



Chapter

5

Advanced Border Gateway Protocol (BGP)

JNCIS EXAM OBJECTIVES COVERED IN THIS CHAPTER:

- ✓ Identify the functionality and alteration of the BGP attributes
- ✓ Describe the operation and configuration of a route reflection BGP network
- ✓ Describe the operation and configuration of a confederation BGP network
- ✓ Identify the characteristics of multiprotocol BGP and list the reasons for enabling it



In this chapter, we examine the methods available within the JUNOS software to manipulate and alter some of the BGP attributes, including Origin, AS Path, Multiple Exit Discriminator, and Local Preference. Following this, we explore methods for scaling an IBGP full mesh using route reflection and confederations. We see how each method operates, how each modifies a route's attributes, and how each is configured on the router. We conclude the chapter with a discussion on Multiprotocol BGP, including when to use it and how it operates.

Modifying BGP Attributes

The JUNOS software provides methods for altering and modifying most of the BGP attributes. This allows you to control which routes are accepted or rejected by a peering session. Additionally, you have the ability to alter the selection of the active BGP route when you modify certain attributes. Some of the attributes are changeable through a configuration, a routing policy action, or both. Let's examine each attribute in some detail.

Origin

The Origin attribute provides a BGP router with information about the original source of the route. The attribute is included in every routing update and is a selection criterion in the route selection algorithm. After discussing the default setting of the attribute, we see how to modify its value with a routing policy.

Default Origin Operation

A Juniper Networks router, by default, forwards all active BGP routes in the `inet.0` routing table to the appropriate established peers. The injection of new routing knowledge into BGP occurs when you apply a routing policy to the protocol. This policy matches some set of routes in the routing table and accepts them. This action causes each newly injected route to receive an Origin value of IGP (I).

Figure 5.1 shows three IBGP peers within AS 65010: Cabernet, Merlot, and Zinfandel. The Cabernet router has locally configured static routes in the `172.16.1.0/24` address range and the Zinfandel router has customer static routes in the `172.16.2.0/24` address space. We've configured a policy on each router to match the static routes and accept them. When applied as an export policy, the Merlot router receives the following routes:

```
user@Merlot> show route protocol bgp terse
```

```
inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
```

+ = Active Route, - = Last Active, * = Both

A Destination	P Prf	Metric 1	Metric 2	Next hop	AS path
* 172.16.1.4/30	B 170	100	0	>10.222.1.1	<u>I</u>
* 172.16.1.8/30	B 170	100	0	>10.222.1.1	<u>I</u>
* 172.16.2.4/30	B 170	100	0	>10.222.3.1	<u>I</u>
* 172.16.2.8/30	B 170	100	0	>10.222.3.1	<u>I</u>

The JUNOS software always displays the Origin attribute in conjunction with the AS Path attribute. In the output from Merlot, we see that the AS path column contains no AS values (native IBGP routes) but has the Origin listed as I, for IGP routes. This default behavior is also exhibited when the Zinfandel router exports its IS-IS learned routes into BGP:

```
user@Zinfandel> show route protocol isis
```

```
inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

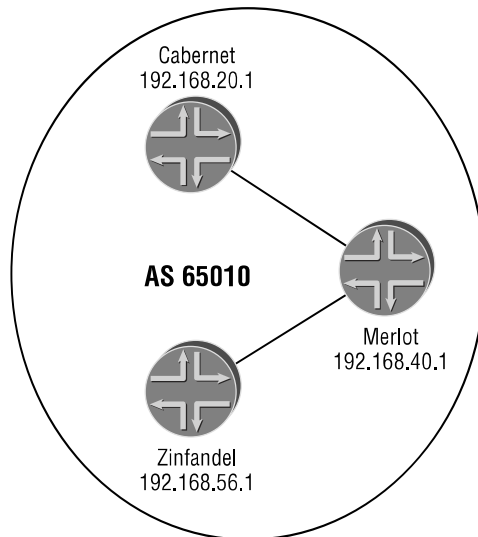
```
10.222.1.0/24      *[IS-IS/18] 15:56:32, metric 20
                  to 10.222.2.1 via at-0/1/0.0
                  > to 10.222.3.2 via at-0/1/1.0
192.168.20.1/32   *[IS-IS/18] 15:56:32, metric 10
                  > to 10.222.2.1 via at-0/1/0.0
192.168.40.1/32   *[IS-IS/18] 15:57:02, metric 10
                  > to 10.222.3.2 via at-0/1/1.0
```

```
user@Zinfandel> show route advertising-protocol bgp 192.168.40.1
```

```
inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
  Prefix          Nexthop          MED    Lclpref  AS path
* 10.222.1.0/24   10.222.3.2      20     100      I
* 172.16.2.4/30   Self            0      100      I
* 172.16.2.8/30   Self            0      100      I
* 192.168.20.1/32 10.222.2.1      10     100      I
* 192.168.40.1/32 10.222.3.2      10     100      I
```



The JUNOS software uses the existing next-hop value for routes redistributed from an IGP. This avoids potential suboptimal routing in the network.

FIGURE 5.1 Origin sample network

Altering the Origin Attribute

You have the option of setting the Origin attribute to any of the three possible values in a routing policy. This is accomplished with the `then origin value` policy action. All of the possible Origin values are represented within the policy action as `egp`, `igp`, and `incomplete`. The current policy used on the Zinfandel router in Figure 5.1 is as follows:

```
user@Zinfandel> show configuration policy-options
policy-statement advertise-routes {
  term statics {
    from protocol static;
    then accept;
  }
  term isis {
    from protocol isis;
    then accept;
  }
}
```

To mirror the operation of another router vendor, the administrators of AS 65010 would like the IS-IS routes to be advertised with an incomplete Origin value. After modifying the routing policy, we can verify its operation:

```
[edit policy-options policy-statement advertise-routes]
user@Zinfandel# show
```

```

term statics {
    from protocol static;
    then accept;
}
term isis {
    from protocol isis;
    then {
        origin incomplete;
        accept;
    }
}

```

```
user@Zinfandel> show route advertising-protocol bgp 192.168.40.1
```

```
inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lclpref  AS path
* 10.222.1.0/24         10.222.3.2      20     100      ?
* 172.16.2.4/30         Self            0      100      I
* 172.16.2.8/30         Self            0      100      I
* 192.168.20.1/32       10.222.2.1      10     100      ?
* 192.168.40.1/32      10.222.3.2      10     100      ?
```

In a similar fashion, we can again alter the **advertise-routes** policy to modify the Origin for the static routes. For the sake of completeness, we advertise these routes with a value of EGP (E):

```

[edit policy-options policy-statement advertise-routes]
user@Zinfandel# show
term statics {
    from protocol static;
    then {
        origin egp;
        accept;
    }
}
term isis {
    from protocol isis;
    then {
        origin incomplete;
        accept;
    }
}

```

```
user@Zinfandel> show route advertising-protocol bgp 192.168.40.1
```

```
inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED      Lc1pref  AS path
* 10.222.1.0/24         10.222.3.2      20       100      ?
* 172.16.2.4/30        Self             0        100      E
* 172.16.2.8/30        Self             0        100      E
* 192.168.20.1/32      10.222.2.1      10       100      ?
* 192.168.40.1/32     10.222.3.2      10       100      ?
```

AS Path

The AS Path attribute is also included in every BGP routing update. It supplies information about the AS networks a particular route has transited, and this information is used to select the active BGP route. Additionally, the AS Path provides a routing loop avoidance mechanism as all received routes are dropped when any local AS value appears in the path. The JUNOS software provides multiple methods for altering this attribute, including both configuration options and routing policy actions. Each requires some explanation before providing the details of its usage.

Modifying the AS Path: Configuration Statements

According to the BGP specifications, an implementation should not remove any information from the AS Path attribute. Only additional information should be added to the beginning of the path using a prepend action. While this is nice in theory, real-world problems have required vendors to bend the letter of the specification while trying to maintain its spirit. To this end, the JUNOS software provides four different methods for altering the default operation of the AS Path attribute using configuration statements. Let's explore each of these in further detail.

Removing Private AS Values

The Internet Assigned Numbers Authority (IANA) has set aside several AS values for private use in networks. These AS numbers begin at 64512 and continue to 65534, with 65535 as a reserved value. Much like the private IP address ranges, these private AS numbers should not be attached to routes advertised to the Internet.

Figure 5.2 shows a service provider with an assigned AS of 1111. This provider has customers who would like to connect using BGP but who don't have an assigned AS number. These customers would like to have multiple links available to the provider for load balancing and fail-over redundancy. To facilitate the needs of the customer, the provider assigns each customer a private AS number and configures BGP between the networks. The result of this configuration is the leaking of the private AS number to the Internet, as seen on the Zinfandel router in AS 2222:


```
user@Zinfandel> show route protocol bgp terse
```

```
inet.0: 15 destinations, 15 routes (15 active, 0 holddown, 0 hidden)
```

+ = Active Route, - = Last Active, * = Both

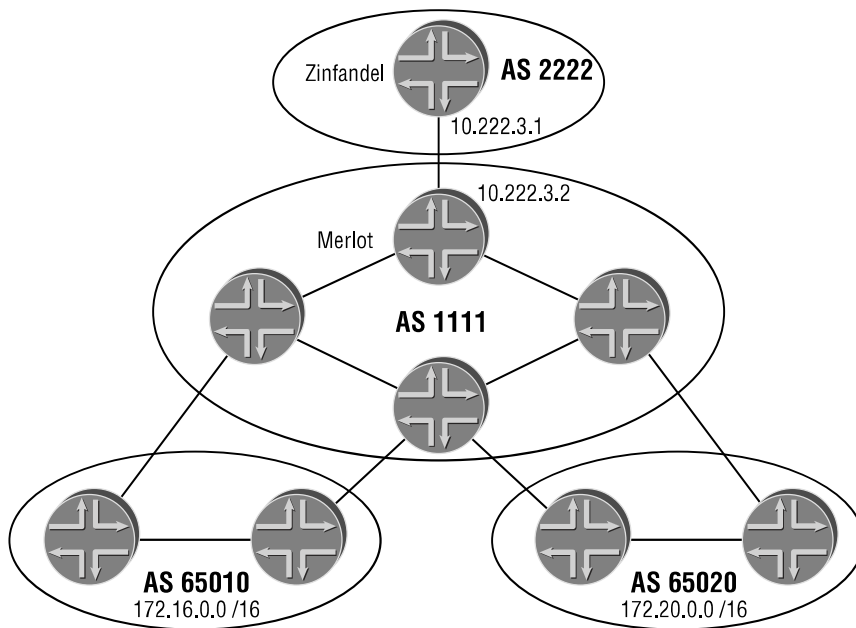
A Destination	P Prf	Metric 1	Metric 2	Next hop	AS path
* 172.16.1.0/24	B 170	100		>10.222.3.2	<u>1111 65010 I</u>
* 172.20.4.0/24	B 170	100		>10.222.3.2	<u>1111 65020 I</u>

One possible solution to this problem is the re-generation of the customer routes within AS 1111. This requires configuring local static routes and advertising them into BGP with a routing policy. While quite effective in preventing the advertisement of the private AS numbers to the Internet, this solution is not very scalable since each possible customer route needs to be duplicated within AS 1111. A more dynamic solution can be found through the use of the `remove-private` configuration option in the JUNOS software. This command is applied to any EBGP peering session where the removal of private AS numbers is needed. Before the default prepend action occurs during the outbound route advertisement, the router checks the current AS Path attribute looking for private AS values. This check starts with the most recent AS value in the path and continues until a globally unique AS is located. During this check, all private AS values are removed from the attribute. The router then adds its local AS value to the path and advertises the route to the EBGP peer.



Private AS values buried within the AS Path attribute are not affected by the `remove-private` command. For example, AS 65333 is not removed from the AS Path of 64888 64999 1111 65333 2222.

FIGURE 5.2 Removing private AS numbers



The Merlot router in Figure 5.2 applies the `remove-private` option to its EBGP peering session with Zinfandel:

```
[edit protocols bgp]
user@Merlot# show group external-peers
type external;
remove-private;
peer-as 2222;
neighbor 10.222.3.1;
```

When we check the routing table on the Zinfandel router, we see that the AS values of 65010 and 65020 are no longer visible within AS 2222:

```
user@Zinfandel> show route protocol bgp terse

inet.0: 15 destinations, 15 routes (15 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination          P Prf  Metric 1  Metric 2  Next hop          AS path
* 172.16.1.0/24        B 170      100           >10.222.3.2      1111 I
* 172.20.4.0/24        B 170      100           >10.222.3.2      1111 I
```

Migrating to a New Global AS Number

A second method for altering the information in the AS Path attribute is best explained in the context of migrating from one globally assigned AS to another. At first, this might not seem like a good reason for altering the path information. After all, the only place you need to make a configuration change is within the `[edit routing-options]` hierarchy. This simple change can be easily accomplished in a maintenance window. Of course, any administrator of a large network (hundreds of routers) knows this type of task is easier said than done. Even if you could swap all of your router's configurations in a few hours, this only alters one side of the peering relationship. Each of your customers and peers also needs to update their configurations to reflect the new AS number—which is a much harder task. To assist with making this transition, the JUNOS software allows you to form a BGP peering session using an AS number other than the value configured within the `routing-options` hierarchy. You accomplish this task by using the `local-as` option within your configuration.

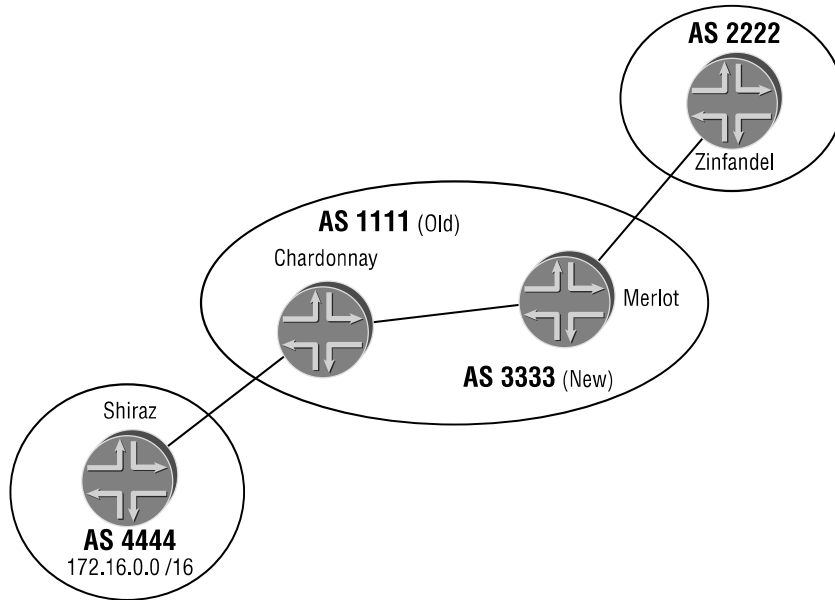
Figure 5.3 shows the Merlot and Chardonnay routers using AS 1111 to establish peering connections with AS 2222 and AS 4444. Routes in the address range of 172.16.0.0/16 are advertised by the Shiraz router in AS 4444. The Zinfandel router in AS 2222 currently sees these routes as

```
user@Zinfandel> show route protocol bgp terse

inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination          P Prf  Metric 1  Metric 2  Next hop          AS path
* 172.16.1.0/24        B 170      100           >10.222.3.2      1111 4444 I
```


FIGURE 5.3 Migrating to a new AS number



The AS 1111 network now merges with another network, which is currently using AS 3333 as its AS number. The newly combined entity decides to use AS 3333 as the AS value on all of its routers and configures this within the routing-options hierarchy. Additionally, the configuration within AS 2222 is altered to reflect the new AS number. This allows the peering session between Merlot and Zinfandel to reestablish. The session between Shiraz and Chardonnay is currently not operational:

```
user@Merlot> show bgp summary
Groups: 2 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths  Suppressed  History Damp State  Pending
inet.0         0          0          0           0         0         0         0
Peer           AS         InPkt     OutPkt     OutQ      Flaps  Last Up/Dwn  State
192.168.32.1   3333      8         9          0         0       3:11 0/0/0
10.222.3.1     2222      5         6          0         0       1:52 0/0/0
```

```
user@Chardonnay> show bgp summary
Groups: 2 Peers: 2 Down peers: 1
Table          Tot Paths  Act Paths  Suppressed  History Damp State  Pending
inet.0         0          0          0           0         0         0         0
Peer           AS         InPkt     OutPkt     OutQ      Flaps  Last Up/Dwn  State
192.168.40.1   3333     10         14          0         0       4:37 0/0/0
10.222.44.1     4444      1          2          0         0       5:29 Active
```

Since the administrators of the Shiraz router in AS 4444 are not able to update their side of the peering session, we can reestablish the session by adding some configuration to Chardonnay. We apply the `local-as` command to the peering session with Shiraz, which uses AS 1111 as our AS number during the session setup. After committing our configuration, the session is once again operational:

```
[edit protocols bgp]
user@Chardonnay# show group external-peers
type external;
peer-as 4444;
local-as 1111;
neighbor 10.222.44.1;
```

```
user@Chardonnay> show bgp summary
Groups: 2 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed  History Damp State  Pending
inet.0          3           3           0           0         0         0
Peer           AS        InPkt    OutPkt    OutQ   Flaps  Last Up/Dwn State
192.168.40.1   3333      20       25       0      0     10:00 0/0/0
10.222.44.1    4444      5         6         0      0     1:08 3/3/0
```

The routes advertised by the Shiraz router are once again visible on Zinfandel in AS 2222:

```
user@Zinfandel> show route protocol bgp terse

inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination      P Prf  Metric 1  Metric 2  Next hop          AS path
* 172.16.1.0/24    B 170    100          >10.222.3.2      3333 1111 4444 I
```

A closer examination of the AS Path in Zinfandel's output reveals some interesting information; both AS 1111 and AS 3333 appear in the path. The addition of the peer-specific AS number in the path output is the default behavior of the `local-as` command. To users in the Internet, it appears as if AS 1111 is still a viable network. Additionally, Chardonnay performs the function of an EBGp peer for its IBGP session with Merlot—it updates the path. The routes are visible on Merlot as:

```
user@Merlot> show route protocol bgp terse

inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination      P Prf  Metric 1  Metric 2  Next hop          AS path
* 172.16.1.0/24    B 170    100          100 >10.222.45.2      1111 4444 I
```

Should you wish to completely remove the old AS information from the AS Path attribute, the JUNOS software provides the *private* option to the `local-as` command. This allows Chardonnay to keep knowledge of AS 1111 to itself and not update the AS Path attribute before advertising the routes to Merlot:

```
[edit protocols bgp]
user@Chardonnay# show group external-peers
type external;
peer-as 4444;
local-as 1111 private;
neighbor 10.222.44.1;
```

We can now verify that AS 1111 no longer appears in the path information on both Merlot and Zinfandel:

```
user@Merlot> show route protocol bgp terse


inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination      P Prf  Metric 1  Metric 2  Next hop      AS path
* 172.16.1.0/24    B 170      100      100 >10.222.45.2  4444 I
```

```
user@Zinfandel> show route protocol bgp terse

inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination      P Prf  Metric 1  Metric 2  Next hop      AS path
* 172.16.1.0/24    B 170      100                >10.222.3.2    3333 4444 I
```

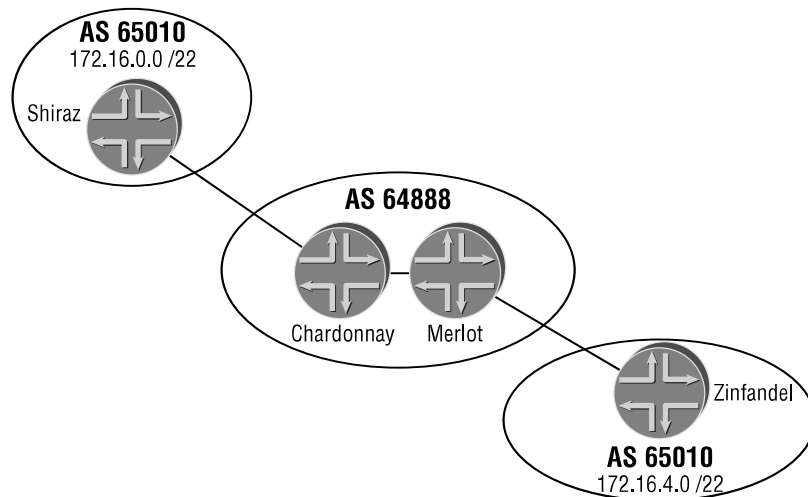


The `local-as` command should only be used in this specific circumstance. Its configuration within a normal BGP configuration could cause unexpected results in your network.

Providing Backbone Service for BGP Peers

The final two methods for altering AS Path information within the JUNOS software are explained through the use of an unusual network configuration.

The sample network in Figure 5.4 provides the backdrop for our discussion. The Chardonnay and Merlot routers in AS 64888 are providing a backbone service to the BGP routers in AS 65010. Both the Shiraz and Zinfandel routers are advertising a portion of the 172.16.0.0/16 address space assigned to that AS, but no internal network connectivity is provided between them. Instead, AS 64888 is providing the backbone and network reachability for the different sections of AS 65010.

FIGURE 5.4 Backbone service for BGP peers

This configuration is similar in nature to a Layer 3 VPN, which we discuss in Chapter 9, “Layer 2 and Layer 3 Virtual Private Networks.”

The problem with our configuration is the successful advertisement of the 172.16.0.0/16 routes to both portions of AS 65010. Let’s begin by examining the routes advertised by the Shiraz router. These routes are received by its EBGP peer of Chardonnay and are transmitted across AS 64888 to Merlot:

```
user@Merlot> show route protocol bgp terse
```

```
inet.0: 19 destinations, 19 routes (19 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

A	Destination	P	Prf	Metric 1	Metric 2	Next hop	AS path
*	172.16.1.0/24	B	170	100	100	>10.222.45.2	65010 I
*	172.16.5.0/24	B	170	100	0	>10.222.3.1	65010 I

These active BGP routes are then readvertised by Merlot to Zinfandel. While Merlot claims to have sent the routes, however, the Zinfandel router doesn’t appear to receive them. They don’t even appear as hidden routes on Zinfandel:

```
user@Merlot> show route advertising-protocol bgp 10.222.3.1 172.16.0/22
```

```
inet.0: 19 destinations, 19 routes (19 active, 0 holddown, 0 hidden)
```

Prefix	Nexthop	MED	Lclpref	AS path
* 172.16.1.0/24	Self			65010 I

```
user@Zinfandel> show route receive-protocol bgp 10.222.3.2
```

```
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
```

```
user@Zinfandel>
```

Some of you may already see the issue here, but let's explain what's occurring. An examination of the AS Path attribute in the output from Merlot states that the current path is 65010. The only AS Path information not seen on Merlot is the default AS prepend accomplished as the packets leave the router. This means that Zinfandel receives a path of 64888 65010. When a BGP router receives an announced route, it first examines the path to determine if a loop exists. In our case, Zinfandel sees its own AS in the path of the received route. Believing this to be a routing loop, the routes are dropped and are not visible by any JUNOS software show command. Administratively, we know that this is not a routing loop and would like Zinfandel to receive the advertised routes. This desire is accomplished through the use of the `as-override` command.

This configuration option is applied to an EBGP peering session and works in a similar fashion to the `remove-private` command. The main difference between the two options is that the AS number of the EBGP peer is located in the path and not all private AS numbers. When the local router finds the peer's AS, it replaces that AS with its own local AS number. This allows the receiving router to not see its own AS in the path and therefore accept the route. We now configure the Merlot router with the `as-override` command:

```
[edit protocols bgp]
user@Merlot# show group external-peers
type external;
peer-as 65010;
as-override;
neighbor 10.222.3.1;
```

The routes are now received by the Zinfandel router with an AS Path of 64888 64888:

```
user@Zinfandel> show route receive-protocol bgp 10.222.3.2
```

```
inet.0: 15 destinations, 18 routes (15 active, 0 holddown, 3 hidden)
```

Prefix	Nexthop	MED	Lclpref	AS path
* 172.16.1.0/24	10.222.3.2			<u>64888 64888 I</u>

Let's now focus on the 172.16.4.0/22 routes advertised by Zinfandel. We see that they are advertised by Chardonnay to Shiraz, which claims not to have received them:

```
user@Chardonnay> show route advertising-protocol bgp 10.222.44.1 172.16.4/22
```

```
inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.16.5.0/24         Self              65010  I
```

```
user@Shiraz> show route receive-protocol bgp 10.222.44.2
```

```
inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
```

```
user@Shiraz>
```

We have the same issue on this side of the network—an AS Path loop. This provides us with the opportunity to use the final JUNOS software configuration option for altering the AS Path information. This involves the *loops* option as part of the `autonomous-system` command. When you use this configuration option, you are allowing the local AS number to appear in the path more than once. In fact, the AS number can appear as many as 10 times, though just twice is enough in our case. We configure this on the receiving router of Shiraz and check our results:

```
[edit]
user@Shiraz# show routing-options
static {
    route 172.16.1.0/24 reject;
}
autonomous-system 65010 loops 2;
```

```
user@Shiraz> show route receive-protocol bgp 10.222.44.2
```

```
inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
```

```
user@Shiraz>
```

We haven't received any routes from Chardonnay, so it might appear that we have a problem with our configuration—but in fact we don't. While the JUNOS software default behavior for BGP peering sessions is effectively a soft inbound reconfiguration, this applies only to routes that are present in the Adjacency-RIB-In table. Since these routes were seen as a routing loop, they were immediately discarded and not retained in that table. This simply means that we need to manually ask Chardonnay to send the routes again:

```
user@Shiraz> clear bgp neighbor soft-inbound
```

```
user@Shiraz> show route protocol bgp terse
```

```
inet.0: 13 destinations, 16 routes (13 active, 0 holddown, 3 hidden)
+ = Active Route, - = Last Active, * = Both
```

A Destination	P Prf	Metric 1	Metric 2	Next hop	AS path
* 172.16.5.0/24	B 170	100		>10.222.44.2	<u>64888 65010 I</u>

The routes now appear in the routing table of Shiraz with an AS Path of 64888 65010.



Extreme care should be taken when you use the `as-override` and `loops` commands. Improper use could result in routing loops in your network.

Modifying the AS Path: Routing Policy

Most network administrators alter the AS Path attribute by using a routing policy to add information to the path. This artificially increases the path length, potentially making the advertised route less desirable to receiving routers. The longer path lengths could therefore affect inbound user traffic flows into the local AS. The JUNOS software provides the ability to prepend your local AS or a customer AS to the path. Let's see how these two options work.

Prepending Your Own AS Number

The most common method of adding information to the AS Path attribute is including more than one instance of your own AS number before advertising the route. This is accomplished with a routing policy applied as routes are sent to an EBGP peer. The policy action of `as-path-prepend value` adds the supplied AS numbers to the AS Path attribute after the default prepend occurs.

FIGURE 5.5 AS Path prepend example

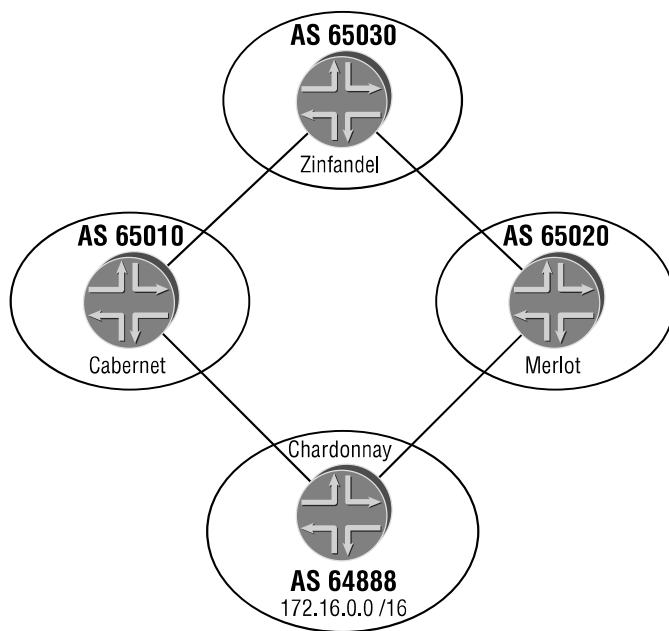


Figure 5.5 provides an example of how prepending your local AS onto the path affects the route decisions of other networks. The Chardonnay router in AS 64888 has peering sessions with both the Cabernet and Merlot routers. Each of these routers, in turn, has a peering session with the Zinfandel router in AS 65030. The administrators of AS 64888 would like user traffic from AS 65030 to transit through Cabernet on its way to the 172.16.0.0/16 address space. The current BGP routes on Zinfandel include

```
user@Zinfandel> show route protocol bgp

inet.0: 10 destinations, 11 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.0.0/16      *[BGP/170] 00:01:04, localpref 100
                  AS path: 65020 64888 I
                  > to 10.222.3.2 via at-0/1/1.0
                  [BGP/170] 00:01:00, localpref 100
                  AS path: 65010 64888 I
                  > to 10.222.61.2 via so-0/3/0.0
```

Zinfandel has received the 172.16.0.0/16 route from both AS 65020 and AS 65010. Each of the route announcements has a path length of 2, which prevents Zinfandel from using this attribute to select the best BGP route. To accomplish our administrative goal, we configure the **prepend-to-aggregate** policy on the Chardonnay router. This policy is applied to AS 65020 and the Merlot router:

```
[edit policy-options]
user@Chardonnay# show policy-statement prepend-to-aggregate
term prepend {
  from protocol aggregate;
  then {
    as-path-prepend "64888 64888";
    accept;
  }
}

[edit]
user@Chardonnay# show protocols bgp
group external-peers {
  type external;
  export adv-routes;
  neighbor 10.222.45.1 {
    export prepend-to-aggregate;
    peer-as 65020;
```



```

    }
    neighbor 10.222.6.1 {
        peer-as 65010;
    }
}

```



While the JUNOS software allows you to enter and advertise any AS value using this type of policy, it is considered a best practice to only prepend your own AS number.

After committing our configuration, we can check to see what information Chardonnay thinks it is sending to its peers:

```
user@Chardonnay> show route advertising-protocol bgp 10.222.6.1
```

```
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc|pref  AS path
* 172.16.0.0/16         Self              0

```

```
user@Chardonnay> show route advertising-protocol bgp 10.222.45.1
```

```
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc|pref  AS path
* 172.16.0.0/16         Self              0          64888 64888 [64888] I

```

It would appear that we've succeeded in our goal. The 172.16.0.0/16 route is advertised to both peers, but the version sent to Merlot has AS 64888 prepended twice onto the path. The [64888] notation in the router output reminds us that the default prepend action is still occurring as the routes are advertised. In essence, we've included three instances of our AS with that route advertisement. Similar results are seen from the perspective of the Zinfandel router in AS 65030:

```
user@Zinfandel> show route protocol bgp
```

```
inet.0: 10 destinations, 11 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

```

```
172.16.0.0/16      *[BGP/170] 00:06:15, localpref 100
                   AS path: 65010 64888 I
                   > to 10.222.61.2 via so-0/3/0.0
                   [BGP/170] 00:06:27, localpref 100
                   AS path: 65020 64888 64888 64888 I
                   > to 10.222.3.2 via at-0/1/1.0

```

Prepending a Customer AS Number

Certain network topologies lend themselves to the prepending of a customer AS number to the path. This often occurs when a service provider has a customer attached via BGP across multiple peering points. The routes advertised by that customer into the provider's network are not prepended to allow for potential load balancing of traffic into the customer network. However, the customer would like the majority of its traffic from the Internet to arrive via a particular upstream AS. Figure 5.6 provides such an example.

The service provider network (AS 65010) consists of the Merlot, Chardonnay, and Sangiovese routers. It has two connections to the customer AS of 64888. In addition, two upstream peers are connected to the provider network: Cabernet in AS 65020 and Zinfandel in AS 65030. The customer routes are within the address space of 172.16.0.0/22 and are visible on the Merlot router:

```
user@Merlot> show route protocol bgp terse
```

```
inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

A Destination	P Prf	Metric 1	Metric 2	Next hop	AS path
* 172.16.1.0/24	B 170	100	0	>10.222.45.2	64888 I

Merlot then advertises the routes to its upstream peers of Cabernet and Zinfandel:

```
user@Cabernet> show route protocol bgp terse
```

```
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

A Destination	P Prf	Metric 1	Metric 2	Next hop	AS path
* 172.16.1.0/24	B 170	100		>10.222.1.2	65010 64888 I

```
user@Zinfandel> show route protocol bgp terse
```

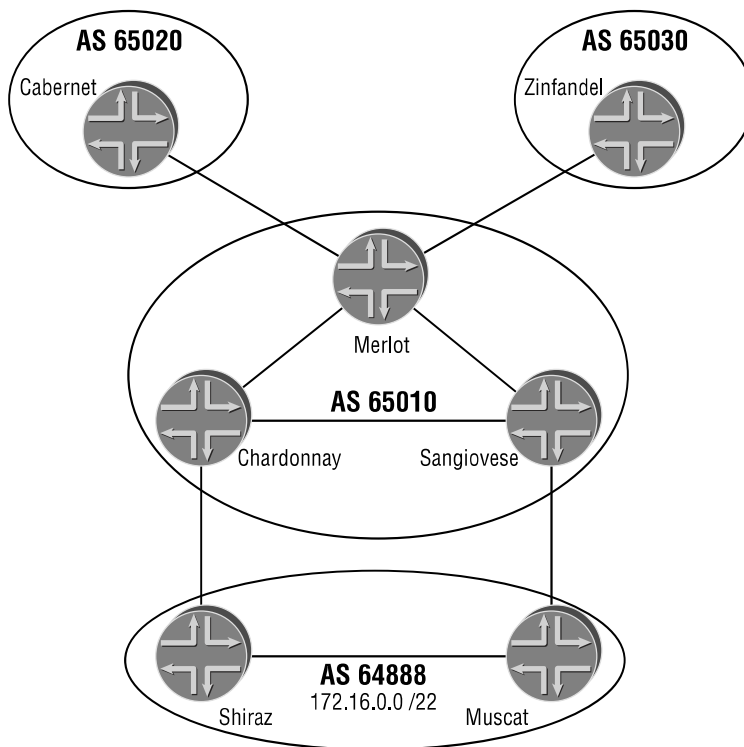
```
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

A Destination	P Prf	Metric 1	Metric 2	Next hop	AS path
* 172.16.1.0/24	B 170	100		>10.222.3.2	65010 64888 I

The customer in AS 64888 would like traffic forwarded from the Internet to traverse the Zinfandel router in AS 65030. They have requested that the provider enforce this requirement by prepending the AS Path length. One option available to the administrators of AS 65010 is to prepend the AS of their customer using the `as-path-expand last-as count value` policy

action. When applied to an EBGP peering session, this policy action examines the AS Path attribute before the default prepend action takes place. The first AS number located, which is also the last value added to the path, is prepended as specified in the *value* area (up to 32 times). The router then performs its default prepend action and advertises the routes.

FIGURE 5.6 Customer AS prepend example



For our example, we create the ***prepend-customer-as*** policy on the Merlot router and apply it to the peering session with Cabernet. The policy locates any routes with AS 64888 in the path and prepends it three additional times onto the path:

```
[edit]
user@Merlot# show policy-options
policy-statement prepend-customer-as {
  term prepend {
    from as-path AS64888;
    then {
      as-path-expand last-as count 3;
    }
  }
}
as-path AS64888 ".* 64888 .*";
```

```
[edit]
user@Merlot# show protocols bgp group external-peers
type external;
neighbor 10.222.3.1 {
    peer-as 65030;
}
neighbor 10.222.1.1 {
    export prepend-customer-as;
    peer-as 65020;
}
```

After committing our configuration, we can check what routes Merlot is sending to Cabernet and Zinfandel:

```
user@Merlot> show route advertising-protocol bgp 10.222.1.1

inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED  Lc|pref  AS path
* 172.16.1.0/24        Self              64888 64888 64888 64888 I
```

```
user@Merlot> show route advertising-protocol bgp 10.222.3.1

inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED  Lc|pref  AS path
* 172.16.1.0/24        Self              64888 I
```

The routes advertised by Merlot are adhering to our administrative policy. A closer examination of the route output shows that the default prepend action is not represented, but remember that it still occurs. We can check this by examining the routing table on the Zinfandel router:

```
user@Zinfandel> show route protocol bgp

inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 00:30:21, localpref 100
                   AS path: 65010 64888 I
                   > to 10.222.3.2 via at-0/1/1.0
```

Multiple Exit Discriminator

The Multiple Exit Discriminator (MED) attribute is optional and doesn't need to be included in each routing update. Its use is valuable when two ASs have multiple physical links between

themselves. A MED value is set by the administrators of one AS for all routes advertised into its peer. This allows the peer AS to make routing decisions based on the advertised MEDs and potentially load-balance traffic across the multiple physical links. We first discuss how the JUNOS software evaluates the MED attribute and some ways to alter that selection process. We then see how to modify the attribute value using both configuration options as well as routing policy actions.

MED Selection Mechanisms

By default, a BGP router only compares the MED values of two path advertisements when they arrive from the same neighboring AS. The JUNOS software automatically groups advertisements from the same AS together to compare their MED values. The best value from each group is then compared with each other using other route attributes. This evaluation process is known as *deterministic MED*. To better appreciate how the deterministic MED process works, suppose that the 192.168.1.0/24 route is received from three different peers. The possible path advertisements are

- Path 1—via EBGp; AS Path of 65010; MED of 200
- Path 2—via IBGP; AS Path of 65020; MED of 150; IGP cost of 5
- Path 3—via IBGP; AS Path of 65010; MED of 100; IGP cost of 10

Using the deterministic MED scheme, the router groups paths 1 and 3 together since they were both received from AS 65010. Between these two advertisements, path 3 has a better (lower) MED value. The router then eliminates path 1 from the selection process and evaluates path 3 against path 2. These two paths were received from different AS networks, so the router doesn't evaluate the MEDs. Instead, it uses other route attributes to select the active path—in this case, the IGP metric cost to the IBGP peer. The IGP cost of path 2 is better (lower) than the IGP cost of path 3. This allows the router to install path 2 as the active path for the 192.168.1.0/24 route.

You have two options for altering this default method of operation within the JUNOS software. Let's discuss each of them separately.

Always Comparing MED Values

The first method for altering the MED behavior allows the router to always use the MED values to compare routes. This occurs regardless of the neighboring AS the path advertisement was received from. You enable this feature by using the `path-selection always-compare-med` command at the BGP global hierarchy level. Let's assume that the same three paths are received for the 192.168.1.0/24 route:

- Path 1—via EBGp; AS Path of 65010; MED of 200
- Path 2—via IBGP; AS Path of 65020; MED of 150; IGP cost of 5
- Path 3—via IBGP; AS Path of 65010; MED of 100; IGP cost of 10

After enabling the `always-compare-med` function, the router begins to group all received advertisements into a single group. The MED values of each route are then compared against each other. Because a lower MED value is preferred, the router chooses path 3 as the active path to the destination. This is installed in the local routing table and user data packets are forwarded using the attributes of this path.



Exercise care when using this configuration option. Not all network operators agree on what a good MED value is. One AS might use 50 as a good value, while another might choose 5. Worse yet, some operators might not set a MED at all, which is interpreted as a 0 value.

Emulating the Cisco Systems Default Behavior

The second MED evaluation method allows you to emulate the default behavior of a Cisco Systems router. This operational mode evaluates routes in the order that they are received and doesn't group them according to their neighboring AS. In essence, this is the opposite of the deterministic MED process. This feature is also configured at the global BGP hierarchy level with the `path-selection cisco-non-deterministic` command. To accurately determine the effect of this feature, let's use the same three path advertisements for the 192.168.1.0 /24 route:

- Path 1—via EBGp; AS Path of 65010; MED of 200
- Path 2—via IBGP; AS Path of 65020; MED of 150; IGP cost of 5
- Path 3—via IBGP; AS Path of 65010; MED of 100; IGP cost of 10

These advertisements are received in quick succession, within a second, in the order listed. Path 3 was received most recently so the router compares it against path 2, the next most recent advertisement. The cost to the IBGP peer is better for path 2, so the router eliminates path 3 from contention. When comparing paths 1 and 2 together, the router prefers path 1 since it was received from an EBGp peer. This allows the router to install path 1 as the active path for the route.



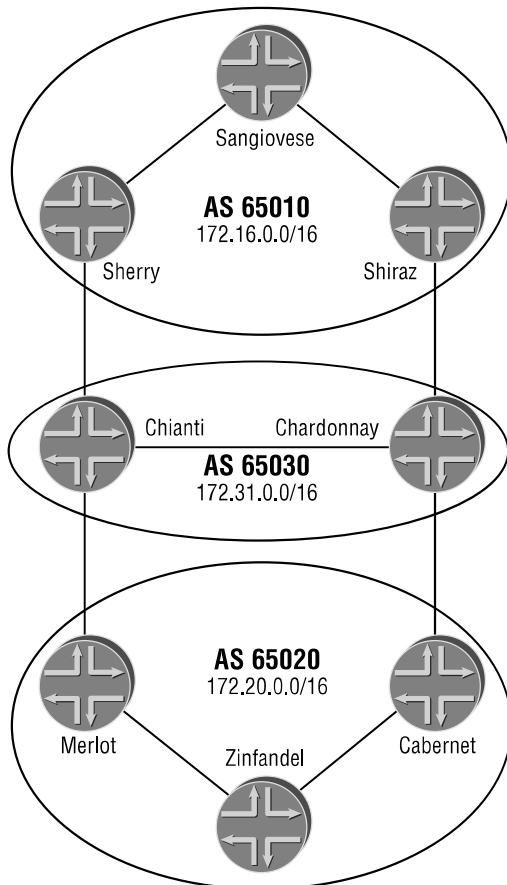
We do not recommend using this configuration option in your network. It is provided solely for interoperability to allow all routers in the network to make consistent route selections.

Altering the MED: Configuration Statements

Routes advertised by a Juniper Networks router may be assigned a MED value using one of several configuration options. You have the ability to use each option at either the global, group, or peer level of the configuration. When you use these options, the router applies the configured MED value to all routes advertised to a peer. In addition to setting a manual value, you may associate the IGP metric used within the AS to the advertised BGP route. Let's see how each of these categories works.

Manually Setting the MED

Network administrators might wish to manually set a MED value on all advertised routes when their AS is connected to only one BGP peer AS.

FIGURE 5.7 MED attribute sample network


We see that AS 65010 in Figure 5.7 contains the Sherry, Sangiovese, and Shiraz routers. The address range of 172.16.0.0/16 is assigned to that network, and the Sangiovese router is advertising routes to its IBGP peers:

```
user@Sangiovese> show route advertising-protocol bgp 192.168.16.1
```

```
inet.0: 15 destinations, 18 routes (15 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED      Lc1pref  AS path
* 172.16.1.0/24         Self             0         100      I
```

The administrators of AS 65010 would like their inbound user traffic to arrive on the Sherry router. Therefore, the routes advertised by Sherry should have a lower MED than the routes advertised by Shiraz. We accomplish this administrative goal by using the `metric-out` command at the BGP peer group level for the EBGP peers. This option allows the routes to receive

a static MED value between 0 and 4,294,967,295. The configuration of the Sherry router now appears as so:

```
[edit protocols bgp]
user@Sherry# show group external-peers
type external;
metric-out 20;
peer-as 65030;
neighbor 10.222.29.2;
```

The Sherry router is now advertising the routes with a MED of 20, where they are received by the Chianti router in AS 65030:

```
user@Sherry> show route advertising-protocol bgp 10.222.29.2 172.16/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.16.1.0/24        Self             20                    I
```

```
user@Chianti> show route receive-protocol bgp 10.222.29.1
```

```
inet.0: 18 destinations, 24 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
172.16.1.0/24        10.222.29.1     20                    65010 I
```

A similar configuration is applied to the Shiraz router. In this instance, however, a MED value of 50 is applied to the advertised routes:

```
[edit protocols bgp]
user@Shiraz# show group external-peers
type external;
metric-out 50;
peer-as 65030;
neighbor 10.222.44.2;
```

```
user@Shiraz> show route advertising-protocol bgp 10.222.44.2 172.16/16
```

```
inet.0: 16 destinations, 19 routes (16 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.16.1.0/24        Self             50                    I
```


The advertised MED values allow the routers in AS 65030 to forward traffic to AS 65010 through the Sherry router:

```
user@Chianti> show route 172.16/16

inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 00:04:29, MED_20, localpref 100
                  AS path: 65010 I
                  > to 10.222.29.1 via ge-0/3/0.0
```

The router output from the Chianti router shows the active BGP route using the ge-0/3/0.0 interface connected to Sherry in AS 65010. The Chardonnay router, on the other hand, is using the routes advertised by Chianti (192.168.20.1) over its so-0/3/0.600 interface. This occurs even as Chardonnay receives the same routes directly from Shiraz on its fe-0/0/0.0 interface:

```
user@Chardonnay> show route 172.16/16

inet.0: 18 destinations, 24 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 00:01:59, MED_20, localpref 100, from 192.168.20.1
                  AS path: 65010 I
                  > to 10.222.100.1 via so-0/3/0.600
                  [BGP/170] 00:01:59, MED 50, localpref 100
                  AS path: 65010 I
                  > to 10.222.44.1 via fe-0/0/0.0
```

Associating the MED to the IGP Metric

Some network administrators correlate their internal IGP metrics to a single standard. This might be a representation of the various link bandwidths in the network, or it might represent the physical distance between the network devices (fiber route miles). Regardless of the details, the correlation allows internal routing to follow the shortest path according to the administrative setup. Should your IGP be configured in such a manner, it would be ideal to have a way to communicate this knowledge to BGP routers in a neighboring AS. This would allow those peers to forward traffic into your AS with the knowledge that it would be using the shortest paths possible.

Referring back to Figure 5.7, we see that the Merlot, Zinfandel, and Cabernet routers in AS 65020 have been assigned the 172.20.0.0/16 address space. The administrators of AS 65020 would like to assign MED values to these routes that represent their internal IGP metrics. The JUNOS software provides two configuration options to assist AS 65020 in reaching its goal. These are the `metric-out igp` and `metric-out minimum-igp` configuration commands. While both options advertise the current IGP metric associated with the IBGP peer that advertised the

route, they perform this function in slightly different manners. The `igp` feature directly tracks the IGP cost to the IBGP peer. When the IGP cost goes down, so does the advertised MED value. Conversely, when the IGP cost goes up the MED value goes up as well.

The Merlot router has a current IGP cost of 20 to Zinfandel, who is advertising the 172.20.0.0/16 routes to BGP:

```
user@Merlot> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:05:50, metric 20
                  > to 10.222.3.1 via at-0/2/0.0
```

```
user@Merlot> show route 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
172.20.1.0/24     *[BGP/170] 23:45:25, MED 0, localpref 100, from 192.168.56.1
                  AS path: I
                  > to 10.222.3.1 via at-0/2/0.0
```

Merlot currently advertises these routes to AS 65030 with no MED value attached:

```
user@Merlot> show route advertising-protocol bgp 10.222.1.1 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lclpref  AS path
* 172.20.1.0/24         Self              0      0         I
```

The `metric-out igp` command is now applied to the EBGP peer group on the Merlot router. This allows the router to assign a MED of 20, the current IGP cost to 192.168.56.1, to the BGP routes:

```
[edit protocols bgp]
```

```
user@Merlot# show group external-peers
```

```
type external;
```

```
metric-out igp;
```

```
peer-as 65030;
```

```
neighbor 10.222.1.1;
```

```
user@Merlot> show route advertising-protocol bgp 10.222.1.1 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lclpref  AS path
* 172.20.1.0/24         Self             20                      I
```

When the IGP cost from Merlot to Zinfandel rises to 50, the advertised MED value to Chianti also changes to 50:

```
user@Merlot> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:00:05, metric 50
                  > to 10.222.3.1 via at-0/2/0.0
```

```
user@Merlot> show route advertising-protocol bgp 10.222.1.1 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lclpref  AS path
* 172.20.1.0/24         Self             50                      I
```

As we would expect, a reduction in the IGP cost to 10 also changes the advertised MED value:

```
user@Merlot> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:00:03, metric 10
                  > to 10.222.3.1 via at-0/2/0.0
```

```
user@Merlot> show route advertising-protocol bgp 10.222.1.1 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lclpref  AS path
* 172.20.1.0/24         Self             10                      I
```

On the Cabernet router across the AS, we configure the `metric-out minimum-igp` option. As the name suggests, the advertised MED value only changes when the IGP cost to the IBGP peer goes down. A rise in the IGP cost doesn't affect the MED values at all. The router monitors and remembers the lowest IGP cost until the routing process is restarted. The current cost from Cabernet to Zinfandel is 30, which matches the advertised MED values to Chardonnay in AS 65030:

```
[edit protocols bgp]
```

```
user@Cabernet# show group external-peers
```

344 Chapter 5 ■ Advanced Border Gateway Protocol (BGP)

```
type external;  
metric-out minimum-igp;  
peer-as 65030;  
neighbor 10.222.6.2;
```

```
user@Cabernet> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)  
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:02:18, metric 30  
                  > to 10.222.61.1 via so-0/3/0.0
```

```
user@Cabernet> show route advertising-protocol bgp 10.222.6.2 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)  
Prefix           Nexthop           MED    Lc1pref    AS path  
* 172.20.1.0/24   Self              30          I
```

The advertised MED value decreases to 20 when the IGP cost also decreases:

```
user@Cabernet> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)  
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:00:04, metric 20  
                  > to 10.222.61.1 via so-0/3/0.0
```

```
user@Cabernet> show route advertising-protocol bgp 10.222.6.2 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)  
Prefix           Nexthop           MED    Lc1pref    AS path  
* 172.20.1.0/24   Self              20          I
```

When the IGP cost to Zinfandel rises to 50, however, the advertised MED value remains at 20:

```
user@Cabernet> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)  
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:00:04, metric 50
                  > to 10.222.61.1 via so-0/3/0.0
```

```
user@Cabernet> show route advertising-protocol bgp 10.222.6.2 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lclpref  AS path
* 172.20.1.0/24         Self             20     I
```

Altering the MED: Routing Policy

When you use the configuration options to set the MED value on advertised BGP routes, all possible routes are affected. In fact, this is a common theme in the JUNOS software BGP configuration. To apply the MED values only to specific routes, use a routing policy to locate those routes and then set your desired value. The options for setting the MED in a routing policy are identical to those available with configuration knobs. You can set the value manually or via a routing policy.

Manually Setting the MED

Using Figure 5.7 as a guide, the network administrators of AS 65010 would like to set the MED only for routes originating in their AS. A routing policy called **set-med** is configured that locates these routes using an AS Path regular expression and sets the advertised MED value to 10. The policy is currently configured as so:

```
user@Sherry> show configuration policy-options | find set-med
policy-statement set-med {
  term only-AS65010-routes {
    from as-path local-to-AS65010;
    then {
      metric 10;
    }
  }
}
as-path local-to-AS65010 "(");
```

We apply the policy to the EBGp peer group, and the 172.16.0.0 /16 routes are advertised with a MED value of 10:

```
[edit protocols bgp]
user@Sherry# show group external-peers
type external;
export set-med;
peer-as 65030;
```

```
neighbor 10.222.29.2;
```

```
user@Sherry> show route advertising-protocol bgp 10.222.29.2 172.16/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.16.1.0/24         Self             10                      I
```

A similar configuration on the Shiraz router in AS 65010 advertises the 172.16.0.0/16 routes with a MED value of 40. This ensures that all inbound traffic from AS 65030 uses the Sherry-Chianti EBGp peering session:

```
[edit policy-options]
user@Shiraz# show | find set-med
policy-statement set-med {
  term only-AS65010-routes {
    from as-path local-to-AS65010;
    then {
      metric 40;
    }
  }
}
as-path local-to-AS65010 "(");
```

```
[edit protocols bgp]
user@Shiraz# show group external-peers
type external;
export set-med;
peer-as 65030;
neighbor 10.222.44.2;
```

```
user@Shiraz> show route advertising-protocol bgp 10.222.44.2 172.16/16
```

```
inet.0: 16 destinations, 19 routes (16 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.16.1.0/24         Self             40                      I
```

Associating the MED to the IGP Metric

The administrators of AS 65020 in Figure 5.7 would also like to advertise MED values for just their assigned address space of 172.20.0.0 /16 and not all BGP routes. Appropriate routing policies are

configured on the Merlot and Cabernet routers using the `igp` and `minimum-igp` MED options. The `set-med` policy on the Merlot router appears as so:

```
user@Merlot> show configuration policy-options | find set-med
policy-statement set-med {
  term only-AS65020-routes {
    from {
      route-filter 172.20.0.0/16 orlonger;
    }
    then {
      metric igp;
    }
  }
}
```

As before, the advertised MED value directly tracks the IGP cost to the Zinfandel router:

```
user@Merlot> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:01:53, metric 15
                  > to 10.222.3.1 via at-0/2/0.0
```

```
user@Merlot> show route advertising-protocol bgp 10.222.1.1 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
Prefix           Nexthop          MED    Lc1pref  AS path
* 172.20.1.0/24   Self             15     I
```

A lower IGP results in a lower MED value:

```
user@Merlot> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:00:02, metric 10
                  > to 10.222.3.1 via at-0/2/0.0
```

```
user@Merlot> show route advertising-protocol bgp 10.222.1.1 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.20.1.0/24        Self             10                    I
```

As expected, a higher MED value results from an IGP cost increase:

```
user@Merlot> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:00:03, metric 20
                  > to 10.222.3.1 via at-0/2/0.0
```

```
user@Merlot> show route advertising-protocol bgp 10.222.1.1 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.20.1.0/24        Self             20                    I
```

The Cabernet router in AS 65030 also has a **set-med** policy configured, but this version uses the `minimum-igp` action to set the MED value:

```
[edit policy-options]
user@Cabernet# show | find set-med
policy-statement set-med {
  term only-AS65020-routes {
    from {
      route-filter 172.20.0.0/16 orlonger;
    }
    then {
      metric minimum-igp;
    }
  }
}
```

```
[edit protocols bgp]
user@Cabernet# show group external-peers
type external;
export set-med;
peer-as 65030;
neighbor 10.222.6.2;
```


The MED values advertised to Chardonnay in AS 65030 now represent the smallest known IGP cost to the Zinfandel router:

```
user@Cabernet> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:02:39, metric 15
                  > to 10.222.61.1 via so-0/3/0.0
```

```
user@Cabernet> show route advertising-protocol bgp 10.222.6.2 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop                MED    Lc1pref    AS path
* 172.20.1.0/24         Self                    15     I           I
```

```
user@Cabernet> show route 192.168.56.1
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
192.168.56.1/32    *[IS-IS/18] 00:00:02, metric 30
                  > to 10.222.61.1 via so-0/3/0.0
```

```
user@Cabernet> show route advertising-protocol bgp 10.222.6.2 172.20/16
```

```
inet.0: 18 destinations, 21 routes (18 active, 0 holddown, 0 hidden)
  Prefix                Nexthop                MED    Lc1pref    AS path
* 172.20.1.0/24         Self                    15     I           I
```

Local Preference

The Local Preference attribute is the first value compared between two BGP routes in the route selection process. It is often used to set the exit point out of the local AS for a particular route. The JUNOS software allows you to modify this attribute with a routing policy action as well as a configuration option. As you might expect, the configuration option affects all advertised BGP routes while the routing policy allows you to be more selective.

The Chianti and Chardonnay routers in Figure 5.7 are located in AS 65030 and are advertising the 172.31.0.0/16 aggregate route into AS 65010:

```
user@Chianti> show route advertising-protocol bgp 10.222.29.1 172.31/16
```

```
inet.0: 20 destinations, 24 routes (20 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.31.0.0/16        Self              I
```

```
user@Chardonnay> show route advertising-protocol bgp 10.222.44.1 172.31/16
```

```
inet.0: 20 destinations, 24 routes (20 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.31.0.0/16        Self              I
```

By default, all received EBGP routes are assigned a Local Preference value of 100. We see this on the Sangiovese router:

```
user@Sangiovese> show route 172.31/16
```

```
inet.0: 16 destinations, 20 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.31.0.0/16      *[BGP/170] 00:00:49, localpref 100, from 192.168.16.1
                   AS path: 65030 I
                   > to 10.222.28.1 via fe-0/0/0.0
                   [BGP/170] 00:00:06, localpref 100, from 192.168.36.1
                   AS path: 65030 I
                   > to 10.222.4.2 via fe-0/0/1.0
```

The AS 65010 administrators would like to affect the routing decision made by the Sangiovese router for forwarding data packets to the 172.31.0.0/16 route. The Shiraz router should be used as the exit point out of AS 65010. The first step in accomplishing this goal is reducing the Local Preference value advertised by the Sherry router to 50. We accomplish this by using the `local-preference` configuration command at the BGP neighbor level:

```
[edit protocols bgp group internal-peers]
user@Sherry# show
type internal;
local-address 192.168.16.1;
export nhs;
neighbor 192.168.24.1 {
  local-preference 50;
}
neighbor 192.168.36.1;
```

This change only alters the route as it's advertised to Sangiovese. The local version of the route is not affected, and its Local Preference remains at 100:

```
user@Sherry> show route 172.31/16
```

```
inet.0: 19 destinations, 23 routes (19 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
172.31.0.0/16      *[BGP/170] 00:30:47, localpref 100
                   AS path: 65030 I
                   > to 10.222.29.2 via ge-0/1/0.0
                   [BGP/170] 00:01:09, localpref 100, from 192.168.36.1
                   AS path: 65030 I
                   > to 10.222.28.2 via fe-0/0/0.0
```

```
user@Sherry> show route advertising-protocol bgp 192.168.24.1 172.31/16
```

```
inet.0: 19 destinations, 23 routes (19 active, 0 holddown, 0 hidden)
  Prefix          Nexthop          MED    Lclpref    AS path
* 172.31.0.0/16   Self             50      50         65030 I
```

On the opposite side of the AS, the Shiraz router uses a routing policy to alter the Local Preference to 150 for just the 172.31.0.0/16 route. All other routes sent to Sangiovese don't have the attribute value changed from the default of 100. The routing policy appears as so:

```
user@Shiraz> show configuration policy-options | find set-local-preference
policy-statement set-local-preference {
  term only-AS65030-routes {
    from {
      route-filter 172.31.0.0/16 exact;
    }
    then {
      local-preference 150;
      accept;
    }
  }
}
as-path local-to-AS65010 "()";
```

The policy is applied to the BGP neighbor level configuration so that the Sangiovese router is the only router affected. As we saw with the Sherry router, the local version of the 172.31.0.0/16 route maintains a Local Preference value of 100 while the advertised route has a value of 150:

```
[edit protocols bgp]
user@Shiraz# show group internal-peers
```

```

type internal;
local-address 192.168.36.1;
export nhs;
neighbor 192.168.16.1;
neighbor 192.168.24.1 {
    export [ nhs set-local-preference ];
}

```

```
user@Shiraz> show route 172.31/16
```

```
inet.0: 17 destinations, 21 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```

172.31.0.0/16      *[BGP/170] 00:44:33, localpref 100
                   AS path: 65030 I
                   > to 10.222.44.2 via fe-0/0/1.0
                   [BGP/170] 00:15:38, localpref 100, from 192.168.16.1
                   AS path: 65030 I
                   > to 10.222.4.1 via fe-0/0/0.0

```

```
user@Shiraz> show route advertising-protocol bgp 192.168.24.1 172.31/16
```

```
inet.0: 17 destinations, 21 routes (17 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED      Localpref   AS path
* 172.31.0.0/16        Self              150          150          65030 I
```



Don't forget to copy any applied group level policies to the neighbor level in this type of configuration. For example, the *nhs* policy is also applied to the 192.168.24.1 peer to ensure the BGP Next Hop of the route is reachable.

When we examine the routing table of the Sangiovese router, we see that the 172.31.0.0/16 route is received with different Local Preference values. The higher value from the Shiraz router (192.168.36.1) is preferred over the lower value advertised from the Sherry router (192.168.16.1):

```
user@Sangiovese> show route 172.31/16
```

```
inet.0: 16 destinations, 20 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
172.31.0.0/16      *[BGP/170] 00:06:18, localpref 150, from 192.168.36.1
```

```

AS path: 65030 I
> to 10.222.4.2 via fe-0/0/1.0
[BGP/170] 00:21:57, localpref 50, from 192.168.16.1
AS path: 65030 I
> to 10.222.28.1 via fe-0/0/0.0

```



The examples used in this section are effective for highlighting the operation of the configuration options available to you. They are not recommended as a best practice in your network. Generally speaking, the Local Preference attribute is assigned to all received EBGP routes using an inbound routing policy. This ensures that all routers in the network make consistent routing decisions.

IBGP Scaling Methods

When you ask a network engineer why you need a full mesh of IBGP peering sessions, you often times get a response similar to “Because an IBGP-learned route can’t be readvertised to another IBGP peer.” While this is certainly a valid response, it is also not a complete answer. The reason for preventing the readvertisement of IBGP routes and requiring the full mesh is to avoid routing loops within an AS. Recall that the AS Path attribute is the means by which BGP routers avoid loops. The path information is examined for the local AS number only when the route is received from an EBGP peer. Since the attribute is only modified across AS boundaries, this system works extremely well. Unfortunately, the fact that the attribute is only modified across AS boundaries leaves us with problems internally. As a quick example, suppose that routers A, B, and C are all in the same AS. Router A receives a route from an EBGP peer and sends the route to B, who installs it as the active route. The route is then sent to router C, who installs it locally and sends it back to router A. Should router A install the route, we’ve formed a loop within our AS. We couldn’t detect the loop since the AS Path attribute wasn’t modified during these advertisements. Therefore, the protocol designers decided that the only assurance of never forming a routing loop was to prevent an IBGP peer from advertising an IBGP-learned route within the AS. For route reachability, the IBGP peers are then fully meshed.

Full-mesh networks have inherent scalability issues. For protocols that utilize a neighbor-discovery mechanism, the issues surround database and routing table sizes as well as performance during a network outage. For a BGP network, one additional issue is the explicit configuration of each BGP peer. Let’s assume that an AS has five operational IBGP peers and needs to add a sixth. In addition to the new router configuring its five other peers, each of the current routers must update its configuration to include the sixth router. In addition, the network protocol state grows exponentially as more peers are added. In our six-router network, a total of 15 IBGP peering sessions must be maintained $(n \times (n-1)) \div 2$. Things only worsen as the number of IBGP peers grows. Imagine having to reconfigure 99 existing routers to add a new peer to the cloud. This 100-router IBGP full mesh also requires the maintenance of 4950 peering sessions.

We have two main methods for alleviating these issues and scaling an AS to thousands of routers—route reflection and confederations. Each of these approaches replaces the full mesh of IBGP peering sessions and ensures a loop-free BGP network. These common goals are achieved using different methods and procedures, which we discuss in some depth throughout this section.

Route Reflection

The approach taken in a *route reflection* network to solving the full-mesh problem is allowing an IBGP-learned route to be readvertised, or reflected, to an IBGP peer. This is allowed to occur only on special routers called *route reflectors* (RR), which utilize some BGP attributes defined specifically for a route reflection network. Each RR is assigned certain peers, known as *clients*, for which it reflects IBGP routes. Together, the route reflector and its clients are considered a *cluster* in the network. The route reflector sends and receives BGP routes for all other nonclient peers according to the rules set forth in the original BGP specification.

To prevent routing loops in the network, two new BGP attributes and a new identifier value are defined by the route reflection specification. These items are:

Cluster ID The *cluster ID* is similar in nature to the AS number defined for each router. A unique 32-bit value is assigned to each cluster in the network that identifies and separates it from other network clusters.

Cluster List The *Cluster List* is a BGP attribute that operates like the AS Path attribute. It contains a list of sequential cluster IDs for each cluster a particular route has transited. It is used as the main loop avoidance mechanism and is never transmitted outside the local AS.

Originator ID The *Originator ID* is also a BGP attribute defined for use by route reflectors. It identifies the router that first advertised the route to the route reflector in the network. Route reflectors use the Originator ID as a second check against routing loops within the AS. Like the Cluster List attribute, the Originator ID is local to the AS and is never transmitted across an EBGP peering session.

Operational Theory

Now that we've touched on the basics of what a route reflection network is and the terms used to describe it, let's discuss how routes are propagated. In addition, we'll touch on some design issues related to route reflection.

Figure 5.8 shows the Zinfandel, Chablis, and Cabernet routers in a route reflection cluster. The Zinfandel router is the route reflector for the cluster, assigned an identifier of 1.1.1.1, while Chablis and Cabernet are the clients. Each of the clients forms an IBGP peering session with just the RR and not with each other. This reduction in peering sessions pays large dividends as the network size grows and greatly reduces the number of overall peering sessions in the AS. The route reflector forwards active BGP routes based on how they were received using the following rules:

From an EBGP peer When an RR receives an active BGP route from an EBGP peer, it forwards the route to all clients in its cluster as well as to all other IBGP nonclient peers. The Cluster List and Originator attributes are added only to routes advertised to clients within the cluster.

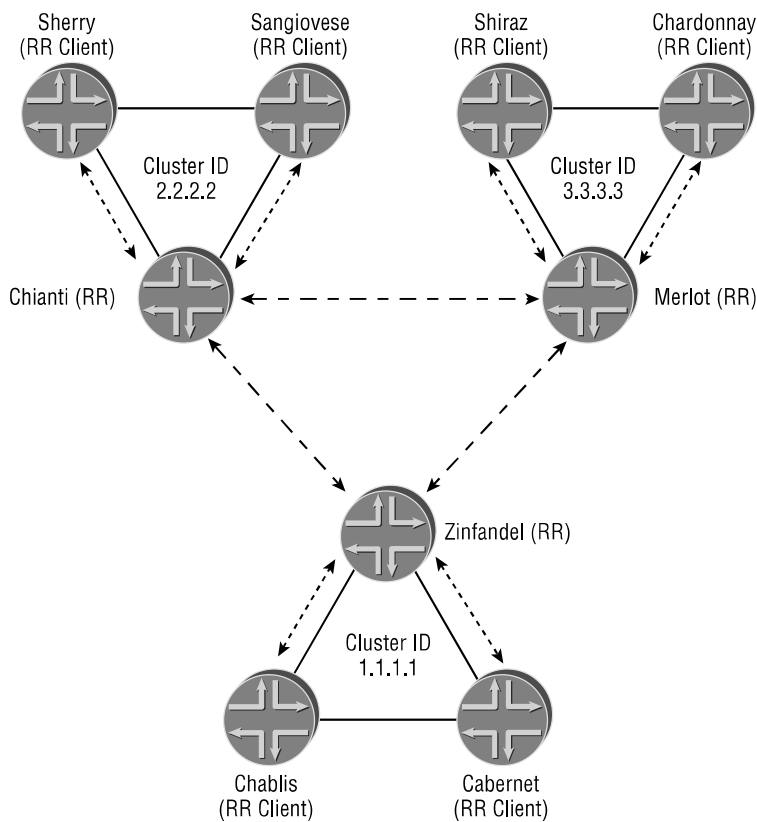
From an IBGP client peer When an RR receives an active BGP route from an IBGP peer that belongs to its cluster, it forwards the route to all other clients in its cluster as well as all other IBGP nonclient peers. These IBGP advertisements contain the Originator ID attribute and a modified Cluster List. The route reflector also advertises the route to all of its EBGP peers without adding the route reflection attributes.

From an IBGP nonclient peer When an RR receives an active BGP route from an IBGP peer that is not within a cluster, it forwards the route to all clients in its cluster with the appropriate attributes attached. The route reflector also advertises the routes to its EBGP peers without the route reflection attributes.



Even though the RR is readvertising routes, it is important to remember that it is still a BGP router. This means that route selection is performed on all path advertisements, a single route is selected and placed in the routing table, and that single route is advertised to its peers.

FIGURE 5.8 Basic route reflection network



Suppose that the Cabernet router receives a path advertisement from an EBGp peer. The advertisement is selected as the active route and is readvertised to its IBGP peers. In our case, Cabernet has only a single peer—Zinfandel. The advertisement is received by Zinfandel, accepted, and placed into the routing table. Since the route was learned from an IBGP peer in its cluster, Zinfandel reflects the route inside the cluster and advertises it to Chablis. During the reflection process, Zinfandel attaches both the Originator ID and Cluster List attributes to the route. The router ID of Cabernet becomes the Originator ID for the route, and the cluster ID of 1.1.1.1 is placed into the Cluster List. None of the other attributes attached to the route are altered by default; they continue to represent the values set by Cabernet. Of course, the Zinfandel router also has additional IBGP peers in the network, so let's see what happens across those peering sessions.

The Chianti router is the RR for cluster 2.2.2.2 and Merlot is the RR for cluster 3.3.3.3, each with two clients in its cluster. These two route reflectors have nonclient peering sessions with each other in addition to the Zinfandel router. In the end, we have an IBGP full mesh established between Zinfandel, Chianti, and Merlot. These nonclient peers of Zinfandel must also receive a copy of the EBGp-learned route from the Cabernet router. This version of the route also includes the Cluster List and Originator ID attributes.

At this point, Zinfandel's participation in route reflection is complete. Both Chianti and Merlot, themselves route reflectors, receive the route and examine the Cluster List attribute for their local cluster ID. Neither of the routers finds a routing loop and installs the route. Since the route was received from a nonclient IBGP peer, both Chianti and Merlot may only readvertise the route to clients in their local cluster. This allows the Sherry, Sangiovese, Shiraz, and Chardonnay routers to receive the route that originally entered the local AS via Cabernet. In the end, each of these four routers sees no difference between a full-mesh IBGP and a route reflection network. The route was received internally with the exact same attribute values as advertised by Cabernet.

Hierarchical Route Reflection

In very large networks, the use of route reflection helps to keep the IBGP peering sessions and protocol reconfiguration to a minimum. However, the possibility exists for the full mesh of sessions between the route reflectors to grow too large. Figure 5.9 represents this exact scenario.

Our sample network now contains five route reflection clusters, each with two clients. There are now 10 IBGP sessions supporting the RR full mesh. As additional clusters are added to the network, we find ourselves facing the same problem that route reflection was designed to solve: large configurations and protocol state. This type of scenario is mitigated through the use of *hierarchical route reflection*, which allows a route reflector for one cluster to be a client in a separate cluster. The end result is the replacement of the RR full mesh with another cluster, as seen in Figure 5.10.

A new router is installed as the route reflector for cluster 6.6.6.6 with the existing reflectors as its clients. The readvertising rules for this new RR are identical to the rules used by the other reflectors. At a high level, let's see how each router in the network receives a route advertised by a client in cluster 1.1.1.1. The client sends the route to the RR for cluster 1.1.1.1, router A, who reflects it inside the cluster. Router A also sends the route to its nonclient IBGP peers, which is only router F in this case. From the viewpoint of router F, a route was just received from an RR client in cluster 6.6.6.6. This allows router F to readvertise the route within its cluster, which sends it to routers B, C, D, and E. Each of these five routers sees a route advertised by a nonclient IBGP peer and reflects that route into its particular cluster.

FIGURE 5.9 Large full mesh of RR peerings

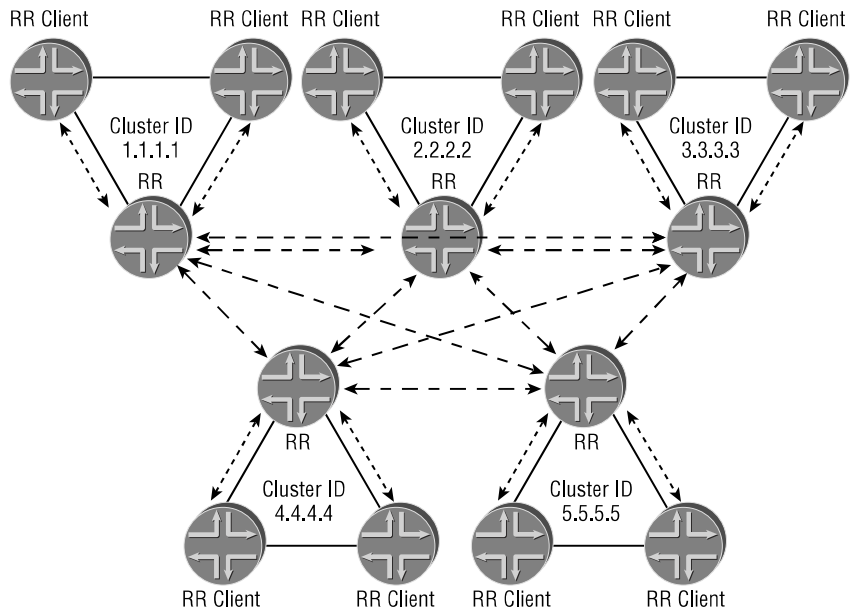
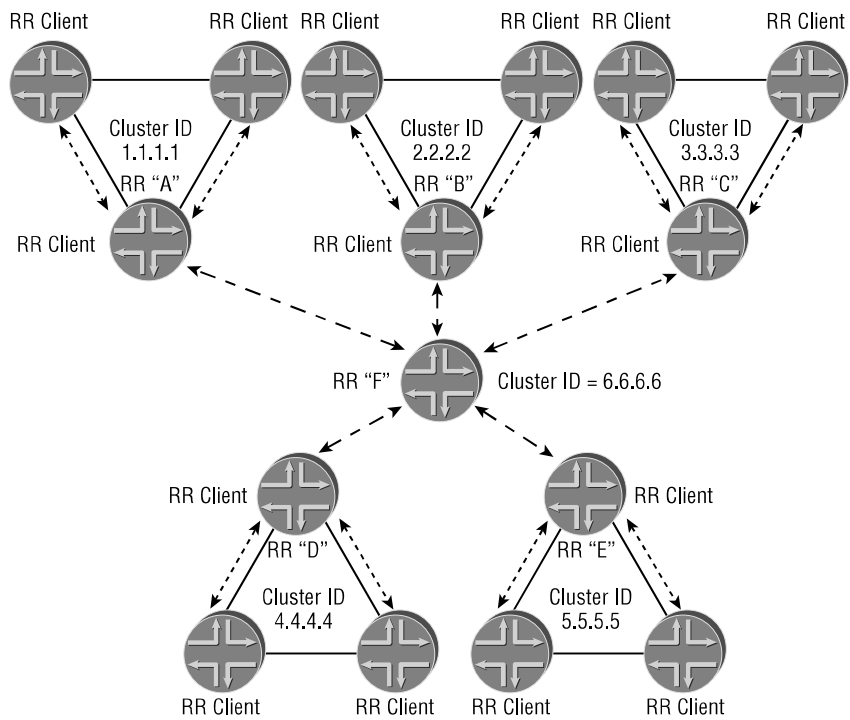


FIGURE 5.10 Hierarchical route reflection



The success of a hierarchical route reflection network relies on the careful establishment of clients and route reflectors. When you assign these roles, always place yourself in the router's position and think about where routes are received from and which peers you can readvertise those routes to.



Hierarchical route reflection has no limit to the number of levels or layers used. Provided that reachability is maintained and no routing loops are introduced, you can build your network in any fashion you desire.

Designing for Redundancy

In the examples we've examined thus far, you may have noticed the potential for disaster to strike. Specifically, the way in which we've used our route reflectors leaves us exposed to a single point of failure—the route reflector itself. When a client establishes an IBGP peering session to a single RR, it becomes reliant on that peer for all network reachability. Should the route reflector stop operating, the client no longer has the ability to forward and receive traffic from the core of the AS. This vulnerability leads many network administrators to use two route reflectors in each cluster.

FIGURE 5.11 Using two route reflectors in a cluster

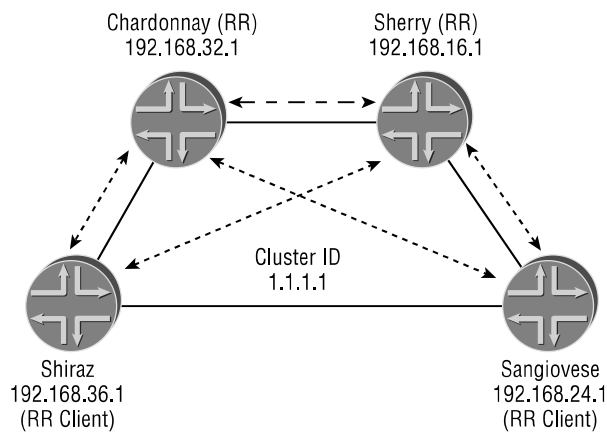


Figure 5.11 shows a cluster with both Sherry and Chardonnay as route reflectors in cluster 1.1.1.1. Both of the reflectors establish a session between themselves as well as sessions within the cluster to the Shiraz and Sangiovese routers. In our example, the RR peering is accomplished outside the cluster, but this is not required. This configuration doesn't alter the operation of the route reflectors with regard to the advertisement of routing information. When the Sangiovese router receives a route from an EBGP peer, it sends it to its two IBGP peers of Sherry and Chardonnay. Both reflectors add the Originator ID and Cluster List attributes to the route and reflect

it within the cluster to Shiraz. Additionally, both Sherry and Chardonnay forward the route to their nonclient IBGP peers (which are each other). If we look at this last piece of the flooding, we gain an interesting insight into the operation of route reflection. Because both Sherry and Chardonnay are configured as route reflectors, each router examines the incoming routes for the Cluster List attribute. If the attribute is attached, the router further looks for their local cluster ID in the Cluster List. In our example, both routers see the ID of 1.1.1.1 in the Cluster List and drop the route. This process occurs even if the route reflectors are peering within the cluster as clients of one another.

Configuring the Network

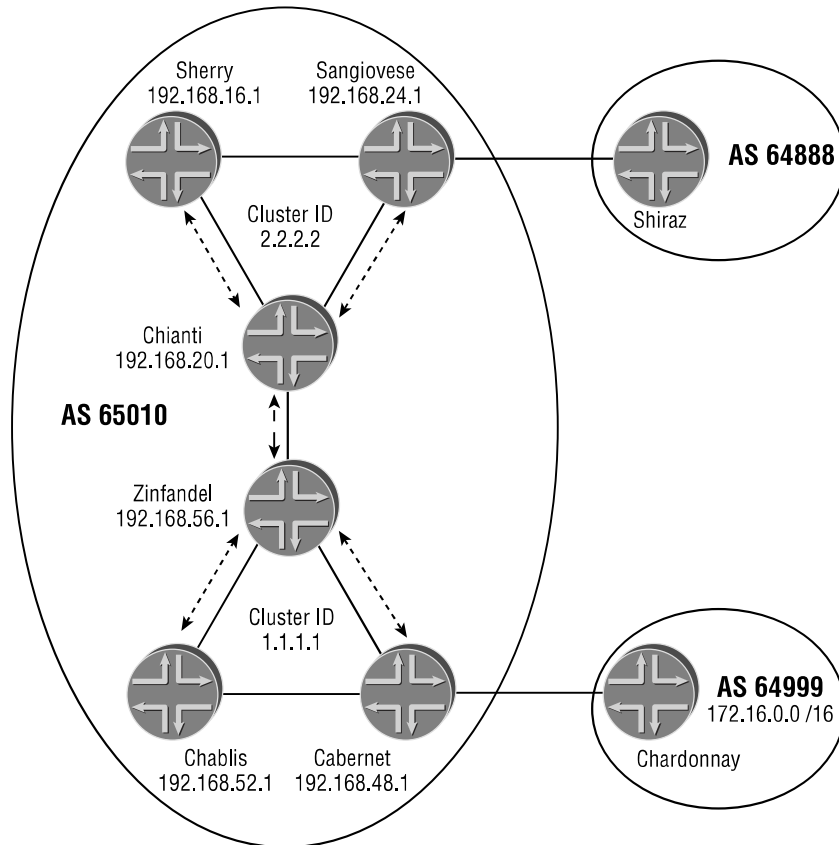
Configuring route reflection within the JUNOS software is a very simple and straightforward process. After establishing the appropriate peering sessions, you only need to configure the route reflector using the `cluster value` command within an appropriate peer group. This command treats each IBGP peer in the peer group as a route reflection client. All other configured peers on the router are treated as nonclient peers. The `value` portion of the command contains the 32-bit unique cluster identifier for your network.

Choosing the Cluster ID

From a technical standpoint, the specific value you assign as the cluster ID is insignificant as long as it is unique throughout your network. It appears within the Cluster List attribute only on internally advertised routes and is used for routing loop avoidance. From a troubleshooting perspective, however, the choice of the cluster ID values may have great importance. Let's talk about two main scenarios.

The first possibility is a route reflection cluster with a single route reflector. In this instance, many network administrators find it helpful to use the router ID of the route reflector as the cluster ID value. When you use this system, you can view the Cluster List of any route in your network and immediately know which route reflectors have readvertised the route.

The second possibility is a cluster with multiple route reflectors, often two routers. Using the router ID from one of the route reflectors is not as helpful in this case, but some networks do use this system. This is especially true when the cluster represents a point of presence (POP) in the network. Seeing the router ID in the Cluster List at least provides you with the POPs the route has traversed. More often than not, the cluster ID for a dual-route reflector cluster is an arbitrary value that makes sense to the network administrators. Many networks use a system similar to the one we've employed in this chapter; 1.1.1.1, followed by 2.2.2.2, followed by 3.3.3.3, etc. As long as the system is consistent and straightforward, it will be easily understood when troubleshooting must occur.

FIGURE 5.12 Basic route reflection sample network

AS 65010 in Figure 5.12 contains six routers running a route reflection network. The Chianti router is a reflector for cluster 2.2.2.2 and the Zinfandel router is a reflector for cluster 1.1.1.1, with each cluster containing two clients. The Cabernet router has an EBGP peering session with Chardonnay in AS 64999 and is receiving routes in the 172.16.0.0 /16 address space. These routes appear on Cabernet as

```
user@Cabernet> show route protocol bgp
```

```
inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
```

```
+ = Active Route, - = Last Active, * = Both
```

```
172.16.1.0/24      *[BGP/170] 00:33:23, MED 0, localpref 100
                  AS path: 64999 I
                  > to 10.222.6.2 via fe-0/0/1.0
```

The configuration of Cabernet shows only a single peering session to its local route reflector of Zinfandel (192.168.56.1):

```
user@Cabernet> show configuration protocols bgp
group internal-peers {
    type internal;
    local-address 192.168.48.1;
    export nhs;
    neighbor 192.168.56.1;
}
group external-peers {
    type external;
    neighbor 10.222.6.2 {
        peer-as 64999;
    }
}
```

We can now examine the routing table on the Zinfandel router and see the 172.16.0.0 /16 routes:

```
user@Zinfandel> show route protocol bgp

inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 00:36:08, MED 0, localpref 100, from 192.168.48.1
                  AS path: 64999 I
                  > to 10.222.61.2 via so-0/3/0.0
```

When we look at the configuration of Zinfandel, we see two peer groups configured. The **internal-peers** group contains the address of Chianti, its nonclient IBGP peer. The **cluster-1** group contains both the Chablis and Cabernet routers. The `cluster 1.1.1.1` command in the **cluster-1** group defines these peers as route reflection clients:

```
user@Zinfandel> show configuration protocols bgp
group internal-peers {
    type internal;
    local-address 192.168.56.1;
    neighbor 192.168.20.1;
}
group cluster-1 {
    type internal;
    local-address 192.168.56.1;
    cluster 1.1.1.1;
```

```

neighbor 192.168.48.1;
neighbor 192.168.52.1;
}

```

Applying what we know about the operation of Zinfandel as a route reflector, we assume that the 172.16.0.0/16 routes are advertised to both Chablis and Chianti. The output of the `show route advertising-protocol bgp neighbor-address` command proves this to be a correct assumption:

```
user@Zinfandel> show route advertising-protocol bgp 192.168.52.1
```

```

inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.16.1.0/24         192.168.48.1    0      100      64999 I

```

```
user@Zinfandel> show route advertising-protocol bgp 192.168.20.1
```

```

inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.16.1.0/24         192.168.48.1    0      100      64999 I

```

When we use the `detail` option, we see the Originator ID attached to the routes. The router ID of Cabernet (192.168.48.1) is used as the Originator ID since it was the router that first advertised the route to a route reflector in the network. In addition, we see the local cluster ID value that the router prepends into the Cluster List attribute. The Cluster List itself is not displayed since Zinfandel adds the attribute during the prepend operation:

```
user@Zinfandel> show route advertising-protocol bgp 192.168.52.1 detail
```

```

inet.0: 20 destinations, 20 routes (20 active, 0 holddown, 0 hidden)
* 172.16.1.0/24 (1 entry, 1 announced)
  BGP group cluster-1 type Internal
    Nexthop: 192.168.48.1
    MED: 0
    Localpref: 100
    AS path: 64999 I
  Communities:
    Cluster ID: 1.1.1.1
    Originator ID: 192.168.48.1

```



Remember that the output of the `show route advertising-protocol bgp neighbor-address` command displays the effect of all outgoing policies with the exception of the default AS Path prepend action for EBGp peers. This same concept holds true for the Cluster List attribute.

Once the routes are installed on the other client in cluster 1.1.1.1 (Chablis), we can see all of the route reflection attributes applied to the route:

```
user@Chablis> show route 172.16.1/24 detail

inet.0: 19 destinations, 19 routes (19 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
  *BGP      Preference: 170/-101
            Source: 192.168.56.1
            Next hop: 10.222.60.2 via fe-0/0/0.0, selected
            Protocol next hop: 192.168.48.1 Indirect next hop: 84cfbd0 57
            State: <Active Int Ext>
            Local AS: 65010 Peer AS: 65010
            Age: 51:40      Metric: 0      Metric2: 10
            Task: BGP_65010.192.168.56.1+179
            Announcement bits (2): 0-KRT 4-Resolve inet.0
            AS path: 64999 I (Originator) Cluster list: 1.1.1.1
            AS path: Originator ID: 192.168.48.1
            Localpref: 100
            Router ID: 192.168.56.1
```

The Chianti router is advertising the route within its cluster since it was received from a non-client IBGP peer (Zinfandel):

```
user@Chianti> show route advertising-protocol bgp 192.168.16.1

inet.0: 22 destinations, 22 routes (22 active, 0 holddown, 0 hidden)
  Prefix          Nexthop          MED    Lclpref    AS path
* 172.16.1.0/24   192.168.48.1    0      100        64999 I

user@Chianti> show route advertising-protocol bgp 192.168.24.1

inet.0: 22 destinations, 22 routes (22 active, 0 holddown, 0 hidden)
  Prefix          Nexthop          MED    Lclpref    AS path
* 172.16.1.0/24   192.168.48.1    0      100        64999 I
```

The 172.16.1.0 /24 route is visible on the Sangiovese router as

```
user@Sangiovese> show route 172.16.1/24 detail

inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
  *BGP      Preference: 170/-101
```

```

Source: 192.168.20.1
Next hop: 10.222.28.1 via fe-0/0/0.0, selected
Protocol next hop: 192.168.48.1 Indirect next hop: 84cfbd0 68
State: <Active Int Ext>
Local AS: 65010 Peer AS: 65010
Age: 54:43      Metric: 0      Metric2: 40
Task: BGP_65010.192.168.20.1+4330
Announcement bits (3): 0-KRT 3-BGP.0.0.0.0+179 4-Resolve inet.0
AS path: 64999 I (Originator) Cluster list: 2.2.2.2 1.1.1.1
AS path: Originator ID: 192.168.48.1
Localpref: 100
Router ID: 192.168.20.1

```

As we should expect, the Originator ID value is still set to 192.168.48.1, the router ID of Cabernet. This attribute remains constant throughout the route reflection network and is stripped from the route when it is advertised across an AS boundary. In addition, the Cluster List attribute shows us that the route transited cluster 1.1.1.1 followed by cluster 2.2.2.2. This second ID value was prepended onto the list by Chianti when it reflected the route. When the routes are received by the Shiraz router in AS 64888, we see that the route reflection attributes are removed and that the Sangiovese router added the local AS of 65010 to the AS Path attribute:

```

user@Shiraz> show route receive-protocol bgp 10.222.4.1 detail

inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
* 172.16.1.0/24 (1 entry, 1 announced)
  Nexthop: 10.222.4.1
  AS path: 65010 64999 I

```

In accordance with the general theory of BGP, Shiraz has no knowledge of the internal connectivity of AS 65010. It only knows that the active path transits this AS.

Hierarchical Route Reflection

As we discussed in the “Operational Theory” section earlier, the operation of a network using hierarchical route reflection is no different than a simple route reflection network. It should come as no surprise, then, that the configuration of hierarchical route reflection is no different.

Figure 5.13 shows seven routers in AS 65010 arrayed in a hierarchical route reflection design. The configuration of Zinfandel, the route reflector for Cluster 1.1.1.1, appears as so:

```

user@Zinfandel> show configuration protocols bgp
group internal-peers {
  type internal;
  local-address 192.168.56.1;
  neighbor 192.168.20.1;
}

```

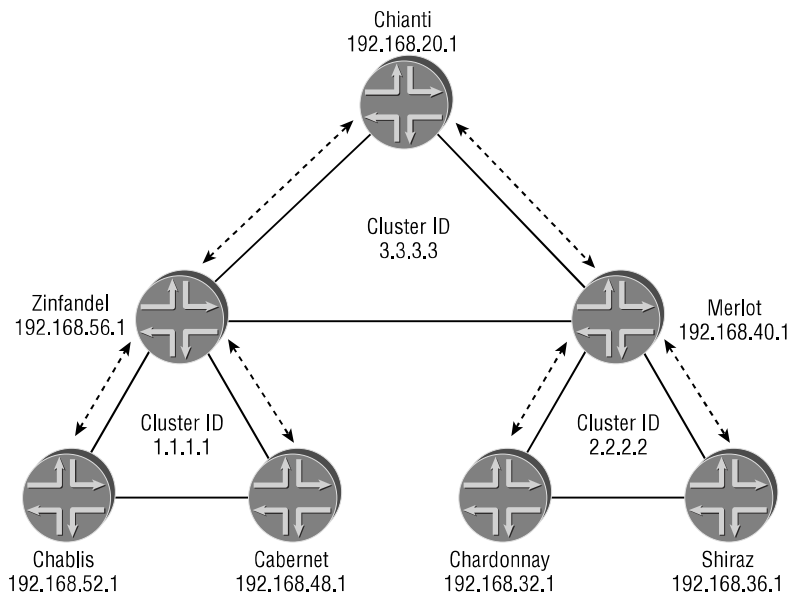


```
group cluster-1 {
  type internal;
  local-address 192.168.56.1;
  cluster 1.1.1.1;
  neighbor 192.168.48.1;
  neighbor 192.168.52.1;
}
```

This is very similar to the route reflector configuration for cluster 2.2.2.2 on the Merlot router:

```
user@Merlot> show configuration protocols bgp
group internal-peers {
  type internal;
  local-address 192.168.40.1;
  neighbor 192.168.20.1;
}
group cluster-2 {
  type internal;
  local-address 192.168.40.1;
  cluster 2.2.2.2;
  neighbor 192.168.32.1;
  neighbor 192.168.36.1;
}
```

FIGURE 5.13 Hierarchical route reflection sample network



In fact, if you were just shown the configurations of these two route reflectors you might not know that hierarchical route reflection was being used. From the viewpoint of these routers, they are each responsible for a single cluster and have an additional IBGP peer. You do have one clue available to you and it lies in the *internal-peers* peer group of each router. In a simple route reflection network, each reflector peers with all of the other route reflectors. This particular peer group contains only a single neighbor statement for 192.168.20.1, which is not a peer route reflector but the same third router. The configuration of this router, Chianti, shows a cluster ID of 3.3.3.3 and route reflector clients of Zinfandel and Merlot:

```
user@Chianti> show configuration protocols bgp
group cluster-3 {
    type internal;
    local-address 192.168.20.1;
    cluster 3.3.3.3;
    neighbor 192.168.40.1;
    neighbor 192.168.56.1;
}
```

Suppose that the Shiraz router receives routes in the 172.16.0.0 /16 address range from an EBGP peer in AS 64999. These routes include the following:

```
user@Shiraz> show route protocol bgp

inet.0: 25 destinations, 25 routes (25 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 12:32:56, MED 0, localpref 100
                  AS path: 64999 I
                  > to 10.222.4.1 via fe-0/0/0.0
```

Based on the configurations of the route reflectors, we can assume that each router in the network has received these same routes and installed them in their local routing table. If we examine the details of the route attributes on the Chablis router, for example, we should find that the Originator ID is set to the router ID of Shiraz (192.168.36.1). Additionally, the Cluster List attribute should appear as 1.1.1.1 3.3.3.3 2.2.2.2 since the route was reflected first by Merlot, then by Chianti, and finally by Zinfandel. Let's verify our assumptions by viewing the 172.16.1.0 /24 route on Chablis:

```
user@Chablis> show route 172.16.1/24 detail

inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
    *BGP Preference: 170/-101
    Source: 192.168.56.1
```

```

Next hop: 10.222.60.2 via fe-0/0/0.0, selected
Protocol next hop: 192.168.36.1 Indirect next hop: 84cfbd0 57
State: <Active Int Ext>
Local AS: 65010 Peer AS: 65010
Age: 12:24:08 Metric: 0 Metric2: 30
Task: BGP_65010.192.168.56.1+179
Announcement bits (2): 0-KRT 4-Resolve inet.0
AS path: 64999 I (Originator) Cluster list: 1.1.1.1 3.3.3.3 2.2.2.2
AS path: Originator ID: 192.168.36.1
Localpref: 100
Router ID: 192.168.56.1

```

Using Two Route Reflectors

For completeness, let's take a quick moment to examine the configuration and operation of a route reflection cluster containing two reflectors. Using Figure 5.11 as a guide, we see that the configuration of the clients, Sangiovese and Shiraz, appear as normal BGP configurations. The main difference, in this case, is the inclusion of two internal peers as opposed to a single peer in our simple route reflection examples:

```

user@Sangiovese> show configuration protocols bgp
group internal-peers {
    type internal;
    local-address 192.168.24.1;
    neighbor 192.168.16.1;
    neighbor 192.168.32.1;
}

```

```

user@Shiraz> show configuration protocols bgp
group internal-peers {
    type internal;
    local-address 192.168.36.1;
    neighbor 192.168.16.1;
    neighbor 192.168.32.1;
}

```

In fact, the configurations of the two route reflectors, Sherry and Chardonnay, also appear very similar to a simple route reflection network. The exception is the configuration of the `cluster 1.1.1.1` command on both routers:

```

user@Sherry> show configuration protocols bgp
group internal-peers {

```

```

    type internal;
    local-address 192.168.16.1;
    neighbor 192.168.32.1;
}
group cluster-1 {
    type internal;
    local-address 192.168.16.1;
    cluster 1.1.1.1;
    neighbor 192.168.24.1;
    neighbor 192.168.36.1;
}

```

```

user@Chardonnay> show configuration protocols bgp
group internal-peers {
    type internal;
    local-address 192.168.32.1;
    neighbor 192.168.16.1;
}
group cluster-1 {
    type internal;
    local-address 192.168.32.1;
    cluster 1.1.1.1;
    neighbor 192.168.24.1;
    neighbor 192.168.36.1;
}

```

Once the IBGP peering sessions are established, the Sangiovese router advertises its received EBGP routes to both route reflectors:

```

user@Sangiovese> show route advertising-protocol bgp 192.168.16.1

inet.0: 15 destinations, 15 routes (15 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.16.1.0/24        Self              0      100     64999 I

```

```

user@Sangiovese> show route advertising-protocol bgp 192.168.32.1

inet.0: 15 destinations, 15 routes (15 active, 0 holddown, 0 hidden)
  Prefix                Nexthop          MED    Lc1pref  AS path
* 172.16.1.0/24        Self              0      100     64999 I

```

Both Sherry and Chardonnay reflect the routes within the cluster to the Shiraz router, providing two path advertisements for the same set of routes. The BGP route selection algorithm is run against these multiple paths and an active route is selected. In this case, the router ID of Sherry (192.168.16.1) is better than the router ID of Chardonnay (192.168.32.1). The `show route detail` output for the 172.16.1.0 /24 routes also shows the correct route reflection attributes attached to the routes:

```
user@Shiraz> show route 172.16.1/24 detail
```

```
inet.0: 15 destinations, 18 routes (15 active, 0 holddown, 0 hidden)
172.16.1.0/24 (2 entries, 1 announced)
  *BGP Preference: 170/-101
    Source: 192.168.16.1
    Next hop: 10.222.4.1 via fe-0/0/0.0, selected
    Protocol next hop: 192.168.24.1 Indirect next hop: 84cfd0 52
    State: <Active Int Ext>
    Local AS: 65010 Peer AS: 65010
    Age: 19:15 Metric: 0 Metric2: 10
    Task: BGP_65010.192.168.16.1+179
    Announcement bits (2): 0-KRT 4-Resolve inet.0
    AS path: 64999 I (Originator) Cluster list: 1.1.1.1
    AS path: Originator ID: 192.168.24.1
    Localpref: 100
    Router ID: 192.168.16.1
  BGP Preference: 170/-101
    Source: 192.168.32.1
    Next hop: 10.222.4.1 via fe-0/0/0.0, selected
    Protocol next hop: 192.168.24.1 Indirect next hop: 84cfd0 52
    State: <NotBest Int Ext>
    Inactive reason: Router ID
    Local AS: 65010 Peer AS: 65010
    Age: 13:11 Metric: 0 Metric2: 10
    Task: BGP_65010.192.168.32.1+1801
    AS path: 64999 I (Originator) Cluster list: 1.1.1.1
    AS path: Originator ID: 192.168.24.1
    Localpref: 100
    Router ID: 192.168.32.1
```

Both Sherry and Chardonnay also reflect the routes to their nonclient IBGP peers—each other. If we just examine the routes sent from Sherry to Chardonnay, we see the following advertisement:

```
user@Sherry> show route advertising-protocol bgp 192.168.32.1 detail
```

```
inet.0: 19 destinations, 19 routes (19 active, 0 holddown, 0 hidden)
```

```
* 172.16.1.0/24 (1 entry, 1 announced)
BGP group internal-peers type Internal
  Nexthop: 192.168.24.1
  MED: 0
  Localpref: 100
  AS path: 64999 I
Communities:
  Cluster ID: 1.1.1.1
  Originator ID: 192.168.24.1
```

It appears as if the appropriate attributes are attached to the route as they are advertised. However, Chardonnay reports that it hasn't received any routes from Sherry:

```
user@Chardonnay> show route receive-protocol bgp 192.168.16.1
```

```
inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
```

```
user@Chardonnay>
```

This is a similar symptom to an AS Path routing loop. One EBGP peer states that it advertised routes, but the other peer states that it didn't receive them. In fact, that's exactly the case we have here except the routing loop is a function of the Cluster List attribute. Because both of the route reflectors are configured with `cluster 1.1.1.1`, the receipt of routes with that value in the Cluster List signals a routing loop. As such, the routes are immediately dropped and not shown via any CLI command. To verify this theory, we can view the received Update packet from Sherry using the `monitor traffic interface` command:

```
user@Chardonnay> monitor traffic interface fe-0/0/0.1 size 4096 detail
```

```
Listening on fe-0/0/0.1, capture size 4096 bytes
```

```
06:33:34.424251 In IP (tos 0xc0, ttl 64, id 24353, len 133)
  192.168.16.1.bgp > 192.168.32.1.3210: P 19:100(81) ack 20 win 16384
  <nop,nop,timestamp 75053260 75008652>: BGP, length: 81
    Update Message (2), length: 81
      Origin (1), length: 1, flags [T]: IGP
      AS Path (2), length: 4, flags [T]: 64999
      Next Hop (3), length: 4, flags [T]: 192.168.24.1
      Multi Exit Discriminator (4), length: 4, flags [O]: 0
      Local Preference (5), length: 4, flags [T]: 100
      Originator ID (9), length: 4, flags [O]: 192.168.24.1
      Cluster List (10), length: 4, flags [O]: 1.1.1.1
    Updated routes:
      172.16.1.0/24
      172.16.2.0/24
      172.16.3.0/24
```

Confederations

A network operating with a *confederation* paradigm approaches the full-mesh problem by breaking the network up into smaller pieces. Each piece is considered to be a *sub-AS*, or *member AS*, of the larger global confederation that is your network. Each sub-AS is assigned its own unique AS number, and the normal rules of BGP still apply within that sub-AS. This means that a full mesh of IBGP peering sessions is required and no router may readvertise an IBGP-learned route to another IBGP peer. Connectivity between the sub-AS networks is maintained using a modified form of EBGp often called *confederation BGP* (CBGP). CBGP peers add their sub-AS number to the AS Path attribute as routes are exchanged, which allows the AS Path to still be used for preventing routing loops. When routes are further advertised out of the confederation, your global AS, the details of the sub-AS networks are removed from the AS Path and replaced with your global AS number. This keeps the details of your internal network invisible to other systems in the spirit of the original BGP specifications.

Throughout our discussion, we use various terms specific to operating and configuring a confederation network. In fact, we've already mentioned a few; these terms include the following:

AS Confederation The *AS confederation* is technically the collection of the sub-AS networks you create. Generally speaking, it is your globally assigned AS number, and it is how other systems in the Internet view you.

AS Confederation ID Your *AS confederation ID* is the value that identifies your AS confederation as a whole to the Internet. In other words, it is your globally unique AS number.

Member AS *Member AS* is the formal name for each sub-AS you create in your network. In essence, each small network is a member of the larger confederation that makes up your global AS.

Member AS Number Each member AS in your confederation receives its own *member AS number*. This unique value is placed into the AS Path attribute and is used for loop prevention.

Operational Theory

Each sub-AS in a confederation network uses BGP in a way that looks and acts like a “real” AS. It's assigned its own unique identifier from the private AS range, all of the peers form IBGP peering relationships, and routes are not readvertised among the internal routers.

A typical confederation network is displayed in Figure 5.14 within the global AS 1111. Sub-AS 64555 contains the Sherry, Sangiovese, and Chianti routers, which have formed an IBGP full mesh within the member AS. The Shiraz, Chardonnay, and Merlot routers in sub-AS 64777 have done the same. Routes advertised into either of the member AS networks are advertised to each of the peer routers in the sub-AS to maintain reachability. The real “power” of a confederation network is the ability to connect the member AS networks together using CBGP peering sessions.

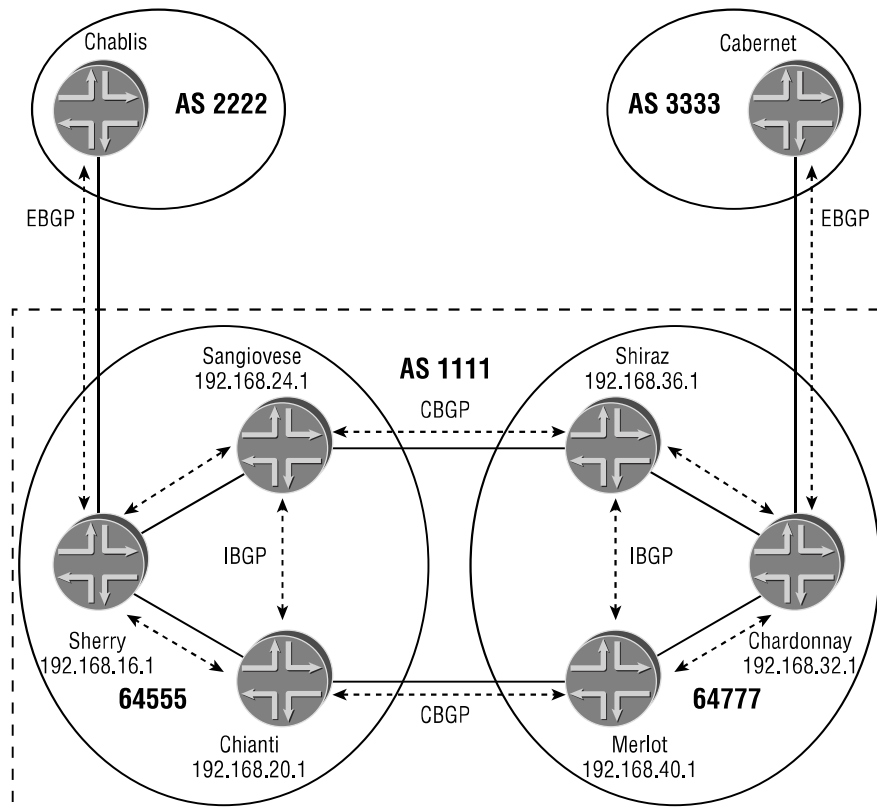
Confederation BGP sessions are very similar to EBGp peering sessions. The two routers share a common physical subnet, they belong to two different sub-AS networks, and they modify the AS Path attribute when advertising routes to each other. The main difference between the two peering types is how the rest of the BGP attributes are treated. For a CBGP peering session, these attributes are not modified or deleted by default. As an example, this action allows the Local Preference value to be seen by all routers in the confederation. In turn, all routers in the global

AS can now make consistent routing and forwarding decisions for all routes. The information added to the AS Path by a CBGP router is contained in one of two newly defined AS Path segments. The *AS Confederation Sequence*, segment type code 3, is an ordered list of the member AS networks through which the route has passed. It is operationally identical to an AS Sequence segment and is the default segment type used by the JUNOS software. The second new path segment is an *AS Confederation Set*, segment type code 4. Much like the AS Set, this new segment contains an unordered list of member AS numbers and is typically generated due to route aggregation within the global AS network.



The AS Confederation Sequence and AS Confederation Set path segments are not used when the router calculates the length of the AS path for route selection purposes. Only global AS values count toward the overall path length.

FIGURE 5.14 BGP confederation network





Real World Scenario

IBGP Scaling in a Sub-AS

Depending on the exact design of your confederation network, it is entirely possible that a single sub-AS portion may grow quite large. Since each sub-AS must maintain an IBGP full mesh of peering sessions, we might end up with the same situation our confederation was supposed to solve. One solution to this issue is segmenting the large sub-AS into smaller sub-AS networks. Another option uses route reflection within the sub-AS for scalability.

Routes received from an EBGP or CBGP peer are advertised to all IBGP peers within the sub-AS. These routes are not readvertised to other IBGP peers due to the full-mesh requirement. Route reflection clusters can effectively operate within a sub-AS since they replace the concept of a full mesh. Let's see how this might work.

Suppose that routers A, B, and C are all within a single sub-AS. Router A has an EBGP peer from which it is receiving routes, and router C has a CBGP peer to advertise routes to. In a normal sub-AS, router A receives routes from its EBGP peer and advertises them directly to router C, where they are sent to the CBGP peer. We now make these three routers a route reflection cluster with router B as the reflector. When router A receives the EBGP routes, it now only sends them to router B, where they are reflected to router C. Router C accepts these routes and then readvertises them to its CBGP peer. In the end, the routes take an "extra hop" as they are advertised across the sub-AS, but the end result is the same. When a large number of routers exist in the sub-AS, the benefit of the route reflection cluster outweighs the liability of the "extra hop."

The confederation as a whole connects to other global AS networks using EBGP peering sessions. Routes advertised across this connection abide by all of the normal BGP rules regarding attributes. Local Preference is removed from the routes, the AS Path is updated with your globally assigned AS, and other nontransitive attributes are removed from the routes.

Within our example confederation network we can see all of the peering types used for connectivity. The two sub-AS networks of 64555 and 64777 are connected using CBGP peering sessions on the Sangiovese-Shiraz link as well as the Chianti-Merlot link. The confederation is assigned the globally unique AS value of 1111 and connects to the Chablis router in AS 2222 as well as the Cabernet router in AS 3333. Let's see how these peering sessions affect the advertisement of routes in the confederation.

Suppose that the Cabernet router in AS 3333 advertises routes to its EBGP peer of Chardonnay. The routes are selected as active, placed in the local routing table, and readvertised by Chardonnay to any additional EBGP peers, all CBGP peers, and all IBGP peers. The only routers fitting any of these descriptions are Merlot and Shiraz, which are IBGP peers of Chardonnay. Each of these routers selects the routes as active and places them in the local routing table. Since the routes were received over an IBGP session, Merlot and Shiraz can only advertise them to EBGP and CBGP peers. As such, Merlot sends the routes to Chianti and Shiraz sends the routes to Sangiovese. During this announcement, both routers add the member AS value of 64777 as an AS Confederation Sequence within the AS Path attribute.

The routes are now received in member AS 64555 by Sangiovese and Chianti, which each check the AS Path attribute for their local member AS number. Not finding it in the attribute, they assume that a routing loop is not forming and accept the routes. Because the routes were received from a CBGP peer, these routers can advertise them to any EBGP, CBGP, or IBGP peers with established sessions. In our case, the IBGP full mesh means that each router sends the routes to each other as well as to Sherry. At this point, Sherry accepts the routes, installs them, and advertises them to any CBGP or EBGP peers. The Sherry router has only a single EBGP peering session to Chablis, so the routes are advertised into AS 2222. During this announcement, Sherry removes all AS Confederation Sequence and AS Confederation Set path segments from the AS Path attribute. In their place, the global AS value of 1111 is added to the path using the BGP default prepend action.



The removal of the member AS numbers, which are usually private AS values, is completed automatically by the configuration of the confederation. Using the `remove-private` command for this purpose does not accomplish this goal. In fact, the command interferes with reachability within your confederation. For a further explanation of this negative characteristic, please see the *JNCIP Study Guide*.

Configuring the Network

The configuration of a confederation network within the JUNOS software occurs entirely within the `[edit routing-options]` configuration hierarchy. You first assign the local member AS value to the router using the `autonomous-system value` command, where the global AS value is normally configured. You then inform your router that it is participating in a confederation network by using the `confederation value members [member-AS-numbers]` command. The `value` portion of this command is the confederation identifier assigned to your network—your globally assigned AS number. Each of the member AS values you've assigned within your confederation, including your local member AS, are included in the `member-AS-numbers` portion of the command. The `confederation` command allows the router to know if the external session you've established should operate as a CBGP session or an EBGP session.

Using Figure 5.14 as a guide, we can see that the BGP configuration of the Chardonnay router is quite ordinary. It has a peer group for its EBGP peer and a peer group for its internal sub-AS peers:

```
user@Chardonnay> show configuration protocols bgp
group EBGP-Peers {
    type external;
    peer-as 3333;
    neighbor 10.222.6.1;
}
group sub-AS-Peers {
    type internal;
    local-address 192.168.32.1;
    export nhs;
    neighbor 192.168.36.1;
    neighbor 192.168.40.1;
}
```

Choosing the Member AS Values

Technically speaking, the values you assign to your member AS networks are completely contained within your confederation network, assuming you've configured everything correctly. This means that the values can be any AS number that is different from your globally unique AS value. However, it is considered a best practice by most network administrators that the member AS values be assigned from the private AS range. This is helpful for several reasons.

First, the member AS values are placed into the AS Path attribute within the confederation. When private AS numbers are used, you can easily spot these values in the output of the `show route` command to view the path taken by the route or troubleshoot why a particular route is not being used to forward traffic. Second, using private AS numbers allows for easier readability of your configuration. You'll see that CBGP and EBGP peer configurations look very similar, even identical. Without you constantly referring to a network map or consulting the `[edit routing-options]` hierarchy, the private AS numbers clearly show which peers are CBGP and which are EBGP.

Besides, if you use a nonprivate AS value within your confederation you might still receive a BGP route with that nonprivate AS value in the AS Path. In this situation, you'll drop that route since the local router believes that a routing loop is forming.

The details of the confederation configuration are within the `routing-options` configuration hierarchy. When we examine this portion of the configuration, we see that Chardonnay has configured its sub-AS value using the `autonomous-system` command:

```
user@Chardonnay> show configuration routing-options
autonomous-system 64777;
confederation 1111 members [ 64555 64777 ];
```

The `confederation` command contains the globally unique AS number assigned to this network. In addition, each member AS in the confederation is listed. When taken together, these commands allow the routers to form peering relationships using the proper AS information. For example, the output of the `show bgp neighbor` command on the Cabernet router shows that the remote AS is 1111:

```
user@Cabernet> show bgp neighbor
Peer: 10.222.6.2+3801 AS 1111 Local: 10.222.6.1+179 AS 3333
Type: External State: Established Flags: <>
Last State: OpenConfirm Last Event: RecvKeepAlive
Last Error: Open Message Error
Export: [ adv-routes ]
Options: <Preference HoldTime PeerAS Refresh>
Holdtime: 90 Preference: 170
Number of flaps: 0
```

```

Error: 'Open Message Error' Sent: 4 Recv: 0
Peer ID: 192.168.32.1    Local ID: 192.168.48.1    Active Holdtime: 90
Keepalive Interval: 30
Local Interface: fe-0/0/1.0
---(more)---

```

The CBGP peering configurations in the network are very similar, so let's just examine the session between Chianti and Merlot. As with each other router in the confederation, both Chianti and Merlot have their member AS number and the confederation information configured within the routing-options hierarchy:

```

user@Chianti> show configuration routing-options
autonomous-system 64555;
confederation 1111 members [ 64555 64777 ];

```

```

user@Merlot> show configuration routing-options
autonomous-system 64777;
confederation 1111 members [ 64555 64777 ];

```

When we look at the BGP configuration of Chianti, we see two peer groups configured. The **sub-AS-Peers** group contains the addresses of Sherry and Sangiovese, its member AS IBGP peers. The **CBGP-Peers** group contains information on the Merlot router:

```

user@Chianti> show configuration protocols bgp
group sub-AS-Peers {
    type internal;
    local-address 192.168.20.1;
    neighbor 192.168.16.1;
    neighbor 192.168.24.1;
}
group CBGP-Peers {
    type external;
    multihop;
    local-address 192.168.20.1;
    peer-as 64777;
    neighbor 192.168.40.1;
}

```

While the CBGP peer group looks similar to a typical EBGP configuration, there are some differences. In following a confederation best practice, the CBGP sessions are configured to use the loopback address of the peer. Because the session is external in nature, the `multihop` command is required before the session is established. Reachability to the peer's loopback address is provided by the network's IGP, which is operational throughout the entire AS. You might also notice that no time-to-live (TTL) was specified in this configuration as we normally see for an EBGP peering session. This is an appropriate option for a typical EBGP peering since we

want the session to fail when a physical link failure occurs between the two routers. This core belief is not valid when considering a CBGP peering session. In fact, should the physical link between two peers fail, we want the session to remain active using whatever network links are available. This ensures that routes are still advertised to all routers in the confederation.



The omission of the TTL allows the router to use the default value of 64.

At this point, all of the peering sessions are established and Cabernet begins advertising routes to Chardonnay. These routes represent the 172.16.0.0 /16 address space and appear as so:

```
user@Chardonnay> show route protocol bgp terse
```

```
inet.0: 27 destinations, 27 routes (27 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

A Destination	P Prf	Metric 1	Metric 2	Next hop	AS path
* 172.16.1.0/24	B 170	100	0	>10.222.6.1	3333 I

These EBGP-learned routes are then advertised by Chardonnay to its IBGP peers of Shiraz and Merlot:

```
user@Chardonnay> show route advertising-protocol bgp 192.168.36.1
```

```
inet.0: 27 destinations, 27 routes (27 active, 0 holddown, 0 hidden)
  Prefix          Nexthop          MED    Lc1pref  AS path
* 172.16.1.0/24  Self             0      100     3333 I
```

```
user@Chardonnay> show route advertising-protocol bgp 192.168.40.1
```

```
inet.0: 27 destinations, 27 routes (27 active, 0 holddown, 0 hidden)
  Prefix          Nexthop          MED    Lc1pref  AS path
* 172.16.1.0/24  Self             0      100     3333 I
```

When we examine details of the 172.16.1.0 /24 route on Merlot, we see no information about our confederation network. This is not surprising since the routes have only been advertised within a single sub-AS at this point:

```
user@Merlot> show route 172.16.1/24 detail
```

```
inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
  *BGP   Preference: 170/-101
        Source: 192.168.32.1
        Next hop: 10.222.45.2 via so-0/3/1.0, selected
```

```

Protocol next hop: 192.168.32.1 Indirect next hop: 84cfbd0 49
State: <Active Int Ext>
Local AS: 64777 Peer AS: 64777
Age: 1d 6:37:44 Metric: 0 Metric2: 10
Task: BGP_64777.192.168.32.1+179
Announcement bits (3): 0-KRT 3-BGP.0.0.0.0+179 4-Resolve inet.0
AS path: 3333 I
Localpref: 100
Router ID: 192.168.32.1

```

The Merlot router now advertises the 172.16.0.0/16 routes to just its CBGP peer of Chianti and not its IBGP peer of Shiraz:

```
user@Merlot> show route advertising-protocol bgp 192.168.36.1
```

```
user@Merlot> show route advertising-protocol bgp 192.168.20.1
```

```
inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
  Prefix                Nexthop                MED    Lc1pref  AS path
* 172.16.1.0/24         192.168.32.1          0      100      3333 I
```



Remember that the output of the `show route advertising-protocol bgp neighbor-address` command doesn't display the default AS Path prepend action. This affects our ability to verify the addition of the AS Confederation Sequence with this output.

Once the routes are installed on Chianti, we can see sub-AS 64777 appear within the AS Path attribute for each route:

```
user@Chianti> show route protocol bgp terse
```

```
inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination      P Prf  Metric 1  Metric 2  Next hop      AS path
* 172.16.1.0/24    B 170    100        0 >10.222.1.2  (64777) 3333 I
```

Since Chianti learned these routes from a CBGP peer, it may advertise them to all of its IBGP peers, including Sherry. A quick look at the routing table of Sherry shows these routes:

```
user@Sherry> show route protocol bgp
```

```
inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
```

+ = Active Route, - = Last Active, * = Both

```
172.16.1.0/24      *[BGP/170] 00:09:44, MED 0, localpref 100, from 192.168.20.1
                  AS path: (64777) 3333 I
                  to 10.222.29.2 via ge-0/1/0.0
                  > to 10.222.28.2 via fe-0/0/0.0
```

As a final step, the routes are advertised to Sherry’s EBGP peer of Chablis in AS 2222:

```
user@Sherry> show route advertising-protocol bgp 10.222.5.2
```

```
inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
  Prefix          Nexthop          MED      Lclpref    AS path
* 172.16.1.0/24   Self              0         0          (64777) 3333 I
```

While we don’t see the removal of the AS Confederation Sequence with this command, a look at the actual packet transmission shows AS 1111 correctly prepended to the AS Path:

```
user@Sherry> monitor traffic interface fe-0/0/2.0 size 4096 detail
```

Listening on fe-0/0/2.0, capture size 4096 bytes

```
05:30:17.678882 Out IP (tos 0xc0, ttl 1, id 57922, len 107)
  10.222.5.1.4813 > 10.222.5.2.bgp: P 19:74(55) ack 100 win 16384
  <nop,nop,timestamp 92018522 91989155>: BGP, length: 55
    Update Message (2), length: 55
      Origin (1), length: 1, flags [T]: IGP
      AS Path (2), length: 6, flags [T]: 1111 3333
      Next Hop (3), length: 4, flags [T]: 10.222.5.1
      Updated routes:
        172.16.1.0/24
        172.16.2.0/24
        172.16.3.0/24
```

Of course, the correct AS Path information is also visible when we examine the routing table of Chablis:

```
user@Chablis> show route protocol bgp
```

```
inet.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
172.16.1.0/24      *[BGP/170] 00:02:08, localpref 100
                  AS path: 1111 3333 I
                  > to 10.222.5.1 via fe-0/0/1.0
```

Using Multiprotocol BGP

The maturity of BGP and its widespread use across the Internet make it a unique platform for advertising information both between ASs as well as inside them. This information might include IPv6 routes for forwarding user data traffic or IPv4 routes used in a multicast network for reverse path forwarding checks. Recently, information associated with virtual private networks (VPN) and Multiprotocol Label Switching (MPLS) has also been transmitted across BGP peering sessions. The ability of BGP to transmit this information is generally referred to as *Multiprotocol BGP* (MBGP). More specifically, MBGP is a capability negotiated between two peers during the establishment of the peering session. Each peer describes its ability to support different reachability information by sending a Capability option in the BGP Open message. Figure 5.15 shows the format of the Capability option, whose fields include the following:

Capability Type This field displays the actual capability being negotiated between the peers. For MBGP, this field is set to a constant value of 1, which signifies multiprotocol extensions.

Capability Length This field displays the length of the remaining fields in the Capability option. A constant value of 4 is used for all MBGP negotiations.

Address Family Identifier The *Address Family Identifier* (AFI) field encodes the type of network layer information that the peer would like to use during the session. Possible AFI values used by the JUNOS software include

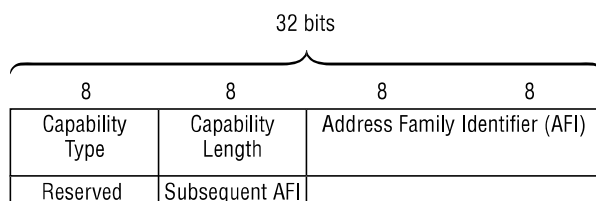
- 1—IPv4
- 2—IPv6
- 196—Layer 2 VPN

Reserved This field is not used and is set to a constant value of 0x00.

Subsequent Address Family Identifier The *Subsequent Address Family Identifier* (SAFI) field provides further information about the routing knowledge transmitted between the peers. The possible SAFI values used by the JUNOS software include the following:

- 1—Unicast
- 2—Multicast
- 4—Labeled unicast
- 128—Labeled VPN unicast
- 129—Labeled VPN multicast

FIGURE 5.15 MBGP capability negotiation format



Each set of routing information used by the router is uniquely described by both its AFI and SAFI codes. With the exception of IPv4 unicast routes, all other *Network Layer Reachability Information* (NLRI) is advertised and withdrawn using the MP-Reach-NLRI and MP-Unreach-NLRI BGP attributes. (We discuss the format of these attributes in Chapter 4, “Border Gateway Protocol (BGP).”) Let’s examine the possible advertised NLRI by seeing how each attribute is configured, negotiated, and stored by the router.

Internet Protocol Version 4

Routing knowledge transmitted using an AFI of 1 represent IPv4 routes. The NLRI sent in routing updates is a 32-bit value represented by a prefix and subnet mask. Depending on the SAFI value associated with the NLRI, it may contain special attributes or be used for a particular function.

IPv4 Unicast Routes

The SAFI of 1 implies that the IPv4 NLRI is a unicast route. Routes received with this SAFI are placed into the `inet.0` routing table and are used for forwarding user data traffic to the advertised NLRI. There is nothing really new or special about IPv4 unicast routes since they are the default set of knowledge advertised by the JUNOS software.

FIGURE 5.16 MBGP sample network

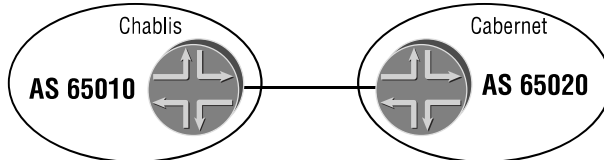


Figure 5.16 shows the Chablis router in AS 65010 and the Cabernet router in AS 65020. Chablis is configured for an EBGPeering session as so:

```
user@Chablis> show configuration protocols bgp
group external-peers {
  type external;
  neighbor 10.222.60.2 {
    peer-as 65020;
  }
}
```

We see the unicast IPv4 capability negotiation in the BGP Open messages sent by Chablis:

```
user@Chablis> monitor traffic interface fe-0/0/0 size 4096 detail
Listening on fe-0/0/0, capture size 4096 bytes
```

```
17:08:47.888135 Out IP (tos 0xc0, ttl 1, id 24676, len 97)
```

```

10.222.60.1.1147 > 10.222.60.2.bgp: P 1:46(45) ack 1 win 17376
<nop,nop,timestamp 110857412 110848025>: BGP, length: 45
  Open Message (1), length: 45
    Version 4, my AS 65010, Holdtime 90s, ID 192.168.52.1
    Optional parameters, length: 16
      Option Capabilities Advertisement (2), length: 6
        Multiprotocol Extensions, length: 4
          AFI IPv4 (1), SAFI Unicast (1)
      Option Capabilities Advertisement (2), length: 2
        Route Refresh (Cisco), length: 0
      Option Capabilities Advertisement (2), length: 2
        Route Refresh, length: 0

```

Once the session is established between the routers, we can see that any received NLRI for the session is placed into the `inet.0` routing table:

```

user@Chablis> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed  History  Damp State  Pending
inet.0                0          0           0         0         0         0
Peer           AS         InPkt   OutPkt   OutQ   Flaps Last Up/Dwn State
10.222.60.2    65020      5       10       0       2       13 0/0/0

```

IPv4 Multicast Routes

Within the context of MBGP, we often talk about sending multicast routes to a peer. Unfortunately, this name is a bit of a misnomer and can be misleading. In reality, what we are sending to the peer are IPv4 unicast routes to be used for a different purpose. MBGP multicast routes are used to perform reverse path forwarding (RPF) checks for received multicast data streams. The establishment of the forwarding tree and the sending of multicast traffic are handled by the Protocol Independent Multicast (PIM) configuration of the network. Within the JUNOS software, IPv4 routes received with a SAFI value of 2 (representing multicast) are placed into the `inet.2` routing table.

At the global, group, or neighbor level of the BGP configuration, the family `inet multicast` command allows the router to negotiate support for IPv4 routes with a SAFI of 2. Both the Chablis and Cabernet routers in Figure 5.16 have altered their configuration to only advertise multicast routes over their MBGP session. The configuration of the Chablis router now appears as so:

```

user@Chablis> show configuration protocols bgp
family inet {
  multicast;
}

```

```
group external-peers {
    type external;
    neighbor 10.222.60.2 {
        peer-as 65020;
    }
}
```

The appropriate AFI and SAFI values are transmitted in the Open messages sent by Chablis:

```
user@Chablis> monitor traffic interface fe-0/0/0 size 4096 detail
Listening on fe-0/0/0, capture size 4096 bytes
```

```
13:23:11.991401 Out_IP (tos 0xc0, ttl 1, id 24357, len 97)
10.222.60.1.3805 > 10.222.60.2.bgp: P 1:46(45) ack 1 win 17376
<nop,nop,timestamp 92224007 92215022>: BGP, length: 45
  Open Message (1), length: 45
    Version 4, my AS 65010, Holdtime 90s, ID 192.168.52.1
    Optional parameters, length: 16
      Option Capabilities Advertisement (2), length: 6
        Multiprotocol Extensions, length: 4
          AFI IPv4 (1), SAFI Multicast (2)
      Option Capabilities Advertisement (2), length: 2
        Route Refresh (Cisco), length: 0
      Option Capabilities Advertisement (2), length: 2
        Route Refresh, length: 0
```

The `inet.2` routing table is now used to store the NLRI received from Cabernet across the peering session:

```
user@Chablis> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table      Tot Paths  Act Paths  Suppressed  History Damp State  Pending
inet.2      0           0           0           0           0           0
Peer        AS      InPkt     OutPkt     OutQ    Flaps Last Up/Dwn State
|#Active/Received/Damped...
10.222.60.2 65020    7         9         0       1     1:25 0/0/0
           0/0/0
```



The default output of the `show bgp summary` command extends beyond the limit of an 80-character terminal screen. The second set of 0/0/0 routing information represents the `inet.2` routing table. When IPv4 multicast routes are sent in addition to other MBGP routes, the output of this command is altered for clarity. You can see this new format in the next section, “IPv4 Labeled Unicast Routes.”

We can view the `inet.2` information without wrapping the output by using the `| trim value` option. This option removes the number of columns specified in the `value` portion from the left side of the router output:

```
user@Chablis> show bgp summary | trim 25
eers: 0
Act Paths Suppressed History Damp State Pending
      0      0      0      0      0
InPkt OutPkt OutQ Flaps Last Up/Dwn State|#Active/Received/Damped...
  7      9      0      0      3:10 0/0/0      0/0/0
```

IPv4 Labeled Unicast Routes

Suppose you have an environment where two separate ASs are providing VPN services to their customers. At some point, one of these customers would like to set up a single VPN with multiple locations within each of the different AS networks. Several methods are available for configuring this type of setup, one of which includes the advertisement of IPv4 routes that are assigned an MPLS label. These routes represent the internally reachable addresses of each AS and allow for the establishment of a label-switched path across both domains.



The exact configuration and operation of this type of network is outside the scope of this book.

IPv4 labeled unicast routes are transmitted once you configure the peer routers with the family `inet labeled-unicast` command. Referring back to Figure 5.16, we see the configuration of the Chablis router is altered to support this NLRI:

```
user@Chablis> show configuration protocols bgp
family inet {
    labeled-unicast;
}
group external-peers {
    type external;
    neighbor 10.222.60.2 {
        peer-as 65020;
    }
}
```

IPv4 labeled unicast routes use a SAFI value of 4. We see this on Open messages sent by Cabinet and received on the Chablis router:

```
user@Chablis> monitor traffic interface fe-0/0/0 size 4096 detail
Listening on fe-0/0/0, capture size 4096 bytes
```

```

13:56:43.444993 In IP (tos 0xc0, ttl 1, id 18265, len 97)
10.222.60.2.bgp > 10.222.60.1.2093: P 1:46(45) ack 46 win 17331
<nop,nop,timestamp 92416161 92425150>: BGP, length: 45
  Open Message (1), length: 45
    Version 4, my AS 65020, Holdtime 90s, ID 192.168.48.1
    Optional parameters, length: 16
      Option Capabilities Advertisement (2), length: 6
        Multiprotocol Extensions, length: 4
          AFI IPv4 (1), SAFI labeled Unicast (4)
      Option Capabilities Advertisement (2), length: 2
        Route Refresh (Cisco), length: 0
      Option Capabilities Advertisement (2), length: 2
        Route Refresh, length: 0

```

The output of the `show bgp summary` command displays the `inet.0` routing table as the recipient of the labeled unicast NLRI. Since the routes are truly MBGP routes, the exact configuration of the output is modified. The State column now shows the Established state as `Establ` and the negotiated MBGP NLRI appears as separate routing tables below each peer's address:

```

user@Chablis> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths  Suppressed  History  Damp State  Pending
inet.0          0           0           0           0         0         0         0
Peer           AS         InPkt     OutPkt     OutQ     Flaps  Last Up/Dwn  State
10.222.60.2    65020      10        12         0         1       20  Establ
  inet.0: 0/0/0

```

IPv4 Labeled VPN Unicast Routes

When an Internet Service Provider (ISP) is providing a Layer 3 VPN service to its customers, it is actively participating in the routing domain of each customer. The active routes from one customer site are received on the near-end router, where they are advertised to the far end of the ISP network using MBGP. The far-end router then advertises these routes to the second customer site. For complete connectivity, the same process happens in reverse.



We discuss Layer 3 VPNs in further detail in Chapter 9.

The routes advertised between the near-end and far-end ISP routers contain attributes that provide separation between the ISP's customers. These attributes include BGP extended communities and MPLS labels. The peering session between these routers is established with the family `inet-vpn unicast` command and uses a SAFI value of 128. While the Chablis and

Cabernet routers in Figure 5.16 aren't within the same AS, we can use them to view the negotiation of the labeled VPN unicast routes. The configuration of the Chablis router is now

```
user@Chablis> show configuration protocols bgp
family inet-vpn {
    unicast;
}
group external-peers {
    type external;
    neighbor 10.222.60.2 {
        peer-as 65020;
    }
}
```

The BGP Open messages sent by Chablis show an AFI of 1 for IPv4 and a SAFI of 128 for labeled VPN unicast routes:

```
user@Chablis> monitor traffic interface fe-0/0/0 size 4096 detail
Listening on fe-0/0/0, capture size 4096 bytes

13:27:22.784757 Out IP (tos 0xc0, ttl 1, id 24386, len 97)
  10.222.60.1.2759 > 10.222.60.2.bgp: P 1:46(45) ack 1 win 17376
  <nop,nop,timestamp 92249086 92240100>: BGP, length: 45
    Open Message (1), length: 45
      Version 4, my AS 65010, Holdtime 90s, ID 192.168.52.1
      Optional parameters, length: 16
        Option Capabilities Advertisement (2), length: 6
          Multiprotocol Extensions, length: 4
            AFI IPv4 (1), SAFI labeled VPN Unicast (128)
          Option Capabilities Advertisement (2), length: 2
            Route Refresh (Cisco), length: 0
          Option Capabilities Advertisement (2), length: 2
            Route Refresh, length: 0
```

All received NLRI from Cabernet over this peering session is placed into the `bgp.13vpn.0` routing table:

```
user@Chablis> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths  Suppressed  History Damp State  Pending
bgp.13vpn.0          0           0           0           0           0           0
Peer           AS         InPkt     OutPkt     OutQ     Flaps Last Up/Dwn State
10.222.60.2    65020      5         7         0         1         11 Establ
bgp.13vpn.0: 0/0/0
```

IPv4 Labeled VPN Multicast Routes

Labeled VPN multicast routes are related to labeled VPN unicast routes in a manner similar to how IPv4 multicast and unicast routes are related. The labeled VPN multicast routes are actually IPv4 NLRI with extended communities and MPLS labels attached to associate them with a specific customer VPN. They are placed into a separate routing table, where they are used to perform multicast RPF checks. This NLRI uses a SAFI value of 129 and is configured with the family `inet-vpn multicast` command at the global, group, or neighbor level of the BGP hierarchy.

We can once again modify the configuration of the routers in Figure 5.16 to view the negotiation of this NLRI. The Chablis router is now configured to support labeled VPN multicast routes:

```
user@Chablis> show configuration protocols bgp
family inet-vpn {
    multicast;
}
group external-peers {
    type external;
    neighbor 10.222.60.2 {
        peer-as 65020;
    }
}
```

Chablis advertises this capability in the Open messages sent to Cabernet:

```
user@Chablis> monitor traffic interface fe-0/0/0 size 4096 detail
Listening on fe-0/0/0, capture size 4096 bytes

13:31:52.446070 Out IP (tos 0xc0, ttl 1, id 24420, len 97)
10.222.60.1.2077 > 10.222.60.2.bgp: P 1:46(45) ack 1 win 17376
<nop,nop,timestamp 92276052 92267066>: BGP, length: 45
  Open Message (1), length: 45
    Version 4, my AS 65010, Holdtime 90s, ID 192.168.52.1
    Optional parameters, length: 16
      Option Capabilities Advertisement (2), length: 6
        Multiprotocol Extensions, length: 4
          AFI IPv4 (1), SAFI labeled VPN Multicast (129)
      Option Capabilities Advertisement (2), length: 2
        Route Refresh (Cisco), length: 0
      Option Capabilities Advertisement (2), length: 2
        Route Refresh, length: 0
```

As you would expect, the JUNOS software maintains a separate routing table for all received labeled VPN multicast routes. The output of the `show bgp summary` command on Chablis reveals that the `bgp.13vpn.2` routing table is used for this purpose:

```
user@Chablis> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths Suppressed  History Damp State  Pending
bgp.13vpn.2          0          0          0          0          0          0
Peer           AS        InPkt   OutPkt   OutQ   Flaps Last Up/Dwn State
10.222.60.2    65020     8       10      0      1      24 Establ
  bgp.13vpn.2: 0/0/0
```

Layer 2 Virtual Private Networks

The AFI of 196 represents reachability knowledge used in a Layer 2 VPN environment. Unlike its counterparts in IPv4 and IPv6, the NLRI for the Layer 2 VPN AFI is not an actual route used for forwarding. In fact, it isn't even an IP route at all. Instead, it is information concerning the Layer 2 logical circuit information used to connect the customer to the ISP network. While this sounds a bit strange at first, it makes a little more sense when we describe it in some context.

We saw in the “IPv4 Labeled VPN Unicast Routes” section earlier that an ISP providing a Layer 3 VPN service actively participates in the routing domain of the customer. This active participation does not occur in a Layer 2 VPN environment. The ISP in this configuration is simply providing a logical circuit between the customer end points. This circuit, in turn, is used by the customer to route its own traffic across the ISP network. The routers at the edge of the ISP network simply transmit circuit information and MPLS label information to each other. The peering session between the ISP edge routers is established using the `family 12vpn unicast` command using a SAFI value of 128.



We also discuss Layer 2 VPNs in further detail in Chapter 9.

Using Figure 5.16 as a guide, we once again update the configuration of the Chablis and Cabernet routers to view the establishment of their BGP session. In this scenario, the configuration of the Chablis now appears as so:

```
user@Chablis> show configuration protocols bgp
family 12vpn {
  unicast;
}
group external-peers {
  type external;
  neighbor 10.222.60.2 {
    peer-as 65020;
  }
}
```




Real World Scenario

Advertising Multiple Address Families

Throughout this section we've been configuring our BGP peers to advertise reachability information for a specific AFI/SAFI. While this is good for learning purposes, it's not exactly realistic for the real world. A very common application of MBGP is two IBGP peers in an AS supporting transit service, Layer 3 VPNs, and Layer 2 VPNs. Let's see how this configuration and session negotiation works.

Our sample routers of Chablis and Cabernet are now configured as IBGP peers within AS 65010. We've updated their peering session to advertise multiple NLRI using the following configuration:

```
user@Chablis> show configuration protocols bgp
group internal-peers {
    type internal;
    local-address 192.168.52.1;
    family inet {
        unicast;
    }
    family inet-vpn {
        unicast;
    }
    family l2vpn {
        unicast;
    }
    neighbor 192.168.48.1;
}
```

Each of the configured AFI/SAFI combinations is advertised separately in the BGP Open message sent by the Chablis router to its IBGP peer:

```
user@Chablis> monitor traffic interface fe-0/0/0 size 4096 detail
Listening on fe-0/0/0, capture size 4096 bytes

01:38:23.542234 Out IP (tos 0xc0, ttl 64, id 26978, len 113)
  192.168.52.1.3774 > 192.168.48.1.bgp: P 1:62(61) ack 1 win 16500
  <nop,nop,timestamp 113914947 113905494>: BGP, length: 61
    Open Message (1), length: 61
      Version 4, my AS 65010, Holdtime 90s, ID 192.168.52.1
      Optional parameters, length: 32
        Option Capabilities Advertisement (2), length: 6
          Multiprotocol Extensions, length: 4
            AFI IPv4 (1), SAFI Unicast (1)
```

```

Option Capabilities Advertisement (2), length: 6
  Multiprotocol Extensions, length: 4
    AFI IPv4 (1), SAFI labeled VPN Unicast (128)
Option Capabilities Advertisement (2), length: 6
  Multiprotocol Extensions, length: 4
    AFI Layer-2 VPN (196), SAFI labeled VPN Unicast (128)
Option Capabilities Advertisement (2), length: 2
  Route Refresh (Cisco), length: 0
Option Capabilities Advertisement (2), length: 2
  Route Refresh, length: 0

```

After the peering session reaches the Established state, we see the various routing tables used to store received NLRI from the remote peer:

```

user@Chablis> show bgp summary
Groups: 1 Peers: 1 Down peers: 0
Table          Tot Paths  Act Paths  Suppressed    History Damp State   Pending
inet.0                0          0          0             0          0          0
bgp.13vpn.0           0          0          0             0          0          0
bgp.12vpn.0           0          0          0             0          0          0
Peer           AS         InPkt     OutPkt     OutQ    Flaps  Last Up/Dwn  State
192.168.48.1  65010      19        21         0       1      3:33 Establ
inet.0: 0/0/0
bgp.13vpn.0: 0/0/0
bgp.12vpn.0: 0/0/0

```

When we view the BGP Open messages sent by Chablis to Cabernet, we see an AFI of 196 representing the Layer 2 VPN and a SAFI of 128 for labeled VPN unicast routes:

```

user@Chablis> monitor traffic interface fe-0/0/0 size 4096 detail
Listening on fe-0/0/0, capture size 4096 bytes

13:51:10.408862 Out IP (tos 0xc0, ttl 1, id 24577, len 97)
10.222.60.1.4034 > 10.222.60.2.bgp: P 1:46(45) ack 1 win 17376
<nop,nop,timestamp 92391847 92382858>: BGP, length: 45
  Open Message (1), length: 45
    Version 4, my AS 65010, Holdtime 90s, ID 192.168.52.1
    Optional parameters, length: 16
      Option Capabilities Advertisement (2), length: 6

```

```

Multiprotocol Extensions, length: 4
  AFI Layer-2 VPN (196), SAFI labeled VPN Unicast (128)
Option Capabilities Advertisement (2), length: 2
Route Refresh (Cisco), length: 0
Option Capabilities Advertisement (2), length: 2
Route Refresh, length: 0

```

The NLRI received by Chablis across this peering session is placed into the `bgp.12vpn.0` routing table:

```
user@Chablis> show bgp summary
```

```
Groups: 1 Peers: 1 Down peers: 0
```

Table	Tot Paths	Act Paths	Suppressed	History	Damp	State	Pending
<u>bgp.12vpn.0</u>	0	0	0	0	0	0	0
Peer	AS	InPkt	OutPkt	OutQ	Flaps	Last Up/Dwn	State
10.222.60.2	65020	7	9	0	1	15	<u>Establ</u>
bgp.12vpn.0: 0/0/0							

Summary

In this chapter, we saw the various methods available within the JUNOS software for modifying the BGP attributes. The Origin attribute was altered using a routing policy as routes were advertised to a peer. We then saw how both configuration options and routing policies affected the AS Path attribute. The AS Path was prepended using our local as well as our customer's AS value. The attribute also had values removed or modified using configuration options such as `remove-private` and `as-override`. We then discussed the Multiple Exit Discriminator and how to set the MED value using a policy and a configuration knob. We also saw two different methods for associating the advertised MED to the internal IGP metric in our AS. Finally, methods for altering how the JUNOS software evaluates and uses the MED attribute were discussed. We concluded our attribute discussion by using a routing policy and a configuration option to change the Local Preference attribute before advertising a route to a peer.

We then explored two different methods for scaling a large IBGP full mesh of routers. The first option was route reflection, which allows an IBGP-learned route to be readvertised to another IBGP peer. The router responsible for this was the route reflector within a cluster. The second method of scaling an IBGP network is confederations. A BGP confederation network breaks the AS into smaller member AS networks, or sub-AS networks. Within each sub-AS an IBGP full mesh was still required, but each sub-AS was connected using an EBGP-like connection known as confederation BGP.

We concluded the chapter with a discussion on Multiprotocol BGP (MBGP). We examined some reasons for using MBGP inside a local AS or between multiple ASs. The configuration and verification of each unique AFI/SAFI set was explored.

Exam Essentials

Be able to list the configuration options available for modifying the AS Path attribute. The JUNOS software provides four different configuration options related to the AS Path attribute. The `remove-private` command selectively removes private AS values in the path before advertising a route. The `local-as` command allows a BGP router to establish a session using an AS value other than the value configured within the `[edit routing-options hierarchy]`. The `as-override` command removes instances in the AS Path of an EBGp peer and replaces them with the local AS value. This is used when a network is providing a VPN-like service to a customer. The final configuration option, `loops`, also assists in a VPN-like service offering. This command, however, allows multiple instances of the local AS to appear in the path.

Be familiar methods available for using the MED attribute in path selections. The default behavior of the JUNOS software is to group routes received from the same AS together and compare their attached MED values. This is known as deterministic MED evaluation. You have the option of allowing the router to always compare the MED values, regardless of the neighboring AS that advertised the route. In addition, you can mimic the default behavior of the Cisco Systems MED operation, which evaluates routes based on when the local router received them from a peer.

Be able to describe the two methods for altering the Local Preference attribute. The JUNOS software provides the `local-preference` configuration option to set the attribute value on all advertised BGP routes. In addition, the `local-preference` keyword is used as a routing policy action to change the Local Preference value. The policy action can be used as routes are advertised to a peer in an export policy. More commonly, the attribute value is changed by an import routing policy.

Be able to describe the operation of a BGP route reflection network. The use of route reflection within an AS allows a router called the route reflector to send IBGP-learned routes to other IBGP peers. Each route reflector is assigned clients within a cluster that it is responsible for. Routing loops are avoided through the addition of two new BGP attributes: the Originator ID and the Cluster List. Each route readvertised by the route reflector has the Cluster List attribute modified with the local cluster ID value. Any received route that already contains the local cluster ID is dropped.

Be able to describe the operation of a BGP confederation network. A BGP network using confederations reduces the problem of the IBGP full mesh into smaller, more manageable groups of routers. Each group of routers is called a sub-AS and receives its own unique sub-AS number from the private AS range. Within each sub-AS, the IBGP full mesh is maintained. Each sub-AS is connected through a CBGP peering session that modifies the AS Path attribute for loop prevention. The information included in the AS Path is either an AS Confederation Sequence or an AS Confederation Set.

Be able to configure Multiprotocol BGP. The configuration of MBGP occurs at the global, group, or neighbor level when the `family` command is used. This command requires the addition of various keywords to uniquely describe the AFI/SAFI being negotiated. For example, `family inet multicast` enables the advertisement and receipt of IPv4 multicast routes.

Review Questions

1. Which routing policy action sets the Origin attribute to its worst possible value?
 - A. `then origin igp`
 - B. `then origin egp`
 - C. `then origin incomplete`
 - D. `then origin unknown`
2. Your local AS value is 1234. Instead of sending your EBGP peer an AS Path of 1234 64678 4321, you want to send a path of 1234 4321. What JUNOS software command accomplishes this?
 - A. `as-override`
 - B. `as-loops`
 - C. `local-as`
 - D. `remove-private`
3. Which statement best describes the default operation of the JUNOS software in relation to using the MED value on a BGP route?
 - A. The routes are grouped by neighboring AS, and the MED is compared against routes in each group.
 - B. The routes are combined together regardless of the neighboring AS, and the MED is compared against all routes.
 - C. The MED values of the routes are compared as they were received in an oldest-to-youngest fashion.
 - D. The MED values of the routes are compared as they were received in a youngest-to-oldest fashion.
4. The AS value assigned to your AS is 5432. Which routing policy action results in an advertised AS Path of 5432 5432 5432 1234 6789?
 - A. `then as-path-prepend 2`
 - B. `then as-path-prepend 3`
 - C. `then as-path-prepend "5432 5432"`
 - D. `then as-path-prepend "5432 5432 5432"`
5. What value identifies a grouping of a BGP route reflector(s) and its clients within an AS?
 - A. Cluster ID
 - B. Originator ID
 - C. Router ID
 - D. Peer ID

6. Which BGP attribute is modified by a route reflector to signify that the route has been readvertised within the IBGP network?
 - A. Cluster ID
 - B. Cluster List
 - C. Originator ID
 - D. Router ID

7. What type of route reflection design is used when the route reflector full mesh grows excessively large?
 - A. Basic route reflection
 - B. Hierarchical route reflection
 - C. Two route reflectors in a single cluster
 - D. Fully meshed route reflection clients

8. In a BGP confederation network, what type of peering session is used between each sub-AS?
 - A. IBGP
 - B. CBGP
 - C. EBGP
 - D. MBGP

9. What BGP attribute is modified, by default, when a route is advertised between sub-AS networks?
 - A. Next Hop
 - B. Local Preference
 - C. AS Path
 - D. Multiple Exit Discriminator

10. Which form of BGP allows for the use of reachability information that is not an IPv4 unicast route?
 - A. IBGP
 - B. CBGP
 - C. EBGP
 - D. MBGP