic:

Default route

The simplest method of defining routes to external destinations is by configuring a default route (0.0.0.0). The default route will match any destination for which a more specific route is not known via the client's IGP. The configuration for the singly-homed client using a default route is identical to TraderMary's configuration in "Getting BGP Running." The single default route works well in a singly-homed scenario.

Partial routing table along with default routes

When a client is multi-homed—say, to two different ISPs—the two outbound paths are not equal. Some networks in the Internet will be closer via one path while others will be closer via the second path. A single default route cannot address this asymmetry. The client has two options here. The first option is for the client routers to import a partial set of routes from each ISP and use a default route for the remaining routes. The second option is to import the full routing table.

Full routing table

The full routing table may be useful in multi-homed environments to allow the most informed decision to be made. However, at the time that this book is being written, there are around 75,000 prefixes in the Internet. This places high demands on router memory and CPU resources. You should have a very good reason to import the full routing table.

Inbound routing decisions are made in the routing tables in other, external ASs. The client may set one or more BGP attributes in an attempt to influence the flow of

inbound traffic. However, ISPs have their own routing policies and may disregard or override these attributes. Hence, the inbound routing policies should be implemented in conjunction with the ISP. The methods described here use the AS-PATH attribute.

Identical AS-PATH length

The option here is for the client organization to advertise all its routes on all paths with identical AS-PATH information.

AS-PATH prepend

The client organization may consider the AS-PATH *prepend* option to influence the path that inbound traffic will take.

The following case study implements the AS-PATH *prepend* option to load-balance traffic over two links.

A Case Study

BollywoodFilms has offices in Bombay and Madras in India. The corporation desires Internet connectivity. Given the notoriously poor telecommunications infrastructure in the South Asian subcontinent, they decide to establish two connections to the local ISP, ISP-SouthAsia. One connection is out of router *Bombay* and the other is out of router *Madras*. BollywoodFilms thus multi-homes to the same ISP. Figure 7-13 shows a topology of BollywoodFilms’s external network.

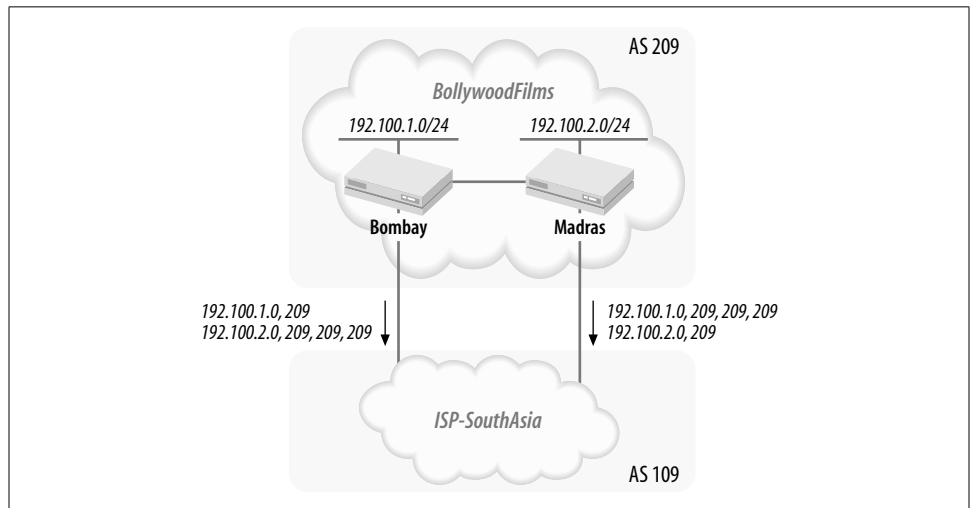


Figure 7-13. Internet connectivity for BollywoodFilms

BollywoodFilms owns networks 192.100.1.0/24 and 192.100.2.0/24. 192.100.1.0/24 connects to *Bombay* (line 51) and 192.100.2.0/24 connects to *Madras* (line 61). The configurations on the routers are as follows:


```
hostname Bombay
!  
51 interface Ethernet0  
    ip address 192.100.1.1 255.255.255.0  
    !  
52 interface Serial0  
    ip address 146.100.100.254 255.255.255.252  
    !  
53 interface Serial1  
    ip address 10.1.1.4 255.255.255.0  
    !  
router bgp 209  
54 network 192.100.1.0  
55 network 192.100.2.0  
56 neighbor 10.1.1.1 remote-as 209  
57 neighbor 192.100.100.253 remote-as 109  
58 neighbor 192.100.100.253 route-map metric-adj2 out  
    !  
access-list 1 permit 192.100.2.0 0.0.0.255  
access-list 2 permit 192.100.1.0 0.0.0.255  
    !  
59 route-map metric-adj2 permit 10  
    match ip address 1  
    set as-path prepend 209 209  
    !  
60 route-map metric-adj2 permit 20  
    match ip address 2  
    !  
hostname Madras  
!  
61 interface Ethernet0  
    ip address 192.100.2.1 255.255.255.0  
    !  
62 interface Serial0  
    ip address 146.1.1.254 255.255.255.252  
    !  
63 interface Serial1  
    ip address 10.1.1.1 255.255.255.0  
    !  
router bgp 209  
    no synchronization  
64 network 192.100.1.0  
65 network 192.100.2.0  
66 neighbor 10.1.1.4 remote-as 209  
67 neighbor 200.1.1.253 remote-as 109  
68 neighbor 200.1.1.253 route-map metric-adj out  
    !  
access-list 1 permit 192.100.1.0 0.0.0.255  
access-list 2 permit 192.100.2.0 0.0.0.255  
    !  
69 route-map metric-adj permit 10  
    match ip address 1
```

```

    set as-path prepend 209 209
    !
70 route-map metric-adj permit 20
    match ip address 2

```

Note that both routers advertise 192.100.1.0 and 192.100.2.0 to the ISP (lines 54, 55, 64, and 65). This ensures that inbound traffic has an alternate path if one link fails. So, if *Bombay*'s link to the ISP fails, external users on 192.100.1.0 will receive inbound traffic via *Madras*, since *Madras* advertises 192.100.1.0 to the ISP. This configuration works because *Bombay* and *Madras* have a link between each other (lines 53 and 63) over which they have an I-BGP session (lines 56 and 66).

If you look a little deeper into the above configurations (lines 58, 59, 68, and 69), you'll see that *Bombay* and *Madras* do not advertise 192.100.1.0 and 192.100.2.0 equally. *Bombay* makes the AS-PATH attribute for 192.100.2.0 look unattractive to routers in the Internet by lengthening its AS-PATH attribute using the *as-path prepend* command. *Madras* uses the same technique to make 192.100.1.0 look unattractive to Internet routers. This ensures that when both ISP links are up, inbound traffic for 192.100.1.0 prefers to come directly into *Bombay* and inbound traffic for 192.100.2.0 prefers to come directly into *Madras*. Thus, a router in the ISP's network would have BGP table entries with two paths for each prefix:

```

ISP-SAsia#sh ip bgp
...
Network          Next Hop          Metric LocPrf Weight Path
71 * 192.100.1.0   146.1.1.254      0          0 209 209 209 I
   *>             146.100.100.254 0              0 209 i
72 *> 192.100.2.0   146.1.1.254      0          0 209 i
   *              146.100.100.254 1              0 209 209 209 i

```

However, the ISP router would prefer the shorter AS-PATH:

```

ISP-SAsia#sh ip route
...
73 B 192.100.1.0/24 [20/0] via 192.100.100.254, 00:05:58
74 B 192.100.2.0/24 [20/0] via 200.1.1.254, 00:01:44

```

ISP-SouthAsia thus sees a shorter path for 192.100.1.0 via *Bombay* (line 71) and a shorter path for 192.100.2.0 via *Madras* (line 72). If one of these lines goes down, all inbound traffic will reroute to the other link, since the BGP tables store both paths.

However, this completes only half the screenplay. The organization decides that outbound traffic from *Bombay* should exit via *Bombay* and outbound traffic from *Madras* should exit via *Madras*. To implement this policy, ISP-SouthAsia sends a default route to *Bombay* and *Madras*, respectively. So *Bombay* installs a default route that points out of its serial interface to ISP-SouthAsia (line 75) and *Madras* installs a default route that points out of its serial interface to ISP-SouthAsia (line 76):

```

Bombay#sh ip route
...
Gateway of last resort is 192.100.100.253 to network 0.0.0.0

```

75 B* 0.0.0.0/0 [20/0] via 192.100.100.253, 00:07:21
...

```
Madras#sh ip route
...
Gateway of last resort is 200.1.1.253 to network 0.0.0.0
76 B* 0.0.0.0/0 [20/0] via 200.1.1.253, 00:04:47
...
```

If *Bombay* loses its link to ISP-SouthAsia, it will use the default route it receives from *Madras* via the I-BGP peering relationship:

```
Bombay#sh ip bgp
BGP table version is 72, local router ID is 192.100.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop          Metric LocPrf Weight Path
77  *> 0.0.0.0        192.100.100.253    0           0 109 ?
   * i              200.1.1.253        0    100       0 109 ?
```

And, likewise, *Madras* will use the default route it receives from *Bombay*:

```
Madras#sh ip bgp
BGP table version is 80, local router ID is 192.100.2.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop          Metric LocPrf Weight Path
78  *> 0.0.0.0        200.1.1.253        0           0 109 ?
   * i              192.100.100.253    0    100       0 109 ?
```

Choosing an ISP

ISPs may be evaluated against several criteria that may be broadly classified under the headings Services, Architecture, Addressing/Routing, Operations Support, and Pricing. Some criteria will, of course, be more important to your organization than others.

Services

First and foremost is whether the ISP provides the services you need at your location. If you need T-3 access in Lisbon, Portugal, can the ISP meet your requirement?

Network Architecture

Is your application so critical that even a short outage would be intolerable? If so, you should look closely at the ISP's network architecture. Are there redundant routers at the points of presence (POPs)? Are there redundant links between POPs? What is the speed of the links between POPs?

What other ISPs and providers are peers of the ISP? If you are a wine merchant in Portugal and the majority of your distributors and customers are on ISP-Japan, you should find out how far your ISP is from ISP-Japan. Do they have direct peering? If not, how many intermediary networks are involved?

Addressing/Routing

Can the ISP carry the prefixes you want advertised? This may be an issue if you borrow a prefix from ISP-P but want to advertise the same prefix to ISP-Q.

Will the ISP support the routing policies you desire?

Operations

How is the ISP managed? What processes and resources does the ISP have in place for managing its resources? How does the ISP monitor traffic on its backbone? At what level of utilization is the network bandwidth upgraded? How is the network monitored for failures? What is the average downtime?

A client's first contact with the ISP will usually be with its Customer Service Department. The quality of response from the Customer Service Department can make a big difference when you are experiencing an outage and need "real"-time access to high-level engineering support. Readily available engineering design support and readily available, proficient engineers to help troubleshoot are important.

Price

Price, of course, is a concern. You may get a better price from a smaller, regional ISP than from one of the major ISPs. You may even get better service from a regional ISP. However, a larger ISP may score higher marks on its network architecture and performance.

Troubleshooting BGP

You might encounter your first problem with BGP when configuring a new peering relationship. The BGP session between the peers may not enter the Established state (the output of *show ip bgp neighbor* may show other states, such as *Idle*, *Connect*, or *Active*). Here are the first steps you should take when troubleshooting BGP:

1. Check the infrastructure between the peers: an ICMP ping test between the peers is a quick test of layer-3 reachability between the peers.
2. If layer 3 reachability exists between the peers, the configuration of the peers may be in error. Check the BGP configuration of neighbor IP addresses and AS numbers on each peer.

3. Check for IP filters (access lists) or firewalls between the peers that would prevent a BGP TCP session (on port 179) from being formed.
4. Is there a BGP version mismatch between the peers? Look for the BGP *neighbor ip-address version number* sub-command.

Once the neighbors have entered into the Established state, you may find that no prefixes or only some prefixes are being exchanged. Here are the steps you should take:

- If no prefixes are being exchanged, you may want to go back and make sure that the peers are indeed in an Established state.
- If some but not all prefixes are being exchanged, check to see if a filter (an access list, a prefix filter, or an attribute filter such as an AS-PATH filter) is blocking the update. The *show ip bgp neighbor ip-address* command will show the filters that are applied to the BGP session.
- Filter changes on established BGP sessions do not take effect until the BGP session is reset via a *clear* command. BGP connections need to be reset if any BGP policy (filter) changes are made. If the BGP table does not reflect the filters in place, issue one of the *clear* commands described here. This command:

```
clear ip bgp [* | ip-address | peer-group]
```

will tear down BGP sessions with the specified neighbors. Once the sessions are reestablished, the routing tables will reflect the current filter lists. However, the neighbors will lose routes learned from each other, disrupting traffic between the peers during session reestablishment. Cisco offers a new, softer approach that enables new policies to take effect without resetting the BGP TCP connection. This command:

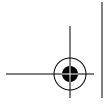
```
clear ip bgp [* | ip-address | peer-group] [soft [in|out]]
```

is less harsh. The *in* option applies the current filter list on prefixes in the BGP table and the *out* option causes updates to be sent again.

Intermittent connectivity between end-stations is often the result of a flapping interface. Check the output of the *show ip bgp neighbor* command—an increasing version number indicates a problem in the infrastructure between the peers.

You may encounter a seemingly bizarre situation in which you are able to get to some parts of the Internet but not others. This may be because of the way ISPs advertise or block various prefixes depending on the length of the prefix. This can be a messy situation to troubleshoot, but there are several Internet sites that allow one to trace routes to any IP address. These sites are also called “looking-glass sites.” <http://www.nanog.org> provides a wealth of data with pointers to other sites, including looking-glass sites. You can also check out <http://www.merit.edu>.

Also, some ISPs will tell you what routes they are seeing for your network numbers if you call and ask. AT&T even maintains a router on which you can check routes



(*telnet route-server.cerf.net*), which will respond to *show ip route* commands (no password is required).

Summing Up

It's hard to remember that BGP is based on the simple DV algorithm. Instead of one metric (as in the case of RIP or IGRP), the protocol uses multiple attributes to choose the best path.

The environment on which I focused in this chapter—the use of BGP by ASs such as TraderMary to connect to one or more ISPs—is even simpler in that only a few attributes are necessary to implement most routing policies. The essentials in this environment are deciding on the optimum routing policy for inbound and outbound traffic given the client's topology and requirements, and using BGP to implement this policy well, so that failures and load-balancing issues are adequately addressed.

ISPs typically employ BGP in more complex ways, especially with route reflectors and confederations (to solve the problem of I-BGP meshes). This chapter did not focus on these issues.

