Chapter 1

Network Discovery and Verification

JNCIE LAB SKILLS COVERED IN THIS CHAPTER:

- ✓ Verify Out of Band (OoB) management network
- ✓ Discover and verify IGP topology and route redistribution
- ✓ Discover and verify IBGP topology
- ✓ Discover and verify EBGP topology and routing policy

In this chapter, you will be exposed to a series of network discovery and verification tasks that are indicative of those typically encountered at the beginning of the JNCIE examination. While the ability to reverse engineer an unfamiliar network is an invaluable skill that all experts should possess, the primary purpose of the discovery scenario is to allow the candidate to "become one" with the provided baseline network topology before the candidate is expected to begin modifying it during the course of subsequent configuration scenarios.

Because the JNCIE examination focuses on advanced topics such as Multi Protocol Label based Switching (MPLS), firewall filters, and VPNs, the JNCIE examination begins with a preconfigured network with regards to the OoB management network, user accounts, interface configuration, interior gateway protocol (IGP) configuration, Internal and External Border Gateway Protocol (IBGP/EBGP) configuration, and a simple set of redistribution and IBGP/EBGP policies. Though spared the need to actually configure this baseline functionality, the JNCIE candidate begins the examination by discovering the initial network topology and by verifying the overall operation of this baseline network. Because the emphasis of the JNCIE is on higher-level applications and services, the candidate might assume that their interfaces are properly configured and operational. While you will likely find nothing "fancy" about your interface configurations, it is suggested that you give the interface portion of each router's configuration a quick glance; the memory of a non-default logical unit or Virtual Router Redundancy Protocol (VRRP) group configuration may come back to prevent you from making a mistake in a later configuration task.

Although candidates are never intentionally provided with faulty equipment, you should be on guard for any symptoms of hardware malfunction during the network discovery task. In some cases, you may find that the provided configurations require some tweaking. Some versions of the JNCIE examination might require that the candidate perform fault isolation and take corrective actions during the discovery scenario.

Two sets of complete baseline configurations for all routers in the test bed are provided in this chapter. It is suggested that you load your test bed with the same baseline configuration as called out in each chapter to maximize your ability to follow along with each chapter's configuration scenarios.

To kick things off, you will need to access the routers in the test bed either using terminal server–based console attachment, or through the preconfigured Out of Band (OoB) network. Once connected, you can begin to reverse engineer and become one with your new network!

Task 1: Verify OoB Network

As described in the introduction, your JNCIE test bed consists of seven M-series routers, a terminal server, and a 100Mbps Fast Ethernet LAN segment that will serve as your network's

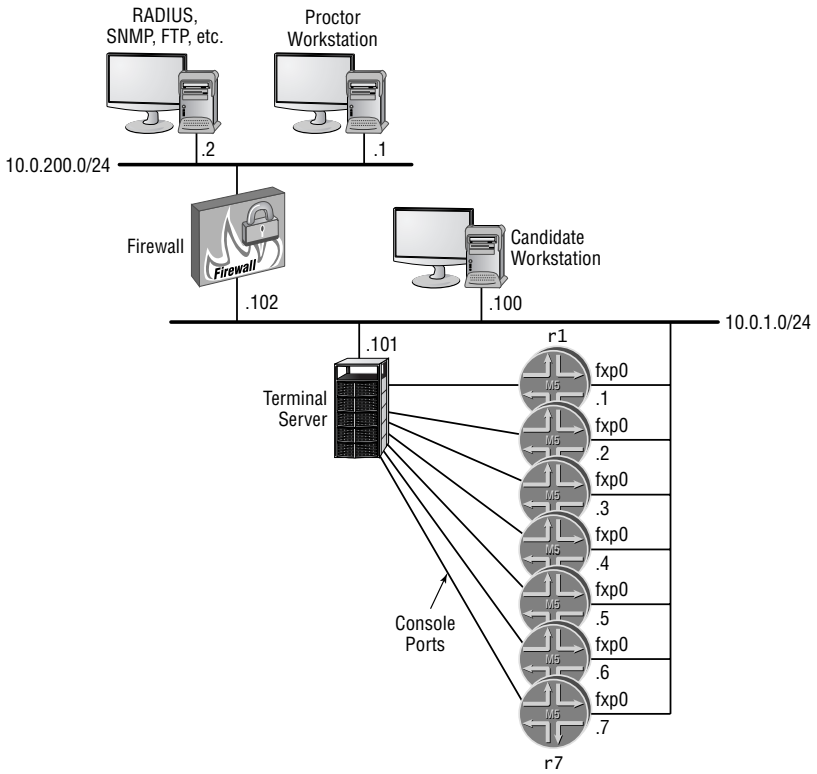
OoB management network. You will likely find that there is no need for terminal server–based attachment because the baseline network has a preconfigured OoB network.

Although you can use the router console ports for the JNCIE examination, most candidates find that it saves time to open multiple telnet sessions (one per router) using the OoB management network that is configured during the examination. You should use the terminal server whenever you are performing router maintenance (such as upgrading JUNOS software), or when routing problems cause telnet access problems.

The OoB Topology

The Out of Band management topology is illustrated in Figure 1.1. Based on this figure, you can see that the IP address of the terminal server is 10.0.1.101, and that its asynchronous interfaces are connected in ascending order to the console ports of each router that is associated with your test bed. The preconfigured fxp0 addressing is also shown in the figure.

FIGURE 1.1 The Out of Band (OoB) management network



The testing center will provide you with both user EXEC and privileged EXEC mode passwords for the terminal server (or their equivalents, should a non–Internetwork Operating System (IOS) based terminal server be in use). This chapter will focus on fxp0-based router access; please see the *JNCIP Study Guide* (Sybex, 2003) for a detailed discussion of terminal server usage.

Accessing Routers Using Telnet

Using the addressing shown earlier in Figure 1.1 and the predefined user account information provided in Table 1.1, verify that you can open a telnet connection to each router using the `lab` login. (A *`root`* login requires terminal server–based console attachment because secure shell [SSH] is not enabled by default).

TABLE 1.1 User Account Parameters

User	Password	Class/Permission	Notes
`root`	`root`	superuser	RADIUS/local password with automatic login in the event of RADIUS failure RADIUS secret is *`juniper`*
`lab`	`lab`	superuser	Same as for user `root`

A successful telnet session will be similar to this capture, which shows the telnet session to `r1` being successfully established:

```
r1 (ttyp1)

login: lab
Password:
Last login: Wed Feb  5 02:44:47 from 10.0.1.100

--- JUNOS 5.6R1.3 built 2003-01-02 20:38:33 UTC

lab@r1>
```

After opening telnet sessions to all seven routers, you quickly confirm the static routing needed to reach the proctor subnet and RADIUS/FTP server by performing some ping testing:

```
lab@r1> ping 10.0.200.2
PING 10.0.200.2 (10.0.200.2): 56 data bytes
64 bytes from 10.0.200.2: icmp_seq=0 ttl=255 time=1.228 ms
64 bytes from 10.0.200.2: icmp_seq=1 ttl=255 time=0.701 ms
^C
--- 10.0.200.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.701/0.964/1.228/0.264 ms
```

Verification of the OoB network is complete once you open telnet sessions to all seven routers and verify that each can ping the proctor workstation.

Task 2: Discover and Verify IGP Topology and Route Redistribution

Your next goal is to discover the provided IGP topology, and to verify that there are no operational issues in the core IGP, or in the operation of any IGP route redistribution that may be going on. Figure 1.2 details the JNCIE test bed topology that has been provided in this example. It is suggested that you mark up a copy of the network topology as you proceed in your discovery to assist you in later configuration scenarios.

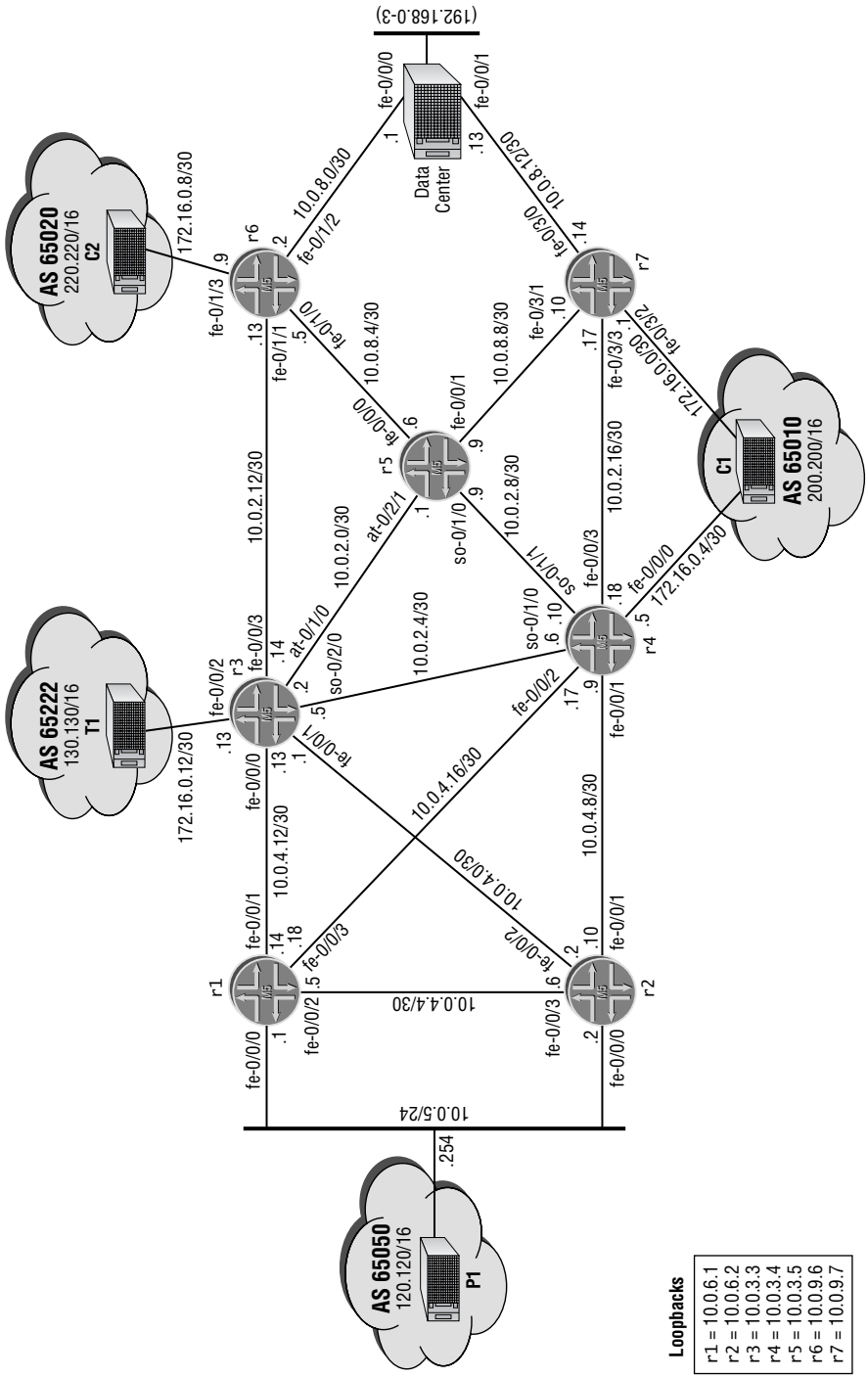
Using the IGP Operation to Verify Interface Operation

Because your IGP relies on proper interface configuration and operation, you can effectively kill two birds with one stone by starting your verification with your IGP. Proper interface operation is effectively confirmed when you have all expected IGP adjacencies and IGP routes, and when traceroute testing confirms optimal paths through the network. You should confirm interface operation when IGP problems are detected even though the IGP's configuration seems correct. It is also a good idea to note any non-default logical units in place for future reference as the JNCIE examination progresses. Note that for the IS-IS routing protocol, proper adjacency formation can occur even if errors are present in the IP configuration of the corresponding interface. Newer versions of JUNOS software, such as the 5.6 release used as the basis for this book, will not form an IS-IS adjacency when IP parameters are mismatched, as reflected in the trace output shown here:

```
lab@r2# run show log isis
Mar  5 08:04:13 Received L1 LAN IIH, source id r1 on fe-0/0/3.0
Mar  5 08:04:13     intf index 5, snpa 0:a0:c9:6f:7b:84
Mar  5 08:04:13     max area 0, circuit type l1, packet length 56
Mar  5 08:04:13     hold time 9, priority 64, circuit id r1.03
Mar  5 08:04:13     neighbor 0:a0:c9:6f:70:d (ourselves)
Mar  5 08:04:13     speaks IP
Mar  5 08:04:13     speaks IPV6
Mar  5 08:04:13     IP address 10.0.6.1
Mar  5 08:04:13     area address 49.0001 (3)
Mar  5 08:04:13     restart RR reset RA reset holdtime 0
Mar  5 08:04:13 ERROR: IIH from r1 without matching addresses,
   interface fe-0/0/3.0
```

The tracing output in this example was obtained at r2 after the IP address was removed from r1's fe-0/0/2 interface.

FIGURE 1.2 The JNCIE test bed topology



The reader who is familiar with the previous book in this series should immediately recognize numerous similarities between the JNCIP and JNCIE topologies. This level of similarity may or may not occur in the actual JNCIE examination, which is why the candidate begins the examination with a discovery scenario designed to familiarize the candidate with their "new" topology.

Figure 1.2 (shown earlier) holds a wealth of information about your test bed. From the figure, you can see that you have a mix of EBGP peers, and that route redistribution will likely be in place between `r6`, `r7`, and the data center routers. You will also note that your test bed once again consists of a mix of interface types, including Fast Ethernet, OC-3c POS, and ATM.

Discovering and Verifying Core IGP

While there are many ways in which a candidate might decide to attack the discovery of their network's IGP, this author has chosen to begin with `r3`, `r4`, and `r7`, because their central placement implies that much can be learned by examining the configuration (and operation) of their IGP. You take a deep breath and begin by displaying `r3`'s protocol stanza:

```
[edit]
lab@r3# show protocols
bgp {
    advertise-inactive;
    group int {
        type internal;
        local-address 10.0.3.3;
        export nhs;
        neighbor 10.0.6.1;
        neighbor 10.0.6.2;
        neighbor 10.0.3.4;
        neighbor 10.0.3.5;
        neighbor 10.0.9.6;
        neighbor 10.0.9.7;
    }
    group ext {
        import ebgp-in;
        export ebgp-out;
        neighbor 172.16.0.14 {
            peer-as 65222;
        }
    }
}
ospf {
    area 0.0.0.1 {
        stub default-metric 10;
```

```
        interface fe-0/0/0.0;
        interface fe-0/0/1.0;
    }
    area 0.0.0.0 {
        interface so-0/2/0.100;
        interface at-0/1/0.0;
    }
    area 0.0.0.2 {
        nssa;
        interface fe-0/0/3.0;
    }
}
```

The highlighted portion relating to r3's IGP configuration is the area of concern at this stage. From r3's IGP configuration, you can determine the following:

- The core IGP is OSPF (Open Shortest Path First).
- r3 is an Area Border Router (ABR) for areas 1 and 2.
- Area 1 is a stub network and r3 is configured to generate a default route into that area.
- Area 2 is configured as a NSSA (not-so-stubby area) network. No default route is generated by r3 into area 2.
- No address aggregation or restriction of summary LSAs is occurring at r3.
- OSPF authentication is not configured in areas 0, 1, and 2.
- r3 will run OSPF on all interfaces in the baseline topology, except its lo0 and external fe-0/0/2 interfaces.

The omission of the router's lo0 interface from the area declarations results in advertisement of the router's loopback address (the lo0 interface is the default source of the RID) in the router LSAs injected into all areas. Although not shown here, the OSPF configuration for r4 is virtually identical to that of r3. Now that you have some idea of what to expect, it makes sense to quickly assess the state of r3's adjacencies:

```
[edit]
lab@r3# run show ospf neighbor
  Address          Interface              State      ID               Pri  Dead
10.0.2.1         at-0/1/0.0             Full       10.0.3.5         128   36
10.0.2.6         so-0/2/0.100           Full       10.0.3.4         128   34
10.0.4.14        fe-0/0/0.0             Full       10.0.6.1         128   32
10.0.4.2         fe-0/0/1.0             Full       10.0.6.2         128   31
10.0.2.13        fe-0/0/3.0             Full       10.0.9.6         128   39
```

The results confirm that all five of r3's adjacencies have been correctly established. A quick look at r4's adjacencies confirms that it too has the five adjacencies one would expect,

given this topology:

```
[edit]
lab@r4# run show ospf neighbor
  Address          Interface              State      ID               Pri  Dead
10.0.2.5         so-0/1/0.100           Full       10.0.3.3         128   34
10.0.2.9         so-0/1/1.0             Full       10.0.3.5         128   39
10.0.4.10        fe-0/0/1.0             Full       10.0.6.2         128   35
10.0.4.18        fe-0/0/2.0             Full       10.0.6.1         128   35
10.0.2.17        fe-0/0/3.0             Full       10.0.9.7         128   32
```

You now quickly examine the OSPF configuration for r1 and r2:

```
[edit]
lab@r1# show protocols ospf
area 0.0.0.1 {
    stub;
    interface fe-0/0/0.0 {
        passive;
    }
    interface fe-0/0/1.0;
    interface fe-0/0/2.0;
    interface fe-0/0/3.0;
}
```

r1's configuration allows you to determine that it is configured to run a passive OSPF instance on its fe-0/0/0 interface, and that its overall configuration is commensurate with the stub area settings discovered in r3. The passive setting on its fe-0/0/0 interface will prevent adjacency formation on the P1 peering subnet, while allowing the 10.0.5/24 prefix to be carried as an OSPF internal route. With r2's configuration being virtually identical (not shown), you expect to see three OSPF adjacencies in place on both r1 and r2:

```
lab@r1# run show ospf neighbor
  Address          Interface              State      ID               Pri  Dead
10.0.4.13        fe-0/0/1.0             Full       10.0.3.3         128   34
10.0.4.6         fe-0/0/2.0             Full       10.0.6.2         128   35
10.0.4.17        fe-0/0/3.0             Full       10.0.3.4         128   34
```

As anticipated, r1 has the correct number of adjacent neighbors. With area 1 configured as a stub area, there should be no external routes in r1's link state database:

```
[edit]
lab@r2# run show ospf database extern
```

Because network summaries (LSA Type 3s) are not being filtered at the ABR (r3), you expect to see OSPF routes to the loopback addresses of all routers making up your test bed. Some

creative CLI (command-line interface) work makes this determination a snap:

```
[edit]
lab@r2# run show route protocol ospf | match /32
10.0.3.3/32        *[OSPF/10] 00:12:37, metric 1
10.0.3.4/32        *[OSPF/10] 01:52:14, metric 1
10.0.3.5/32        *[OSPF/10] 00:12:37, metric 2
10.0.6.1/32        *[OSPF/10] 01:52:14, metric 1
10.0.9.6/32        *[OSPF/10] 00:12:37, metric 2
10.0.9.7/32        *[OSPF/10] 01:52:14, metric 2
224.0.0.5/32       *[OSPF/10] 03:32:36, metric 1
```

The highlighted output generated by r2 confirms that the loopback addresses of the other six routers are being learned through the OSPF protocol. As a final check, you confirm the presence of a default route in accordance with the configuration found on ABR r3:

```
[edit]
lab@r2# run show route

inet.0: 118111 destinations, 118118 routes (118111 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[OSPF/10] 00:47:14, metric 11
                    > to 10.0.4.9 via fe-0/0/1.0
                      to 10.0.4.1 via fe-0/0/2.0
. . .
```

The default route is present in area 1, and the two viable next hops listed indicate that both r3 and r4 are sourcing a default route into the stub area. So far, things are looking pretty good for the operation of the test bed's IGP!

Discovering and Verifying IGP Redistribution

Having confirmed the overall operation of the OSPF protocol for r1 through r4, you next examine the OSPF configuration at r5:

```
[edit]
lab@r5# show protocols ospf
area 0.0.0.0 {
    interface at-0/2/1.0;
    interface so-0/1/0.0;
}
area 0.0.0.2 {
    nssa;
    interface fe-0/0/0.0;
```

```
    interface fe-0/0/1.0;
}
```

The output indicates that r5 is an ABR interconnecting area 2 and the backbone. You also note that, like r3, r5 considers area 2 to be a NSSA. The lack of the default metric keyword indicates that r5 will not generate a default route into area 2. With the same finding made at r3 and r4, you conclude that the NSSA will not have an OSPF derived default route. You quickly confirm your suspicions regarding the absence of a default route in area 2 using the following command on r6:

```
[edit]
lab@r6# run show route | match 0.0.0.0/0

[edit]
lab@r6#
```

Considering that nothing you have uncovered so far can be considered "broken," you simply note the lack of a default route in the NSSA, and you move on with your discovery task.

Why Is There No Default Route in the NSSA?

You may find it odd that none of area 2's ABRs are configured to generate a default route into the NSSA. Because network summaries are permitted in the NSSA, and because there are no OSPF AS-externals (LSA Type 5s) being generated in areas 0 or 1, the lack of a default route in the NSSA may not represent a problem. If all external routing information associated with areas 0 and 1 is carried in BGP, for example, the routers in area 2 should not have trouble reaching external destinations.

You expect to find four OSPF adjacencies in place at r5. The output from r5 quickly confirms your expectations on this front:

```
[edit]
lab@r5# run show ospf neighbor
  Address          Interface              State       ID               Pri  Dead
10.0.2.2           at-0/2/1.0             Full        10.0.3.3         128    32
10.0.2.10          so-0/1/0.0             Full        10.0.3.4         128    38
10.0.8.5           fe-0/0/0.0             Full        10.0.9.6         128    39
10.0.8.10          fe-0/0/1.0             Full        10.0.9.7         128    37
```

With r5's IGP configuration analyzed, you move on to r7 to inspect its IGP configuration:

```
[edit]
lab@r7# show interfaces
fe-0/3/0 {
    unit 0 {
```

```
        family inet {
            address 10.0.8.14/30;
        }
        family iso;
    }
}
fe-0/3/1 {
    unit 0 {
        family inet {
            address 10.0.8.10/30;
        }
    }
}
fe-0/3/2 {
    unit 0 {
        family inet {
            address 172.16.0.1/30;
        }
    }
}
fe-0/3/3 {
    unit 0 {
        family inet {
            address 10.0.2.17/30;
        }
    }
}
fxp0 {
    unit 0 {
        family inet {
            address 10.0.1.7/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.0.9.7/32;
        }
        family iso {
            address 49.0002.7777.7777.7777.00;
```

```
        }
    }
}
[edit]
lab@r7# show protocols
bgp {
    group int {
        type internal;
        local-address 10.0.9.7;
        export nhs;
        neighbor 10.0.6.1;
        neighbor 10.0.6.2;
        neighbor 10.0.3.3;
        neighbor 10.0.3.4;
        neighbor 10.0.3.5;
        neighbor 10.0.9.6;
    }
    group c1 {
        type external;
        export ebgp-out;
        neighbor 172.16.0.2 {
            peer-as 65010;
        }
    }
}
isis {
    export ospf-isis;
    level 2 disable;
    level 1 external-preference 149;
    interface fe-0/3/0.0;
    interface lo0.0;
}
ospf {
    export isis-ospf;
    area 0.0.0.2 {
        nssa;
        interface fe/0/3/1.0;
        interface fe-0/3/0.0 {
            passive;
        }
```

```
        interface fe-0/3/3.0;
    }
}
```

Once again, the IGP related portions of the router's configuration are called out with highlights. Though not shown here, the configuration of `r6` is virtually identical to that shown for `r7`. The presence of both OSPF and IS-IS stanzas tells you that `r7` is most likely acting as a redistribution source for the 192.168.0/22 routes associated with the data center. You also note the following:

- `r7` is configured to operate as a Level 1 IS-IS router on its fe-0/3/0 interface, which implies that the DC router is running IS-IS Level 1.
- The global preference for IS-IS Level 1 external routes has been modified to ensure that the IS-IS routes are preferred over their OSPF equivalents when they are redistributed into OSPF as NSSA externals, which have a default preference of 150.
- `r7` has been set to run a passive OSPF instance on its fe-0/3/0 interface; this will result in advertisement of the 10.0.8.12/30 subnet as an OSPF internal route while also guarding against unwanted OSPF adjacencies to the DC router.
- Export policy is in place for both the OSPF and IS-IS IGPs.

You start by quickly accessing the state of IGP adjacencies at `r6` or `r7`; based on the configuration displayed, you expect a total of three adjacencies, two of which should be OSPF and one that is IS-IS Level 1:

```
[edit]
lab@r6# run show ospf neighbor
  Address         Interface             State      ID              Pri   Dead
10.0.8.6          fe-0/1/0.0            Full       10.0.3.5        128   36
10.0.2.14         fe-0/1/1.0            Full       10.0.3.3        128   32
```

The display confirms the expected number of OSPF adjacencies at `r6`. You next confirm its IS-IS adjacency status:

```
[edit]
lab@r6# run show isis adjacency
Interface       System         L State        Hold (secs)  SNPA
fe-0/1/2.0          dc            1 Up                   7  0:a0:c9:69:c5:27
```

Excellent! All expected IGP adjacencies are established at `r6`. You now display the *`ospf-isis`* export policy to get a handle on what routes should be redistributed from OSPF to the DC router:

```
[edit]
lab@r6# show policy-options policy-statement ospf-isis
term 1 {
    from {
        protocol ospf;
        route-filter 0.0.0.0/0 exact;
    }
```

```
    then accept;
}
```

The *ospf-isis* export policy is designed to redistribute a default route from OSPF into IS-IS. Most likely, this default route is intended to provide the data center router with reachability to internal and external prefixes, as it is assumed that a DC router will not be running BGP. But wait—a previous test confirmed that there was no default route in area 2! You quickly re-verify that no OSPF-based default route exists at r6:

```
[edit]
lab@r6# run show route protocol ospf | match 0.0.0.0
```

No default route, OSPF or otherwise. This makes the *ospf-isis* policy more than a bit moot, and this may represent an operational problem. A quick telnet hop to the DC router confirms the magnitude of the situation:

```
lab@dc> show route protocol isis

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

10.0.9.6/32        *[IS-IS/15] 02:00:35, metric 10
                    > to 10.0.8.2 via fe-0/0/0.0
10.0.9.7/32        *[IS-IS/15] 00:49:24, metric 10
                    > to 10.0.8.14 via fe-0/0/1.0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

The output confirms that the only IS-IS routes being advertised to the data center router from r6 and r7 are the prefixes associated with their loopback addresses. Further testing confirms serious reachability problems at the data center when a traceroute to r5 fails:

```
lab@dc> traceroute 10.0.3.5
traceroute to 10.0.3.5 (10.0.3.5), 30 hops max, 40 byte packets
traceroute: sendto: No route to host
 1 traceroute: wrote 10.0.3.5 40 chars, ret=-1
^C
```

In light of the *ospf-isis* policies in place on r6 and r7, and the fact that reachability problems have been confirmed in the data center, it now seems that NSSA area 2 is "broken" by virtue of there being no OSPF-based default route available for redistribution into the data center. Before making any changes to the baseline network, you should document your findings and bring them to the attention of the proctor. In this example, the proctor confirms the need for a default route in area 2 and authorizes the necessary changes on the ABRs that serve the NSSA. The following command is entered on r3, r4, and r5, which configures them to generate a default route into area 2:

```
[edit protocols ospf]
lab@r3# set area 2 nssa default-lsa default-metric 10
```

There is no need to specify an LSA Type 7 for the default route in this case, as summary LSAs are permitted in the NSSA. After the change is committed on r3, the results are confirmed at r6:

```
[edit]
lab@r6# run show route protocol ospf | match 0.0.0.0/0
0.0.0.0/0          *[OSPF/150] 00:00:23, metric 11, tag 0
```

Great, the default route is now present and active as an OSPF route. Before proceeding, you should verify that all three of the NSSA's ABRs are now configured to source a default route into area 2. When correctly configured, both r6 and r7 will display two viable next hops for the OSPF default route. The data center router should now be receiving the default route from both r6 and r7. After telnetting to the data center router, the presence of a default route pointing to r6 and r7 as the next hops is confirmed, as is the data center router's ability to reach various 10.0/16 destinations:

```
lab@dc> show route

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

0.0.0.0/0          *[IS-IS/160] 00:00:05, metric 21
                      to 10.0.8.2 via fe-0/0/0.0
                    > to 10.0.8.14 via fe-0/0/1.0
```

The default route correctly lists both r6 and r7 as viable next hops; this proves that the *ospf-isis* export policy is now functional on both r6 and r7. With the default route present, traceroutes originated at the data center now succeed:

```
lab@dc> traceroute 10.0.3.3
traceroute to 10.0.3.3  (10.0.3.3), 30 hops max, 40 byte packets
 1  10.0.8.14 (10.0.8.14)  0.377 ms  0.221 ms  0.155 ms
 2  10.0.8.9  (10.0.8.9)   0.435 ms  0.391 ms  0.388 ms
 3  10.0.3.3  (10.0.3.3)   0.815 ms  1.120 ms  1.071 ms

lab@dc> traceroute 10.0.6.2
traceroute to 10.0.6.2 (10.0.6.2), 30 hops max, 40 byte packets
 1  10.0.8.14 (10.0.8.14)  0.263 ms  0.185 ms  0.155 ms
 2  10.0.2.18 (10.0.2.18)  0.435 ms  0.374 ms  0.388 ms
 3  10.0.6.2  (10.0.6.2)   0.288 ms  0.285 ms  0.262 ms
```

Both of the traceroutes complete normally, but the reliance on a default route with two equal-cost next hops has resulted in a less than optimal forwarding path to some destinations, because the data center router has installed r7 as the default route's current next hop as this is being written. This situation can be considered normal, so for now you simply note the issue and move on with your network discovery actions.

With OSPF to IS-IS redistribution now confirmed, you examine the *isis-ospf* policy to determine the redistribution behavior expected in the IS-IS to OSPF direction:

```
[edit]
lab@r7# show policy-options policy-statement isis-ospf
term 1 {
    from {
        protocol isis;
        route-filter 192.168.0.0/22 longer;
    }
    then accept;
}
```

The *isis-ospf* policy seems pretty straightforward. Routes learned from IS-IS matching the `192.168.0/22 longer` route filter declaration should be redistributed into area 2 using an LSA Type 7 in accordance with the area's NSSA settings.

You begin verification of the IS-IS to OSPF redistribution aspects of the baseline network by confirming that both `r6` and `r7` have installed the IS-IS versions of the 192.168.0/22 data center routes as active. Recall that the configuration in `r6` and `r7` has adjusted the default preference of IS-IS Level 1 externals from 160 to 149, to ensure that they will be preferred to the versions being redistributed into OSPF by the other router:

```
[edit]
lab@r7# run show route 192.168.0/22

inet.0: 125015 destinations, 125029 routes (125015 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.0/24     *[IS-IS/149] 00:26:16, metric 10
                    > to 10.0.8.13 via fe-0/3/0.0
                    [OSPF/150] 00:25:52, metric 10, tag 0
                    > to 10.0.8.9 via fe-0/3/1.0
                    [BGP/170] 00:25:53, MED 10, localpref 100, from 10.0.9.6
                      AS path: I
                    > to 10.0.8.9 via fe-0/3/1.0
192.168.0.1/32     *[IS-IS/15] 00:26:16, metric 10
                    > to 10.0.8.13 via fe-0/3/0.0
                    [OSPF/150] 00:25:52, metric 10, tag 0
                    > to 10.0.8.9 via fe-0/3/1.0
                    [BGP/170] 00:25:53, MED 10, localpref 100, from 10.0.9.6
                      AS path: I
                    > to 10.0.8.9 via fe-0/3/1.0
```

```
192.168.1.0/24     *[IS-IS/149] 00:26:16, metric 20
                    > to 10.0.8.13 via fe-0/3/0.0
                    [OSPF/150] 00:25:52, metric 20, tag 0
                    > to 10.0.8.9 via fe-0/3/1.0
                    [BGP/170] 00:25:52, MED 20, localpref 100, from 10.0.9.6
                      AS path: I
                    > to 10.0.8.9 via fe-0/3/1.0
192.168.2.0/24     *[IS-IS/149] 00:26:16, metric 20
                    > to 10.0.8.13 via fe-0/3/0.0
                    [OSPF/150] 00:25:52, metric 20, tag 0
                    > to 10.0.8.9 via fe-0/3/1.0
                    [BGP/170] 00:25:52, MED 20, localpref 100, from 10.0.9.6
                      AS path: I
                    > to 10.0.8.9 via fe-0/3/1.0
192.168.3.0/24     *[IS-IS/149] 00:26:16, metric 20
                    > to 10.0.8.13 via fe-0/3/0.0
                    [OSPF/150] 00:25:52, metric 20, tag 0
                    > to 10.0.8.9 via fe-0/3/1.0
                    [BGP/170] 00:25:52, MED 20, localpref 100, from 10.0.9.6
                      AS path: I
                    > to 10.0.8.9 via fe-0/3/1.0
```

The output confirms that r7 has selected the IS-IS versions of the 192.168.0/22 routes as active. You can also determine from this display that r6 has redistributed the 192.168.0/22 routes into both OSPF and IBGP. These points help to confirm the correct operation of r6's redistribution policies. Though not shown, the same command is issued on r6 to confirm that it displays a similar view of the 192.168.0/22 routes. The presence of the data center routes are next confirmed in the backbone area with the following command entered on r3:

```
lab@r3# run show route 192.168.1/24

inet.0: 118083 destinations, 118105 routes (118083 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.1.0/24     *[OSPF/150] 00:12:59, metric 20, tag 0
                    > to 10.0.2.13 via fe-0/0/3.0
                    [BGP/170] 00:12:59, MED 20, localpref 100, from 10.0.9.6
                      AS path: I
                    > to 10.0.2.13 via fe-0/0/3.0
                    [BGP/170] 00:12:59, MED 20, localpref 100, from 10.0.9.7
                      AS path: I
```

```
> via at-0/1/0.0
  via so-0/2/0.100
```

Good, the routes are present in the OSPF backbone as both OSPF and BGP routes; the presence of two BGP next hops for the DC routes further confirms that both r6 and r7 are redistributing the 192.168.0/22 routes into BGP. Before considering your OSPF discovery exercise complete, you should take a few moments to trace routes to various internal destinations to verify there are no forwarding oddities at play in the baseline network. For example, the layout of area 2 results in packets taking an extra hop when r3 or r4 forwards packets to the loopback address of r7 or r6, respectively. This behavior is to be expected, because in this topology r4 learns r6's loopback address from a router LSA in area 2 (as flooded by r7) and from a network summary flooded into the backbone area by r5. Because an OSPF router always prefers internal (intra area) routes over a network summary, r4 forwards through r7 to reach the loopback address of r6. The same behavior is observed when r3 forwards to r7's loopback address, as shown here:

```
lab@r3# run traceroute 10.0.9.7
traceroute to 10.0.9.7 (10.0.9.7), 30 hops max, 40 byte packets
 1  10.0.2.13 (10.0.2.13)  0.776 ms  0.556 ms  0.426 ms
 2  10.0.8.6 (10.0.8.6)  0.700 ms  9.111 ms  0.648 ms
 3  10.0.9.7 (10.0.9.7)  0.577 ms  0.553 ms  0.514 ms
```

This situation can be considered par for the course, or could be corrected with an additional link between r6 and r7, with a static route, or with a redefinition of the OSPF area boundaries. In this case, you are informed that the baseline network is "operating as designed" so no corrective actions are taken at this time. With the completion of your traceroute testing, your operational analysis of area 2 is complete!

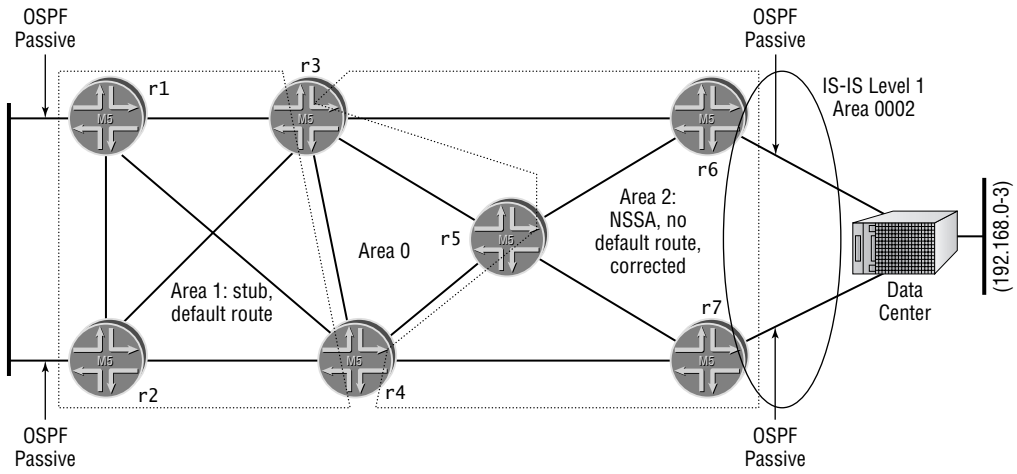
Summary of IGP Discovery

Your discovery activities have resulted in the determination that the baseline network consists of a multi-area OSPF IGP with mutual route redistribution occurring between the network core and data center locations. In this example, you were provided with an IGP that was largely functional and, for the most part, properly configured. The notable exception would be the missing OSPF default route in area 2 that led to connectivity problems for the data center.

Your findings have confirmed that all OSPF (and IS-IS) adjacencies are in place and that, with a few exceptions, optimal forwarding paths are in place. The exceptions you have noted include the data center router, which uses a 0/0 default route with two viable next hops to reach various destinations, and the extra hops incurred by r3 and r4 due to the specific layout of area 2.

Documenting your discovery findings is a good idea. Being able to refresh your memory with an accurate picture of the network that you have inherited may prevent mistakes in subsequent configuration tasks. Figure 1.3 provides an example of the key points that you have discovered in your JNCIE test bed so far.

FIGURE 1.3 Summary of IGP discovery findings



Notes:

Loopback addresses have not been assigned to specific areas (lo0 address advertised in Router LSA in all areas).

Passive OSPF interfaces on P1 and data center segments.

No authentication or route summarization in effect; summaries (LSA type 3) allowed in all areas.

Redistribution of OSPF default route to data center from both `r6` and `r7` was broken. Fixed with default-metric command on `r3`, `r4`, and `r5`.

Data center router running IS-IS, Level 1. `r6` and `r7` compatibly configured and adjacent.

Redistribution of 192.168.0/24 through 192.168.3/24 into OSPF from IS-IS by both `r6` and `r7`.

Adjustment to IS-IS level 1 external preference to ensure `r6` and `r7` always prefer IS-IS Level 1 externals over OSPF externals.

All adjacencies up and full reachability confirmed.

Sub-optimal routing detected at the data center router for some locations, and when `r3` and `r4` forward to some Area 2 addresses. This is the result of random nexthop choice for its default route and Area 2 topology specifics. Considered to be working as designed; no action taken.

Task 3: IBGP Discovery and Verification

With your network's IGPs and route redistribution confirmed as operational, it is time to take things up a notch by analyzing the network's IBGP configuration and operation. Once again, you begin your analysis on a backbone router:

```
[edit]
lab@r5# show protocols bgp
group int {
    type internal;
```

```
    local-address 10.0.3.5;
    neighbor 10.0.6.1;
    neighbor 10.0.6.2;
    neighbor 10.0.3.3;
    neighbor 10.0.3.4;
    neighbor 10.0.9.6;
    neighbor 10.0.9.7;
}
[edit]
lab@r5# show routing-options
static {
    route 10.0.200.0/24 {
        next-hop 10.0.1.102;
        no-readvertise;
    }
}
autonomous-system 65412;
```

Well, there certainly appears to be nothing fancy going on here! You now know that your test bed is (once again) using Autonomous System Number (ASN) *65412*. Further, the IBGP configuration at `r5` indicates that you have been provided with a full mesh of IBGP sessions using lo0-based peering. A quick glance at the status of `r5`'s IBGP sessions confirms that all six of its IBGP sessions have been correctly established:

```
[edit]
lab@r5# run show bgp summary
Groups: 1 Peers: 6 Down peers: 0
Table    Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0      125100     125088          0          0          0          0
Peer      AS    InPkt OutPkt OutQ Flaps  Last Up/Dwn State|#Active/Received/Damped...
10.0.3.3  65412 24421    166    0     0     1:21:54 125085/125085/0      0/0/0
10.0.3.4  65412   168    168    0     0     1:22:46 1/1/0                0/0/0
10.0.6.1  65412   165    167    0     0     1:22:02 1/1/0                0/0/0
10.0.6.2  65412   164    166    0     0     1:21:58 0/1/0                0/0/0
10.0.9.6  65412   167    166    0     0     1:21:52 1/6/0                0/0/0
10.0.9.7  65412   167    167    0     0     1:22:04 0/6/0                0/0/0
```

Seeing that all of `r5`'s loopback-based IBGP sessions are in the established state provides an additional check of your network's IGP, as the IGP is needed to route between the loopback addresses of the routers in the test bed. You also note that `r5` has received at least one route from each IBGP peer, and that it has received a whole bunch of routes from `r3`; you note that `r3` in turn EBGP peers with a transit provider T1, so these findings make a fair bit of sense. Your attention now shifts to the analysis of `r7`'s configuration. You note that the presence of an

EBGP peer in the form of C1 will make r7's configuration differ from that observed at r5:

```
[edit]
lab@r7# show protocols bgp
group int {
    type internal;
    local-address 10.0.9.7;
    export nhs;
    neighbor 10.0.6.1;
    neighbor 10.0.6.2;
    neighbor 10.0.3.3;
    neighbor 10.0.3.4;
    neighbor 10.0.3.5;
    neighbor 10.0.9.6;
}
group c1 {
    type external;
    export ebgp-out;
    neighbor 172.16.0.2 {
        peer-as 65010;
    }
}
```

The IBGP configuration of r7 is similar to that shown for r5, with the exception of the highlighted *nhs* export policy statement and the presence of EBGP-related configuration for the C1 peering session. The *nhs* export policy is displayed on r7:

```
[edit]
lab@r7# show policy-options policy-statement nhs
term 1 {
    from {
        protocol bgp;
        neighbor 172.16.0.2;
    }
    then {
        next-hop self;
    }
}
term 2 {
    from {
        route-filter 192.168.0.0/22 longer;
    }
    then accept;
}
```

The first term in the *nhs* policy resets the BGP next hop for routes learned from C1 to r7's RID. This eliminates the need to carry the various 172.16.0/30 EBGP link addresses in your IGP. The second term in the *nhs* policy results in the data center routes being redistributed into IBGP, presumably so that they can in turn be re-advertised to your network's EBGP peers by the other routers in the test bed. Note that r1 and r2 do not receive the data center routes via OSPF external LSAs due to a stub area's inability to carry external routing information.

The IBGP configuration on the remaining routers is similar to that shown for r7, with the following exceptions noted.

The advertise-inactive option has been set on r3 and r4 as highlighted:

```
[edit]
lab@r4# show protocols bgp
advertise-inactive;
group int {
    type internal;
    local-address 10.0.3.4;
    export nhs;
    neighbor 10.0.6.1;
    neighbor 10.0.6.2;
    neighbor 10.0.3.3;
    neighbor 10.0.3.5;
    neighbor 10.0.9.6;
    neighbor 10.0.9.7;
}
group c1 {
    type external;
    export ebgp-out;
    neighbor 172.16.0.6 {
        peer-as 65010;
    }
```

The presence of active OSPF routes for the data center on r3 and r4 will prevent their advertisement into EBGP without the use of some type of OSPF-to-BGP export policy. The advertise-inactive option on r3 and r4 alleviates this problem in the most expedient way possible with no policy modifications needed. The advertise-inactive option is not needed on r1 and r2 because they do not receive the OSPF advertisements for the DC's routes, thus making the IBGP versions of these routes active and therefore eligible for export using the default BGP policy.

The lack of a next hop self-policy on r1 and r2 is noted, but is not considered an issue at this time. Resetting the EBGP next hop is not needed on these routers because the 10.0.5/24 peering subnet is being advertised into OSPF due to the passive IGP setting on their fe-0/0/0 interfaces. Having the 10.0.5/24 subnet carried in OSPF makes P1's 10.0.5.254 EBGP next hop reachable by all routers in your AS.

As a final check on your network's IBGP operation, you verify that the data center's routes are present in both r1 and r2, and that each router displays two viable BGP next hops, as this will confirm that r1 and r2 are correctly receiving the 192.168.0/22 routes from both r6 and r7:

```
[edit]
lab@r2# run show route 192.168.2/24

inet.0: 118098 destinations, 118113 routes (118098 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.2.0/24     *[BGP/170] 01:27:12, MED 20, localpref 100, from 10.0.9.6
                      AS path: I
                    > to 10.0.4.1 via fe-0/0/2.0
                    [BGP/170] 01:27:12, MED 20, localpref 100, from 10.0.9.7
                      AS path: I
                    > to 10.0.4.9 via fe-0/0/1.0
```

Before moving into the EBGP and policy verification task, you should review each router's IBGP export policy, and you should quickly confirm that all IBGP session are established on all routers. You can assume that in this example all IBGP sessions are established and that no IBGP-related operational problems were detected.

Task 4: EBGP and Routing Policy Discovery

Having verified that your network's overall IGP and IBGP operation are sound, it is time to move on to your final network discovery task—namely the discovery and verification of your test bed's EBGP topology and its related routing policy.

P1 Peering

You begin the EBGP and policy discovery process on r1 by verifying its EBGP session status to P1:

```
[edit]
lab@r1# run show bgp neighbor 10.0.5.254
Peer: 10.0.5.254+179  AS 65050 Local: 10.0.5.1+1544   AS 65412
  Type: External    State: Established    Flags: <>
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ ebgp-out ]
  Options: <Preference HoldTime PeerAS Refresh>
```

```
  Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 120.120.0.1      Local ID: 10.0.6.1         Active Holdtime: 90
  Keepalive Interval: 30
  Local Interface: fe-0/0/0.0
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:            1
    Received prefixes:          1
    Suppressed due to damping:  0
  Last traffic (seconds): Received 23   Sent 2    Checked 2
  Input messages:  Total 884    Updates 16      Refreshes 0     Octets 17434
  Output messages: Total 894    Updates 23      Refreshes 0     Octets 17960
  Output Queue[0]: 0
```

The output confirms that the EBGP session to P1 is in the established state, and that one prefix has been received and installed as an active route over this session. The EBGP sessions' established state is an auspicious beginning, so you decide to analyze the EBGP configuration stanza on r1:

```
lab@r1# show protocols bgp
group int {
    type internal;
    local-address 10.0.6.1;
    neighbor 10.0.6.2;
    neighbor 10.0.3.3;
    neighbor 10.0.3.4;
    neighbor 10.0.3.5;
    neighbor 10.0.9.6;
    neighbor 10.0.9.7;
}
group p1 {
    type external;
    export ebgp-out;
    neighbor 10.0.5.254 {
        peer-as 65050;
    }
}
```

The highlighted EBGP portion of the configuration reveals a rather vanilla setup. There is no EBGP import policy in place, and a single export policy called, conveniently enough, *ebgp-out* has been applied. You display the *ebgp-out* policy to determine the expected EBGP advertisement behavior from r1 to P1:

```
[edit]
lab@r1# show policy-options policy-statement ebgp-out
term 1 {
    from {
        protocol aggregate;
        route-filter 10.0.0.0/16 exact;
    }
    then accept;
}
term 2 {
    from community transit;
    then reject;
}
[edit]
lab@r1# show routing-options aggregate
route 10.0.0.0/16;

[edit]
lab@r1# show policy-options community transit
members 65412:420;
```

The first term in the policy results in the advertisement of a locally defined aggregate route encompassing the addressing space of your AS; the aggregate route is also confirmed as present and active on r1 (not shown). The second term in the *ebgp-out* policy serves to block the advertisement of routes with the *transit* community attached. With the default policy left in place for all remaining BGP routes, you expect to see r1 advertise all remaining (and active) BGP routes to the P1 router. Assuming for the moment that the routes learned from transit peer T1 are being tagged with the *transit* community, you expect to see your AS's 10.0/16 aggregate, the data center routes, and both sets of customer routes being sent to P1.

A series of commands are now issued at r1 to confirm the advertisement of the expected routes to P1. These commands also serve to provide an ongoing check of the overall operations of your test bed, as the lack of advertisement for a given set of EBGP routes may constitute cause for further investigation:

```
lab@r1> show route advertising-protocol bgp 10.0.5.254 10.0/16

inet.0: 118092 destinations, 118107 routes (118092 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/16             Self                                    I
```

The aggregate for your AS is correctly being advertised to P1. This should allow P1 to respond to pings and traceroutes issued from within your AS.

The presence of a locally defined 10.0/16 aggregate is not causing reachability problems on r1 and r2 due to the presence of network summary (LSA Type 3) in their stub area. If network summaries were blocked by the area's ABRs, this aggregate definition would result in a black hole for internal destinations outside of area 1. This situation was documented, and solved, in the *JNCIP Study Guide* (Sybex, 2003).

The next command confirms that the data center routes are being advertised to P1:

```
lab@r1> show route advertising-protocol bgp 10.0.5.254 192.168.0/22

inet.0: 118092 destinations, 118107 routes (118092 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 192.168.0.0/24          Self                                    I
* 192.168.0.1/32          Self                                    I
* 192.168.1.0/24          Self                                    I
* 192.168.2.0/24          Self                                    I
* 192.168.3.0/24          Self                                    I
```

The next set of commands confirms that both sets of customer routes are being sent to P1:

```
lab@r1> show route advertising-protocol bgp 10.0.5.254 200.200/16

inet.0: 118093 destinations, 118108 routes (118093 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 200.200.0.0/16          Self                                    65010 I

lab@r1> show route advertising-protocol bgp 10.0.5.254 220.220/16

inet.0: 118094 destinations, 118109 routes (118094 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 220.220.0.0/16          Self                                    65020 I
```

The output (or lack thereof) from the last command in this series confirms that the 130.130/16 routes, as received from EBGP peer T1, are not being sent to the P1 router in accordance with the *ebgp-out* export policy's rejection of routes with the *transit* community:

```
lab@r1> show route advertising-protocol bgp 10.0.5.254 130.130/16
```

The results shown here indicate that all is well with the r1-P1 EBGP peering session and its related routing policy. Although not shown here, the same verification steps are also performed on r2 and similar results are obtained. These findings confirm that EBGP peering to the P1 router is operational.

T1 Peering

You next analyze the EBGP peering session to the T1 router using an approach similar to that demonstrated for the P1 peering session. Once again, you begin by verifying the EBGP session status to T1:

```
[edit]
lab@r3# run show bgp summary
Groups: 2 Peers: 7 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0            125079     125067          0          0          0          0
Peer        AS    InPkt OutPkt OutQ Flaps Last Up/Dwn State|#Active/Received/Damped...
172.16.0.14 65222  23868  24684  0     0     1:35:16 125064/125064/0     0/0/0
10.0.3.4    65412  214    24730  0     0     1:46:51 1/1/0               0/0/0
10.0.3.5    65412  215    24748  0     0     1:46:49 0/0/0               0/0/0
10.0.6.1    65412  215    24765  0     0     1:46:59 1/1/0               0/0/0
10.0.6.2    65412  215    24765  0     0     1:46:55 0/1/0               0/0/0
10.0.9.6    65412  218    24765  0     0     1:46:54 1/6/0               0/0/0
10.0.9.7    65412  217    24765  0     0     1:46:58 0/6/0               0/0/0
```

The highlighted entry confirms that the EBGP session between r3 and P1 has been correctly established, and that some 125,000 routes have been received over this peering session.

As was the case with the JNCIP examination, making "simple" mistakes when you are dealing with a full BGP routing table can have a significant impact on your network's health and general state of well-being. Extra care should be taken when BGP-related redistribution policies are placed into service with this many routes floating about!

Displaying the EBGP-related configuration on r3 reveals the following settings:

```
[edit]
lab@r3# show protocols bgp
advertise-inactive;
group int {
    type internal;
    local-address 10.0.3.3;
    export nhs;
    neighbor 10.0.6.1;
    neighbor 10.0.6.2;
    neighbor 10.0.3.4;
    neighbor 10.0.3.5;
    neighbor 10.0.9.6;
```

```
    neighbor 10.0.9.7;
}
group ext {
    import ebgp-in;
    export ebgp-out;
    neighbor 172.16.0.14 {
        peer-as 65222;
    }
}
```

The highlighted entries represent another rather basic EBGP peering configuration. Worth noting is the use of `advertise-inactive` to allow the export of the data center routes despite the fact that the routes are active as OSPF routes. Using this option avoids the need for some type of OSPF-to-EBGP export policy for the data center's routes. You also note the presence of group-level import and export policy, the contents of which are displayed next:

```
[edit]
lab@r3# show policy-options policy-statement ebgp-in
term 1 {
    from {
        protocol bgp;
        neighbor 172.16.0.14;
    }
    then {
        community add transit;
    }
}
[edit]
lab@r3# show policy-options community transit
members 65412:420;

[edit]
lab@r3# show policy-options policy-statement ebgp-out
term 1 {
    from {
        protocol aggregate;
        route-filter 10.0.0.0/16 exact;
    }
    then accept;
}
```

The *ebgp-in* policy functions to tag routes received from the T1 peer with the *transit* community. Recall that `r1` and `r2` are filtering routes with this community when sending EBGP

updates to the P1 router. The *ebgp-out* policy causes the advertisement of a locally defined aggregate route representing your AS's addressing space. Based on these findings, you can conclude that all active BGP routes will be sent from r3 to T1, as well as a locally defined 10.0/16 aggregate route. Further, the presence of advertise-inactive will result in the advertisement of the best BGP routes that are currently not active due to protocol preference, which means that in this case, the 192.168.0/22 data center routes should also be advertised to the T1 router.

As with the P1 peering, you now issue a series of **show route advertising-protocol bgp** commands to confirm if r3's EBGP route advertisements to T1 match your predictions:

```
lab@r3> show route advertising-protocol bgp 172.16.0.14 10.0/16

inet.0: 118054 destinations, 118076 routes (118054 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/16             Self                                    I

lab@r3> show route advertising-protocol bgp 172.16.0.14 120.120/16

inet.0: 125150 destinations, 125164 routes (125150 active, 0 holddown, 0 hidden)
  Prefix          Nexthop        MED  Lclpref  AS path
* 120.120.0.0/16  Self                        65050 I                 65050 I

lab@r3> show route advertising-protocol bgp 172.16.0.14 192.168.0/22

inet.0: 118054 destinations, 118076 routes (118054 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  192.168.0.0/24          Self                                    I
  192.168.0.1/32          Self                                    I
  192.168.1.0/24          Self                                    I
  192.168.2.0/24          Self                                    I
  192.168.3.0/24          Self                                    I
```

The output from the commands confirm all your predictions regarding the EBGP advertisement behavior at the r3–T1 EBGP peering. Note that the 192.168.0/22 data center routes are being advertised despite the lack of active route indication (there is no * next to them). Though not shown, you may assume that the 200.200/16 and 220.220/16 routes, owned by C1 and C2 respectively, have also been confirmed in r3's EBGP advertisements to the T1 peer. These results indicate that the r3–T1 EBGP peering session is working as expected.

Customer Peering

The next check of your network's EBGP and routing policy operation involves the discovery and verification of the EBGP peering to customer sites. In this example, the EBGP configuration

and routing policy configurations for both customer sites are virtually identical, so discovery and verification steps will be demonstrated only for the C1 peering points at r4 and r7.

r7 to C1 EBGP Peering

You begin your customer peering analysis and discovery with router r7, with the confirmation that the r7–C1 peering session is in the established state:

```
[edit]
lab@r7# run show bgp summary
Groups: 2 Peers: 6 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0            118032     118022          0          0          0          0
Peer        AS    InPkt OutPkt OutQ Flaps Last Up/Dwn State|#Active/Received/Damped...
172.16.0.2  65010  51182  26385  0     0    2:19:24 1/1/0               0/0/0
10.0.3.3    65412  26308    278  0     0    2:16:29 118012/118012/0    0/0/0
10.0.3.4    65412    274    277  0     0    2:16:22 0/1/0              0/0/0
10.0.3.5    65412    274    277  0     0    2:16:10 0/0/0              0/0/0
10.0.6.1    65412    275    277  0     0    2:16:23 1/1/0              0/0/0
10.0.6.2    65412    275    277  0     0    2:16:12 0/1/0              0/0/0
```

With r7's EBGP session to C1 confirmed as operational, you move on to the inspection of r7's EBGP configuration:

```
[edit]
lab@r7# show protocols bgp
group int {
    type internal;
    local-address 10.0.9.7;
    export nhs;
    neighbor 10.0.6.1;
    neighbor 10.0.6.2;
    neighbor 10.0.3.3;
    neighbor 10.0.3.4;
    neighbor 10.0.3.5;
}
group c1 {
    type external;
    export ebgp-out;
    neighbor 172.16.0.2 {
        peer-as 65010;
    }
}
```

Nothing of note here, except the presence of an *ebgp-out* export policy, the contents of which are displayed next:

```
[edit]
lab@r7# show policy-options policy-statement ebgp-out
term 1 {
    from {
        protocol aggregate;
        route-filter 10.0.0.0/16 exact;
    }
    then accept;
}
term 2 {
    from {
        route-filter 192.168.0.0/22 upto /24;
    }
    then accept;
}
```

The first term in r7's *ebgp-out* export policy functions to advertise a local 10.0/16 aggregate to EBGP peer C1. As with the routers in area 1, the presence of the local aggregate does not cause operational problems in area 2 due to the presence of network summaries (LSA Type 3s). The second policy term results in the redistribution of the data center routes from IS-IS into EBGP. Though not shown in this capture, you should recall that r6 and r7 also redistribute the same routes into IBGP so that r1 and r2 can in turn advertise the DC routes to the P1 router.

The analysis of r7's EBGP peering configuration indicates that C1 should be receiving the 10.0/16 aggregate, the 192.168.0/22 data center routes, C2's routes, T1's routes, and the routes learned from the P1 router. The same set of commands demonstrated for the T1 and P1 peering points are now issued to confirm your analysis. Although not shown here, you can assume that in this example all expected routes are confirmed as present in r7's EBGP advertisements to the C1 router.

r4 to C1 EBGP Peering

You now shift your attention to the C1 peering point at r4. After verifying that the EBGP session is established (not shown), you move onto the inspection of r4's EBGP configuration and routing policy:

```
[edit]
lab@r4# show protocols bgp
advertise-inactive;
group int {
    type internal;
    local-address 10.0.3.4;
```

```
    export nhs;
    neighbor 10.0.6.1;
    neighbor 10.0.6.2;
    neighbor 10.0.3.3;
    neighbor 10.0.3.5;
    neighbor 10.0.9.6;
    neighbor 10.0.9.7;
}
group c1 {
    type external;
    export ebgp-out;
    neighbor 172.16.0.6 {
        peer-as 65010;
    }
}
```

The highlighted entries in the output relate to the C1 EBGP peering, and are virtually identical to the settings shown for `r3`. Once again, the `advertise-inactive` option is being used to allow the export of the data center routes via EBGP when the BGP versions of these routes are not active due to global preference settings. The *`ebgp-out`* policy is now displayed:

```
[edit]
lab@r4# show policy-options policy-statement ebgp-out
term 1 {
    from {
        protocol aggregate;
        route-filter 10.0.0.0/16 exact;
    }
    then accept;
}
```

Based on the contents of the *`ebgp-out`* policy, you conclude that `r4` will advertise the same set of routes to C1 as was described for the `r7`–C1 peering. You now issue a series of **`show route advertising-protocol bgp`** commands on `r4` to confirm the advertisement of the 10.0/16 aggregate, the data center's 192.168.0/22 routes, and the routes learned from the T1, P1, and C2 EBGP peerings:

```
lab@r4> show route advertising-protocol bgp 172.16.0.6 10.0/16

inet.0: 125146 destinations, 125160 routes (125146 active, 5 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/16             Self                                    I
lab@r4> show route advertising-protocol bgp 172.16.0.6 192.168.0/22

inet.0: 125147 destinations, 125161 routes (125147 active, 5 holddown, 0 hidden)
```

```
  Prefix                  Nexthop              MED     Lclpref    AS path
  192.168.0.0/24          Self                                    I
  192.168.0.1/32          Self                                    I
  192.168.1.0/24          Self                                    I
  192.168.2.0/24          Self                                    I
  192.168.3.0/24          Self                                    I
```

This output confirms that the 10.0/16 aggregate and data center routes are correctly advertised from r4 to C1. The next set of commands verifies the remaining routes, all of which have been learned from the various EBGP peerings in your baseline network:

```
lab@r4> show route advertising-protocol bgp 172.16.0.6 130.130/16

inet.0: 125146 destinations, 125160 routes (125146 active, 5 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 130.130.0.0/16          Self                                    65222 I

lab@r4> show route advertising-protocol bgp 172.16.0.6 120.120/16

inet.0: 125146 destinations, 125160 routes (125146 active, 5 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 120.120.0.0/16          Self                                    65050 I

lab@r4> show route advertising-protocol bgp 172.16.0.6 220.220/16

inet.0: 125147 destinations, 125161 routes (125147 active, 5 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 220.220.0.0/16          Self                                    65020 I
```

The results indicate that the r4–C1 EBGP peering and routing policies are fully operational.

Final EBGP and Policy Checks

Before blessing the EBGP and policy operation of the baseline network that you have been lucky enough to inherit, it is a good idea to check for hidden routes and to confirm reachability and forwarding paths to all EBGP peers. You really should inspect all routers in the test bed for hidden routes but, because r5 has no EBGP peerings, any problems with next hop reachability will most likely manifest themselves at r5. The following command is used to determine hidden route status at r5:

```
[edit]
lab@r5# run show route hidden

inet.0: 125144 destinations, 125158 routes (125144 active, 0 holddown, 0 hidden)

[edit]
```

The lack of output from r5 indicates that none of the 125,000 or so routes that it has received are hidden. The absence of hidden routes provides an additional indication that your network's EBGP, IBGP, and IGP protocols are functioning correctly. You now issue a series of traceroute commands from r5 to verify external prefix reachability and to validate the forwarding paths to external destinations:

```
lab@r5> traceroute 120.120.0.1
traceroute to 120.120.0.1 (120.120.0.1), 30 hops max, 40 byte packets
 1  10.0.2.10 (10.0.2.10)  0.994 ms  0.765 ms  0.629 ms
 2  10.0.4.10 (10.0.4.10)  0.533 ms  0.529 ms  0.491 ms
 3  120.120.0.1 (120.120.0.1)  0.641 ms  0.610 ms  0.580 ms

lab@r5> traceroute 130.130.0.1
traceroute to 130.130.0.1 (130.130.0.1), 30 hops max, 40 byte packets
 1  10.0.2.2 (10.0.2.2)  1.295 ms  1.029 ms  1.136 ms
 2  130.130.0.1 (130.130.0.1)  1.078 ms  1.024 ms  1.171 ms

lab@r5> traceroute 200.200.0.1
traceroute to 200.200.0.1 (200.200.0.1), 30 hops max, 40 byte packets
 1  10.0.2.10 (10.0.2.10)  0.834 ms  0.680 ms  0.603 ms
 2  200.200.0.1 (200.200.0.1)  0.532 ms  0.540 ms  0.504 ms

lab@r5> traceroute 220.220.0.1
traceroute to 220.220.0.1 (220.220.0.1), 30 hops max, 40 byte packets
 1  10.0.8.5 (10.0.8.5)  0.724 ms  0.535 ms  0.464 ms
 2  220.220.0.1 (220.220.0.1)  0.575 ms  0.586 ms  0.543 ms
```

The traceroute commands all succeed, which provides confirmation that all EBGP peers are receiving the 10.0/16 aggregate for your AS. The indication that packets take optimal forwarding paths to external destinations provides further validation that all aspects of your baseline network are now operational. Before moving on to the first configuration scenario, it is advisable that you repeat your traceroutes testing from the data center router, taking care to source the packets from one of its 192.168.0/22 prefixes, as doing so will validate the operation of the default route used by the data center router while also confirming that all EBGP peers are receiving advertisements for the data center's routes. Although not shown, you can assume that all traceroute testing from the data center router succeeds in this example.

Summary of EBGP and Policy Discovery

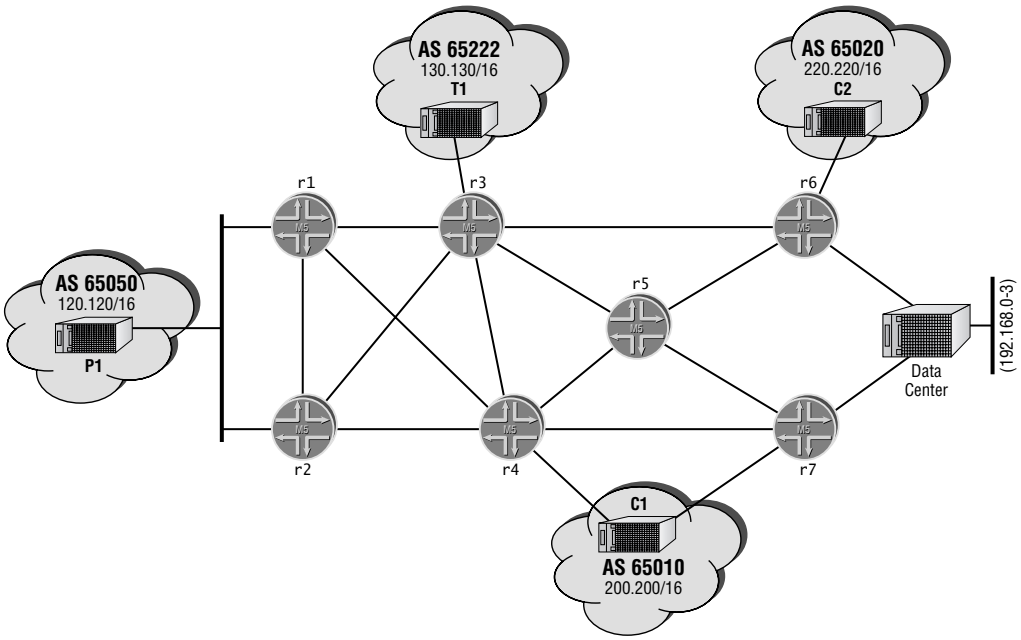
Once again, it is suggested that you take a few moments to document the results of your network discovery for future reference. After all, trying to lay down advanced services such as MPLS on top of a network that you are not intimately familiar with is akin to running with scissors, only more dangerous. Being able to jog your memory with the notes and documentation you make during a discovery scenario can make all the difference in later configuration tasks. A

summary of your IBGP, EBGP, and BGP-related routing policy is provided here:

- Full IBGP mesh between loopback addresses with all IBGP sessions established.
- Next hop self-policies on `r3`, `r4`, `r6`, and `r7`. Not needed on `r1` and `r2`.
- Data center routes redistributed into IBGP at `r6` and `r7`.
- All EBGP sessions established with no hidden routes.
- All active BGP routes being sent to all peers, with the exception of transit routes, which are not advertised to the P1 router.
- Local 10.0/16 aggregate advertised to all peers.
- Data center routes advertised to all peers; using `advertise-inactive` at `r3` and `r4`.
- No Martian filtering is in place.
- Connectivity and forwarding paths confirmed to all EBGP peers.

Figure 1.4 details your BGP-related findings in the context of a simplified topology map.

FIGURE 1.4 EBGP and policy discovery example



Notes:

Full IBGP mesh, all IBGP sessions established. EBGP peering to physical addresses, all EGBP sessions established.

10.0/16 aggregate, and data center routes confirmed to all EBGP peers. Local 10.0/16 aggregate is not black holing due to the presence of network summaries in all areas.

All active BGP routes sent to all EBGP peers, except T1 routes, which are tagged with a transit community and filtered from P1 at `r1` and `r2`.

Advertise inactive at `r3` and `r4`. `r6` and `r7` redistributing data center routes into both IGP and IBGP.

No operational issues detected. Trace routes to EGBP peers are successful and follow optimal paths. No hidden routes detected.

Complete Configurations for OSPF Baseline Network

Listings 1.1 through 1.7 provide the complete baseline configurations for all seven routers in the test bed as they existed at the conclusion of the network discovery and validation techniques demonstrated in the body of this chapter. You might need to modify the specifics to suit your hardware environment before loading the configurations into your test bed, but try to maintain as much similarity as possible. The baseline configuration will serve as the building block for the advanced topics covered in later chapters.

Listing 1.1: *r1* OSPF Baseline Configuration

```
[edit]
lab@r1# show | no-more
version 5.6R1.3;
system {
    host-name r1;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
```

```
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r1-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/0/0 {
        unit 0 {
            family inet {
                address 10.0.5.1/24;
            }
        }
    }
    fe-0/0/1 {
        unit 0 {
            family inet {
                address 10.0.4.14/30;
            }
        }
    }
    fe-0/0/2 {
        unit 0 {
            family inet {
                address 10.0.4.5/30;
            }
        }
    }
    fe-0/0/3 {
        unit 0 {
            family inet {
                address 10.0.4.18/30;
            }
        }
    }
```

```
    fxp0 {
        unit 0 {
            family inet {
                address 10.0.1.1/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.6.1/32;
            }
        }
    }
}
routing-options {
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    aggregate {
        route 10.0.0.0/16;
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.0.6.1;
            neighbor 10.0.6.2;
            neighbor 10.0.3.3;
            neighbor 10.0.3.4;
            neighbor 10.0.3.5;
            neighbor 10.0.9.6;
            neighbor 10.0.9.7;
        }
        group p1 {
            type external;
            export ebgp-out;
```

```
            neighbor 10.0.5.254 {
                peer-as 65050;
            }
        }
    }
    ospf {
        area 0.0.0.1 {
            stub;
            interface fe-0/0/0.0 {
                passive;
            }
            interface fe-0/0/1.0;
            interface fe-0/0/2.0;
            interface fe-0/0/3.0;
        }
    }
}
policy-options {
    policy-statement ebgp-out {
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then accept;
        }
        term 2 {
            from community transit;
            then reject;
        }
    }
    community transit members 65412:420;
}
```

Listing 1.2: *r2* OSPF Baseline Configuration

```
[edit]
lab@r2# show | no-more
version 5.6R1.3;
system {
    host-name r2;
    authentication-order [ radius password ];
    ports {
```

```
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r2-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/0/0 {
        unit 0 {
            family inet {
                address 10.0.5.2/24;
```

```
            }
        }
    }
    fe-0/0/1 {
        unit 0 {
            family inet {
                address 10.0.4.10/30;
            }
        }
    }
    fe-0/0/2 {
        speed 100m;
        unit 0 {
            family inet {
                address 10.0.4.2/30;
            }
        }
    }
    fe-0/0/3 {
        unit 0 {
            family inet {
                address 10.0.4.6/30;
            }
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 10.0.1.2/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.6.2/32;
            }
        }
    }
}
```

```
routing-options {
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    aggregate {
        route 10.0.0.0/16;
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.0.6.2;
            neighbor 10.0.6.1;
            neighbor 10.0.3.3;
            neighbor 10.0.3.4;
            neighbor 10.0.3.5;
            neighbor 10.0.9.6;
            neighbor 10.0.9.7;
        }
        group p1 {
            type external;
            export ebgp-out;
            neighbor 10.0.5.254 {
                peer-as 65050;
            }
        }
    }
    ospf {
        area 0.0.0.1 {
            stub;
            interface fe-0/0/0.0 {
                passive;
            }
            interface fe-0/0/1.0;
            interface fe-0/0/2.0;
```

```
            interface fe-0/0/3.0;
        }
    }
}
policy-options {
    policy-statement ebgp-out {
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then accept;
        }
        term 2 {
            from community transit;
            then reject;
        }
    }
    community transit members 65412:420;
}
```

Listing 1.3: *r3* OSPF Baseline Configuration (with Highlighted Corrections)

```
[edit]
lab@r3# show | no-more
version 5.6R1.3;
system {
    host-name r3;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
```

```
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r3-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/0/0 {
        unit 0 {
            family inet {
                address 10.0.4.13/30;
            }
        }
    }
    fe-0/0/1 {
        unit 0 {
            family inet {
                address 10.0.4.1/30;
            }
        }
    }
    fe-0/0/2 {
```

```
        unit 0 {
            family inet {
                address 172.16.0.13/30;
            }
        }
    }
    fe-0/0/3 {
        unit 0 {
            family inet {
                address 10.0.2.14/30;
            }
        }
    }
    at-0/1/0 {
        atm-options {
            vpi 0 {
                maximum-vcs 64;
            }
        }
        unit 0 {
            point-to-point;
            vci 50;
            family inet {
                address 10.0.2.2/30;
            }
        }
    }
    so-0/2/0 {
        dce;
        encapsulation frame-relay;
        unit 100 {
            dlci 100;
            family inet {
                address 10.0.2.5/30;
            }
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 10.0.1.3/24;
```

```
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.3.3/32;
            }
        }
    }
}
routing-options {
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    aggregate {
        route 10.0.0.0/16;
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        advertise-inactive;
        group int {
            type internal;
            local-address 10.0.3.3;
            export nhs;
            neighbor 10.0.6.1;
            neighbor 10.0.6.2;
            neighbor 10.0.3.4;
            neighbor 10.0.3.5;
            neighbor 10.0.9.6;
            neighbor 10.0.9.7;
        }
        group ext {
            import ebgp-in;
            export ebgp-out;
            neighbor 172.16.0.14 {
```

```
                peer-as 65222;
            }
        }
    }
    ospf {
        area 0.0.0.1 {
            stub default-metric 10;
            interface fe-0/0/0.0;
            interface fe-0/0/1.0;
        }
        area 0.0.0.0 {
            interface so-0/2/0.100;
            interface at-0/1/0.0;
        }
        area 0.0.0.2 {
            nssa {
                default-lsa default-metric 10;
            }
            interface fe-0/0/3.0;
        }
    }
}
policy-options {
    policy-statement nhs {
        term 1 {
            from {
                protocol bgp;
                neighbor 172.16.0.14;
            }
            then {
                next-hop self;
            }
        }
    }
    policy-statement ebgp-out {
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then accept;
```

```
        }
    }
    policy-statement ebgp-in {
        term 1 {
            from {
                protocol bgp;
                neighbor 172.16.0.14;
            }
            then {
                community add transit;
            }
        }
    }
    community transit members 65412:420;
}
```

Note that r3, r4, and r5 had their configurations modified (as highlighted) to resolve a problem with a missing default route in area 2.

Listing 1.4: *r4* OSPF Baseline Configuration (with Highlighted Corrections)

```
[edit]
lab@r4# show | no-more
version 5.6R1.3;
system {
    host-name r4;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
```

```
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r4-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/0/0 {
        unit 0 {
            family inet {
                address 172.16.0.5/30;
            }
        }
    }
    fe-0/0/1 {
        unit 0 {
            family inet {
                address 10.0.4.9/30;
            }
        }
    }
    fe-0/0/2 {
        unit 0 {
            family inet {
                address 10.0.4.17/30;
            }
        }
```

```
    }
    fe-0/0/3 {
        unit 0 {
            family inet {
                address 10.0.2.18/30;
            }
        }
    }
    so-0/1/0 {
        encapsulation frame-relay;
        unit 100 {
            dlci 100;
            family inet {
                address 10.0.2.6/30;
            }
        }
    }
    so-0/1/1 {
        encapsulation ppp;
        unit 0 {
            family inet {
                address 10.0.2.10/30;
            }
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 10.0.1.4/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.3.4/32;
            }
        }
    }
}
routing-options {
```

```
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    aggregate {
        route 10.0.0.0/16;
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        advertise-inactive;
        group int {
            type internal;
            local-address 10.0.3.4;
            export nhs;
            neighbor 10.0.6.1;
            neighbor 10.0.6.2;
            neighbor 10.0.3.3;
            neighbor 10.0.3.5;
            neighbor 10.0.9.6;
            neighbor 10.0.9.7;
        }
        group c1 {
            type external;
            export ebgp-out;
            neighbor 172.16.0.6 {
                peer-as 65010;
            }
        }
    }
    ospf {
        area 0.0.0.1 {
            stub default-metric 10;
            interface fe-0/0/1.0;
            interface fe-0/0/2.0;
        }
        area 0.0.0.0 {
```

```
            interface so-0/1/0.100;
            interface so-0/1/1.0;
        }
        area 0.0.0.2 {
            nssa {
                default-lsa default-metric 10;
            }
            interface fe-0/0/3.0;
        }
    }
}
policy-options {
    policy-statement ebgp-out {
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then accept;
        }
    }
    policy-statement nhs {
        term 1 {
            from {
                protocol bgp;
                neighbor 172.16.0.6;
            }
            then {
                next-hop self;
            }
        }
    }
}
```

Note that r3, r4, and r5 had their configurations modified (as highlighted) to resolve a problem with a missing default route in area 2.

Listing 1.5: *r5* OSPF Baseline Configuration (with Highlighted Corrections)

```
lab@r5# show | no-more
version 5.6R1.3;
system {
    host-name r5;
```

```
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r5-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/0/0 {
        unit 0 {
```

```
        family inet {
            address 10.0.8.6/30;
        }
    }
}
fe-0/0/1 {
    unit 0 {
        family inet {
            address 10.0.8.9/30;
        }
    }
}
so-0/1/0 {
    encapsulation ppp;
    unit 0 {
        family inet {
            address 10.0.2.9/30;
        }
    }
}
at-0/2/1 {
    atm-options {
        vpi 0 {
            maximum-vcs 64;
        }
    }
    unit 0 {
        point-to-point;
        vci 50;
        family inet {
            address 10.0.2.1/30;
        }
    }
}
fxp0 {
    unit 0 {
        family inet {
            address 10.0.1.5/24;
        }
    }
}
```

```
    lo0 {
        unit 0 {
            family inet {
                address 10.0.3.5/32;
            }
        }
    }
}
routing-options {
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.0.3.5;
            neighbor 10.0.6.1;
            neighbor 10.0.6.2;
            neighbor 10.0.3.3;
            neighbor 10.0.3.4;
            neighbor 10.0.9.6;
            neighbor 10.0.9.7;
        }
    }
    ospf {
        area 0.0.0.0 {
            interface at-0/2/1.0;
            interface so-0/1/0.0;
        }
        area 0.0.0.2 {
            nssa {
                default-lsa default-metric 10;
            }
            interface fe-0/0/0.0;
```

```
            interface fe-0/0/1.0;
        }
    }
}
```

Note that r3, r4, and r5 had their configurations modified (as highlighted) to resolve a problem with a missing default route in area 2.

Listing 1.6: *r6* OSPF Baseline Configuration

```
[edit]
lab@r6# show | no-more
version 5.6R1.3;
system {
    host-name r6;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
```

```
        file messages {
            any notice;
            authorization info;
        }
        file r6-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/1/0 {
        unit 0 {
            family inet {
                address 10.0.8.5/30;
            }
        }
    }
    fe-0/1/1 {
        unit 0 {
            family inet {
                address 10.0.2.13/30;
            }
        }
    }
    fe-0/1/2 {
        unit 0 {
            family inet {
                address 10.0.8.2/30;
            }
            family iso;
        }
    }
    fe-0/1/3 {
        unit 0 {
            family inet {
                address 172.16.0.9/30;
            }
        }
    }
    fxp0 {
```

```
        unit 0 {
            family inet {
                address 10.0.1.6/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.9.6/32;
            }
            family iso {
                address 49.0002.6666.6666.6666.00;
            }
        }
    }
}
routing-options {
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    aggregate {
        route 10.0.0.0/16;
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.0.9.6;
            export ibgp;
            neighbor 10.0.6.1;
            neighbor 10.0.6.2;
            neighbor 10.0.3.3;
            neighbor 10.0.3.4;
            neighbor 10.0.3.5;
            neighbor 10.0.9.7;
```

```
        }
        group c2 {
            type external;
            export ebgp-out;
            neighbor 172.16.0.10 {
                peer-as 65020;
            }
        }
    }
    isis {
        export ospf-isis;
        level 2 disable;
        level 1 external-preference 149;
        interface fe-0/1/2.0;
        interface lo0.0;
    }
    ospf {
        export isis-ospf;
        area 0.0.0.2 {
            nssa;
            interface fe-0/1/0.0;
            interface fe-0/1/2.0 {
                passive;
            }
            interface fe-0/1/1.0;
        }
    }
}
policy-options {
    policy-statement ospf-isis {
        term 1 {
            from {
                protocol ospf;
                route-filter 0.0.0.0/0 exact;
            }
            then accept;
        }
    }
    policy-statement isis-ospf {
        term 1 {
```

```
            from {
                protocol isis;
                route-filter 192.168.0.0/22 longer;
            }
            then accept;
        }
    }
    policy-statement ebgp-out {
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then accept;
        }
        term 2 {
            from {
                route-filter 192.168.0.0/22 upto /24;
            }
            then accept;
        }
    }
    policy-statement ibgp {
        term 1 {
            from {
                protocol bgp;
                neighbor 172.16.0.10;
            }
            then {
                next-hop self;
            }
        }
        term 2 {
            from {
                route-filter 192.168.0.0/22 longer;
            }
            then accept;
        }
    }
}
```

Listing 1.7: *r7* OSPF Baseline Configuration

```
[edit]
lab@r7# show | no-more
version 5.6R1.3;
system {
    host-name r7;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r7-cli {
            interactive-commands any;
            archive files 5;
```

```
        }
    }
}
interfaces {
    fe-0/3/0 {
        unit 0 {
            family inet {
                address 10.0.8.14/30;
            }
            family iso;
        }
    }
    fe-0/3/1 {
        unit 0 {
            family inet {
                address 10.0.8.10/30;
            }
        }
    }
    fe-0/3/2 {
        unit 0 {
            family inet {
                address 172.16.0.1/30;
            }
        }
    }
    fe-0/3/3 {
        unit 0 {
            family inet {
                address 10.0.2.17/30;
            }
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 10.0.1.7/24;
            }
        }
    }
    lo0 {
```

```
        unit 0 {
            family inet {
                address 10.0.9.7/32;
            }
            family iso {
                address 49.0002.7777.7777.7777.00;
            }
        }
    }
}
routing-options {
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    aggregate {
        route 10.0.0.0/16;
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.0.9.7;
            export nhs;
            neighbor 10.0.6.1;
            neighbor 10.0.6.2;
            neighbor 10.0.3.3;
            neighbor 10.0.3.4;
            neighbor 10.0.3.5;
            neighbor 10.0.9.6;
        }
        group c1 {
            type external;
            export ebgp-out;
            neighbor 172.16.0.2 {
                peer-as 65010;
            }
```

```
        }
    }
    isis {
        export ospf-isis;
        level 2 disable;
        level 1 external-preference 149;
        interface fe-0/3/0.0;
        interface lo0.0;
    }
    ospf {
        export isis-ospf;
        area 0.0.0.2 {
            nssa;
            interface fe-0/3/1.0;
            interface fe-0/3/0.0 {
                passive;
            }
            interface fe-0/3/3.0;
        }
    }
}
policy-options {
    policy-statement ospf-isis {
        term 1 {
            from {
                protocol ospf;
                route-filter 0.0.0.0/0 exact;
            }
            then accept;
        }
    }
    policy-statement isis-ospf {
        term 1 {
            from {
                protocol isis;
                route-filter 192.168.0.0/22 longer;
            }
            then accept;
        }
    }
    policy-statement ebgp-out {
```

```
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then accept;
        }
        term 2 {
            from {
                route-filter 192.168.0.0/22 upto /24;
            }
            then accept;
        }
    }
    policy-statement nhs {
        term 1 {
            from {
                protocol bgp;
                neighbor 172.16.0.2;
            }
            then {
                next-hop self;
            }
        }
        term 2 {
            from {
                route-filter 192.168.0.0/22 longer;
            }
            then accept;
        }
    }
}
```

Summary

This chapter provided you with a set of network discovery and verification tasks representative of those encountered by candidates as they begin their JNCIE examination. Because all of the topics addressed in this chapter were thoroughly tested in the prerequisite JNCIP examination, a JNCIE candidate is provided with a preconfigured network intended to serve as the starting point for the advanced configuration aspects that are the focus of the JNCIE examination proper. The goal of the network discovery scenario is twofold. The primary purpose is to provide

the candidate with a chance to assimilate and understand the operation of a network that he or she has never seen, before the candidate is asked to begin adding advanced services to the test bed. The secondary goal of the network discovery scenario is to provide a sanity check of the overall operation of the baseline network on the off chance that hardware errors (or configuration errors) are present.

Note that the specific nature of a given discovery scenario may vary over time to keep things interesting. The bottom line is that the prepared JNCIE candidate will possess the skills and protocol understanding needed to quickly reverse engineer an inherited network, and to rapidly access its overall operational state. It is expected that a JNCIE candidate will be able to detect and repair any configuration problems that may be uncovered during network discovery, with the exception of hardware-related failures that require proctor intervention.

Case Study: IS-IS Network Discovery and Validation Techniques

This case study presents a sample IGP discovery and validation scenario designed to demonstrate techniques and commands that are particularly useful when reverse engineering and validating an IS-IS based IGP. The focus of the case study is placed on IGP discovery because the functionality of IBGP, EBGP, and BGP-related routing policy is largely independent of the particular IGP in use. In fact, very few changes have been made to the IBGP, EBGP, and policy configuration documented in the course of the chapter body. Please refer back to Figure 1.2 for the case study topology.

The criteria for the case study are as follows:

- Discover and document the configuration of an IS-IS based network with mutual route redistribution.
- Verify correct IGP operation while documenting and correcting any IGP-related configuration problems encountered.
- Confirm that IBGP, EBGP, and routing policy are operational.

Although a number of discovery approaches can be used to reverse engineer an IGP, this author has found that it is normally more expedient to begin your analysis in the IGP core. The commands and techniques shown here are similar to the approach demonstrated in the chapter body; after all, if something works, why mess with it? You begin on `r3` with the inspection of its IGP-related configuration:

```
[edit]
lab@r3# show protocols
bgp {
    advertise-inactive;
    group int {
        type internal;
        local-address 10.0.3.3;
```

```
        export nhs;
        neighbor 10.0.6.1;
        neighbor 10.0.6.2;
        neighbor 10.0.3.4;
        neighbor 10.0.3.5;
        neighbor 10.0.9.6;
        neighbor 10.0.9.7;
    }
    group ext {
        import ebgp-in;
        export ebgp-out;
        neighbor 172.16.0.14 {
            peer-as 65222;
        }
    }
}
isis {
    interface fe-0/0/0.0 {
        level 2 disable;
    }
    interface fe-0/0/1.0 {
        level 2 disable;
    }
    interface fe-0/0/3.0 {
        level 1 disable;
    }
    interface at-0/1/0.0 {
        level 1 disable;
    }
    interface so-0/2/0.100 {
        level 1 disable;
    }
    interface lo0.0 {
        level 1 disable;
    }
}
```

The highlighted IGP portion of the configuration leads to the following observations:

- That `r3` is an attached router with a mix of L1 and L2 interfaces
- That authentication and route leaking are not configured for its L1 and L2 areas
- That `r3`'s lo0 address will not be injected into the Level 1 area due to the lo0 interface being disabled at IS-IS Level 1

Displaying the IS-IS interface status on r3 yields the following output:

```
[edit]
lab@r3# run show isis interface
IS-IS interface database:
Interface             L CirID Level 1 DR        Level 2 DR        L1/L2 Metric
at-0/1/0.0            2   0x1 Disabled          Point to Point         10/10
fe-0/0/0.0            1   0x3 r1.02             Disabled               10/10
fe-0/0/1.0            1   0x4 r2.03             Disabled               10/10
fe-0/0/3.0            2   0x2 Disabled          r6.03                  10/10
lo0.0                 0   0x1 Disabled          Passive                 0/0
so-0/2/0.100          2   0x1 Disabled          Point to Point         10/10
```

The display indicates that all of r3's interfaces, except its fxp0- and T1-facing fe-0/0/2 interfaces are considered ISO interfaces. This confirms that the iso family has been properly configured on their various logical units. Note that the lo0 interface is also listed as a passive IS-IS Level 2 interface; with some JUNOS software versions, it is critical that you actually run IS-IS on the interface that serves as the source of the ISO NET, making the presence of lo0 in the IS-IS interface display a good thing. The omission of r3's lo0 interface from the Level 1 area is intentional in this case. The intent is to prevent extra hops (through the Level 1 area) when r4 forwards packets to r3's lo0 address. Keeping the lo0 addresses of r3 and r4 from the Level 1 area is necessary if we want r3 and r4 to use their Level 2 link when forwarding to each other's loopback addresses, because an IS-IS router will always prefer a L1 internal route over the same route in Level 2, regardless of preference settings or route metrics. Traceroute testing confirms that r3 and r4 have optimal forwarding between loopback addresses as a result of this configuration:

```
[edit]
lab@r3# run traceroute 10.0.3.4
traceroute to 10.0.3.4 (10.0.3.4), 30 hops max, 40 byte packets
 1  10.0.3.4 (10.0.3.4)  1.061 ms  0.890 ms  0.795 ms
```

You next display r3's lo0 interface configuration. The output displays the ISO NET configured for r3. Based on the display, you are able to determine that r1, r2, r3, and r4 should all be in area 0001, because an IS-IS L1 adjacency will form only between routers that share a common area ID. You also note that the SYS-ID coding is based on the assigned router number. This coding approach should not pose a problem because the resulting SYS-IDs will have the uniqueness required for proper IS-IS operation:

```
[edit]
lab@r3# show interfaces lo0
unit 0 {
    family inet {
        address 10.0.3.3/32;
    }
    family iso {
```

```
        address 49.0001.3333.3333.3333.00;
    }
}
```

With the inspection of r3's IGP configuration completed, it makes sense to ascertain the state of its adjacencies, as you now have some idea of what to expect:

```
[edit]
lab@r3# run show isis adjacency
Interface             System         L State       Hold (secs)  SNPA
at-0/1/0.0            r5             2 Up                   21
fe-0/0/0.0            r1             1 Up                    6  0:a0:c9:6f:7b:3e
fe-0/0/1.0            r2             1 Up                    7  0:a0:c9:6f:7a:ff
fe-0/0/3.0            r6             2 Up                    7  0:d0:b7:3f:af:73
so-0/2/0.100          r4             2 Up                   21
```

The output shows that r3 has the five adjacencies one would expect, given the test bed topology and r3's IS-IS configuration. The output makes it easy to confirm what interfaces are running IS-IS, and at what IS-IS level. The following command quickly determines if the backbone area is correctly receiving LSPs from all of the routers in the test bed:

```
[edit]
lab@r3# run show isis hostname
IS-IS hostname database:
System ID      Hostname                                          Type
1111.1111.1111 r1                                                Dynamic
2222.2222.2222 r2                                                Dynamic
3333.3333.3333 r3                                                Static
4444.4444.4444 r4                                                Dynamic
5555.5555.5555 r5                                                Dynamic
6666.6666.6666 r6                                                Dynamic
7777.7777.7777 r7                                                Dynamic
```

The results could not be any better! You know that IS-IS is working overall, in that the backbone has received LSPs from all routers in the test bed. The final check at r3 determines correct IS-IS functionality with regard to the advertisement of IP routes by verifying that IS-IS routes for the loopback addresses of all routers in the test bed are present:

```
[edit]
lab@r3# run show route protocol isis | match \32
10.0.3.4/32        *[IS-IS/15] 02:49:44, metric 20
10.0.3.5/32        *[IS-IS/18] 00:07:08, metric 20
10.0.6.1/32        *[IS-IS/15] 03:11:24, metric 10
10.0.6.2/32        *[IS-IS/15] 02:49:44, metric 10
10.0.9.6/32        *[IS-IS/18] 00:33:06, metric 10
10.0.9.7/32        *[IS-IS/18] 00:07:08, metric 20
```

The results confirm that r3 has learned routes through IS-IS for the loopback addresses of all remote routers (r3's loopback address is not learned through IS-IS, and is therefore not listed). You can assume that similar results are obtained when the same commands are issued on r4. A quick look at r2's IS-IS configuration returns the following:

```
[edit]
lab@r2# show protocols isis
level 2 disable;
interface fe-0/0/0.0 {
    passive;
}
interface fe-0/0/1.0;
interface fe-0/0/2.0;
interface fe-0/0/3.0;
interface lo0.0;
```

r2's IS-IS configuration is pretty basic; the only real item to note is the fact that the router is running a passive IS-IS instance on its fe-0/0/0 interface. As with the OSPF example in the chapter body, the passive interface setting ensures that the 10.0.5/25 subnet will be reachable as an IS-IS internal route without chancing IGP adjacency formation to peer P1. Although not shown, the same passive interface observation is made when inspecting r1's IS-IS stanza. You next confirm IS-IS adjacency status, at r2:

```
[edit]
lab@r2# run show isis adjacency
Interface           System         L State       Hold (secs)  SNPA
fe-0/0/1.0              r4           1 Up                 21  0:90:69:6b:30:1
fe-0/0/2.0              r3           1 Up                 23  0:90:69:6d:98:1
fe-0/0/3.0              r1           1 Up                  6  0:a0:c9:6f:7b:84
```

With r2 displaying the expected number and types of IS-IS adjacencies, things are looking good for IS-IS functionality in area 0001. You decide to shift your discovery activities to r5 as a result:

```
lab@r5# show
export l1-l2;
interface fe-0/0/0.0 {
    level 2 disable;
}
interface fe-0/0/1.0 {
    level 2 disable;
}
interface so-0/1/0.0 {
    level 1 {
        passive;
```

```
    }
}
interface at-0/2/1.0 {
    level 1 {
        passive;
    }
}
interface lo0.0;
```

As with r3 and r4, r5's configuration indicates that it is a L1/L2 (attached) router by virtue of the mix of L1 and L2 interface statements in its IS-IS stanza. Unlike r3 and r4, r5 is running IS-IS (passively) at both Level 1 and Level 2 on its lo0 interface; this will result in the advertisement of its lo0 address in both its L1 and L2 LSPs. The passive configuration on r5's core facing interfaces prevents inefficient routing in area 0002, by having r5 advertise the 10.0.2.0/30 and 10.0.2.8/30 prefixes in the L1 LSP it sends into area 0002. Disabling Level 1 on r5's core facing interfaces will result in r6 and r7 incurring extra hops when forwarding to these prefixes, as their only routes to these destinations would be learned through the L2 LSPs generated by r3 and r4, respectively. A quick traceroute at r7 confirms proper forwarding paths to core prefixes:

```
[edit]
lab@r7# run traceroute 10.0.2.1
traceroute to 10.0.2.1 (10.0.2.1), 30 hops max, 40 byte packets
 1  10.0.2.1 (10.0.2.1)  0.758 ms  0.558 ms  0.454 ms

[edit]
lab@r7# run traceroute 10.0.2.9
traceroute to 10.0.2.9 (10.0.2.9), 30 hops max, 40 byte packets
 1  10.0.2.9 (10.0.2.9)  0.644 ms  0.489 ms  0.436 ms
```

The presence of an *l1-l2* export policy is also noted and the contents displayed:

```
lab@r5# show policy-options policy-statement l1-l2
term 1 {
    from {
        protocol isis;
        level 1;
        route-filter 192.168.0.0/22 longer;
    }
    to level 2;
    then accept;
}
```

The *l1-l2* policy instructs r5 to leak L1 external routes matching the route filter statement into the backbone (Level 2) area. Note that L1 internals are leaked into L2 areas by default, but policy is needed for the leaking of L1 externals. From this, you surmise that the routes

associated with the data center router are being redistributed by r6 and r7 into IS-IS as Level 1 externals. You next display r5's IS-IS interface status along with its ISO NET:

```
[edit]
lab@r5# run show isis interface
IS-IS interface database:
Interface             L CirID Level 1 DR        Level 2 DR        L1/L2 Metric
at-0/2/1.0            2   0x1 Passive           Point to Point         10/10
fe-0/0/0.0            1   0x2 r6.02             Disabled               10/10
fe-0/0/1.0            1   0x3 r5.03             Disabled               10/10
lo0.0                 0   0x1 Passive           Passive                 0/0
so-0/1/0.0            2   0x1 Passive           Point to Point         10/10

[edit]
lab@r5# show interfaces lo0
unit 0 {
    family inet {
        address 10.0.3.5/32;
    }
    family iso {
        address 49.0002.5555.5555.5555.00;
    }
}
```

Based on the display, you conclude that r5, r6, and r7 are attached to IS-IS area 49.0002, and that r5 should have L2 adjacencies on its core facing interfaces and L1 adjacencies on the Fast Ethernet links to r6 and r7. The IS-IS adjacency status is now verified on r5:

```
[edit]
lab@r5# run show isis adjacency
Interface           System       L State      Hold (secs)      SNPA
at-0/2/1.0          r3           2 Up                 21
fe-0/0/0.0          r6           1 Up                  6       0:d0:b7:3f:af:f
fe-0/0/1.0          r7           1 Up                 21       0:60:94:51:c4:27
so-0/1/0.0          r4           2 Up                 26
```

The results confirm that r5 has the expected number of adjacencies (4), and further validates that area 0002 is a Level 1 area. With r5's configuration and operation looking good, your attention shifts to r7:

```
[edit]
lab@r7# show protocols
bgp {
    group int {
        type internal;
```

```
        local-address 10.0.9.7;
        export nhs;
        neighbor 10.0.6.1;
        neighbor 10.0.6.2;
        neighbor 10.0.3.3;
        neighbor 10.0.3.4;
        neighbor 10.0.3.5;
        neighbor 10.0.9.6;
    }
    group c1 {
        type external;
        export ebgp-out;
        neighbor 172.16.0.2 {
            peer-as 65010;
        }
    }
}
isis {
    export rip-isis;
    interface fe-0/3/0.0 {
        level 2 disable;
        level 1 passive;
    }
    interface fe-0/3/1.0 {
        level 2 disable;
    }
    interface fe-0/3/3.0 {
        level 1 disable;
    }
    interface lo0.0;
}
rip {
    group dc {
        export static-rip;
        neighbor fe-0/3/0.0;
    }
}
```

The highlighted output indicates that r7 is running both IS-IS and RIP. From this, you surmise that the data center router must now be configured to advertise the 192.168.0/22 routes to r6 and r7 using RIP. The *rip-isis* export policy is now displayed. The output

confirms that r7 is configured to redistribute the data center's routes from RIP into IS-IS:

```
[edit]
lab@r7# show policy-options policy-statement rip-isis
term 1 {
    from {
        protocol rip;
        route-filter 192.168.0.0/22 longer;
    }
    then accept;
}
```

Displaying r7's *isis-rip* policy tells you that r7 should be sending a statically defined default route to the data center router:

```
[edit]
lab@r7# show policy-options policy-statement static-rip
term 1 {
    from {
        protocol static;
        route-filter 0.0.0.0/0 exact;
    }
    then accept;
}
```

The static route definition is also confirmed:

```
[edit]
lab@r7# show routing-options static
route 10.0.200.0/24 {
    next-hop 10.0.1.102;
    no-readvertise;
}
route 0.0.0.0/0 reject;
```

Why a Static Default Route?

The astute reader is likely wondering why a static default route has been defined on r6 and r7, especially considering that a Level 1 router normally installs an IS-IS based default route when the presence of an Attached router is detected in a Level 1 area through the setting of the Attached bit in Level 1 LSPs. The key here is the observation that area 0002 has *only* Attached routers, in that r5, r6, and r7 are all L2 and L1 Attached. Because an Attached router will not install a default route based on the presence of the Attached bit in the LSPs received from other Attached routers, a static default (or generated route) route is defined on r6 and r7.

Now that you know what to expect, you confirm r7's adjacency status and the presence of the data center's route RIP routes:

```
[edit]
lab@r7# run show isis adjacency
Interface             System         L State       Hold (secs)   SNPA
fe-0/3/1.0            r5               1 Up                 7   0:90:69:69:70:1
fe-0/3/3.0            r4               2 Up                 8   0:90:69:6b:30:3
```

Good! All of the expected adjacencies are up. You move on to confirm the presence of RIP routes at r7:

```
[edit]
lab@r7# run show route 192.168.0/22

inet.0: 118137 destinations, 118138 routes (118137 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.0/24     *[RIP/100] 01:16:43, metric 2, tag 0
                    > to 10.0.8.13 via fe-0/3/0.0
192.168.1.0/24     *[RIP/100] 01:16:43, metric 2, tag 0
                    > to 10.0.8.13 via fe-0/3/0.0
192.168.2.0/24     *[RIP/100] 01:16:43, metric 2, tag 0
                    > to 10.0.8.13 via fe-0/3/0.0
192.168.3.0/24     *[RIP/100] 01:16:43, metric 2, tag 0
                    > to 10.0.8.13 via fe-0/3/0.0
```

The output from r7 confirms it is correctly learning the 192.168.0/22 DC routes through the RIP protocol. Though the results are not shown here, you can assume that similar results were obtained when inspecting r6. All of the results obtained thus far in your IS-IS IGP discovery case study have been positive. You should now be able to document your IGP discovery findings on a copy of the test bed topology. However, before calling it quits, you wisely opt to further validate the operation of your IGP through some traceroute testing:

```
[edit]
lab@r6# run traceroute 10.0.3.4
traceroute to 10.0.3.4 (10.0.3.4), 30 hops max, 40 byte packets
 1  10.0.2.14 (10.0.2.14)  0.437 ms  0.352 ms  0.270 ms
 2  10.0.3.4 (10.0.3.4)  0.508 ms  0.464 ms  0.436 ms

[edit]
lab@r6# run traceroute 10.0.3.3
traceroute to 10.0.3.3 (10.0.3.3), 30 hops max, 40 byte packets
 1  10.0.3.3 (10.0.3.3)  0.592 ms  0.461 ms  0.422 ms
```

The traceroutes to attached routers r3 and r4 succeed, but things are not so positive for the traceroute to r1:

```
[edit]
lab@r6# run traceroute 10.0.6.1
traceroute to 10.0.6.1 (10.0.6.1), 30 hops max, 40 byte packets
 1  10.0.2.14 (10.0.2.14)  0.390 ms  0.287 ms  0.243 ms
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
```

The timeouts are not expected, so you move to r3, which is the first and only hop shown in the traceroute, to determine what is amiss:

```
[edit]
lab@r3# run traceroute 10.0.6.1
traceroute to 10.0.6.1 (10.0.6.1), 30 hops max, 40 byte packets
 1  10.0.6.1 (10.0.6.1)  0.743 ms  0.553 ms  0.475 ms
```

The traceroute from r3 to r1 is successful, so your attention shifts to r1 itself:

```
[edit]
lab@r1# run traceroute 10.0.3.5
traceroute to 10.0.3.5 (10.0.3.5), 30 hops max, 40 byte packets
traceroute: sendto: No route to host
 1 traceroute: wrote 10.0.3.5 40 chars, ret=-1
^C
[edit]
lab@r1# run show route 10.0.3.5

inet.0: 18 destinations, 19 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/16         *[Aggregate/130] 00:13:15
                      Reject
```

D'oh! The local definition of a 10.0/16 aggregate has created a black hole on r1 and r2 for all 10.0/16 destinations outside of their Level 1 area. Note that the presence of the same 10.0/16 aggregate route had no impact on the operation of the OSPF IGP, as demonstrated in the chapter body, because neither the stub nor the NSSA areas were blocking network summaries. Recalling that an IS-IS L1 area functions much as an OSPF stub/NSSA area with no-summaries, the problems caused by the local aggregate make perfect sense. This problem is not occurring on r6 and r7, because they are attached routers with full routing knowledge of the IS-IS domain.

After noting the problem and obtaining permission from the proctor to make baseline configuration changes, you decide to correct the issue by adjusting the IBGP export policy on r3

and r4 to effect the advertisement of the 10.0/16 aggregate to r1 and r2 through IBGP. You must be careful that your policy advertises a next hop for the aggregate route that is *within* L1 area 0001. The default behavior will be to set the route's next hop to the lo0 address of the advertising router, which will cause the 10.0/16 route to be hidden on r1 and r2 due to their inability to resolve the advertised BGP next hop (10.0.3.3 or 10.0.3.4) through the 10.0/16 route itself. (A BGP route cannot have its next hop resolved through itself because this can result in recursion problems.)

Advertising the aggregate to r1 and r2 through IBGP allows for the removal of the local 10.0/16 aggregate route definition from r1 and r2, while still providing them with the ability to advertise the 10.0/16 route to their EBGP peer P1. The highlighted entries show the modifications that were made to the IBGP portion of r4's configuration to evoke per-neighbor IBGP export policy (similar changes were also made at r3):

```
[edit]
lab@r4# show protocols bgp
advertise-inactive;
group int {
    type internal;
    local-address 10.0.3.4;
    export nhs;
    neighbor 10.0.6.1 {
        export r1;
    }
    neighbor 10.0.6.2 {
        export r2;
    }
    neighbor 10.0.3.3;
    neighbor 10.0.3.5;
    neighbor 10.0.9.6;
    neighbor 10.0.9.7;
}
group c1 {
    type external;
    export ebgp-out;
    neighbor 172.16.0.6 {
        peer-as 65010;
    }
}
```

And the new *r1* and *r2* policies are displayed with the next hop settings for the aggregate route highlighted:

```
[edit]
lab@r4# show policy-options policy-statement r1
```

```
term 1 {
    from {
        protocol aggregate;
        route-filter 10.0.0.0/16 exact;
    }
    then {
        next-hop 10.0.4.17;
        accept;
    }
}
term 2 {
    from {
        protocol bgp;
        neighbor 172.16.0.6;
    }
    then {
        next-hop self;
    }
}

[edit]
lab@r4# show policy-options policy-statement r2
term 1 {
    from {
        protocol aggregate;
        route-filter 10.0.0.0/16 exact;
    }
    then {
        next-hop 10.0.4.9;
        accept;
    }
}
term 2 {
    from {
        protocol bgp;
        neighbor 172.16.0.6;
    }
    then {
```

```
        next-hop self;
    }
}
```

The first term in the *r1* and *r2* export policies results in r3 and r4 advertising their 10.0/16 aggregate to r1 and r2 with the BGP next hop set to an IP address that exists within their Level 1 area. Note that the next hop advertised to r1 differs from that sent to r2 in an effort to help promote optimal forwarding paths wherever possible. Because the next hop for the 10.0/16 now resolves through a more specific route (as opposed to the 10.0/16 route itself), the route is no longer hidden and is therefore eligible for export to the P1 router by r1 and r2. The second policy term functions to set next hop self on the routes being learned from C1 peering. Though not shown, r3 now has similar *r1* and *r2* policies in place.

After deleting the local aggregate at r1 and r2, further testing confirms that all is well:

```
[edit]
lab@r1# run show route 10.0/16

inet.0: 118087 destinations, 118094 routes (118087 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/16        *[BGP/170] 00:10:31, localpref 100, from 10.0.3.3
                      AS path: I
                    > to 10.0.4.13 via fe-0/0/1.0
                    [BGP/170] 00:10:34, localpref 100, from 10.0.3.4
                      AS path: I
                    > to 10.0.4.17 via fe-0/0/4.0
. . .
```

The 10.0/16 aggregate, as received from r3 and r4, is confirmed with the output just shown. You next verify that the aggregate is being correctly sent on to the P1 router:

```
[edit]
lab@r1# run show route advertising-protocol bgp 10.0.5.254 10.0/16

inet.0: 118167 destinations, 118182 routes (118167 active, 0 holddown, 0 hidden)
  Prefix                 Nexthop              MED     Lclpref    AS path
* 10.0.0.0/16            Self
```

The output confirms the advertisement of the 10.0/16 aggregate to router P1. The next set of commands verifies reachability and forwarding paths from Level 1 router r1 to various internal and external destinations:

```
[edit]
lab@r1# run traceroute 10.0.9.7
traceroute to 10.0.9.7 (10.0.9.7), 30 hops max, 40 byte packets
 1  10.0.4.13 (10.0.4.13)  0.409 ms  0.341 ms  0.266 ms
```

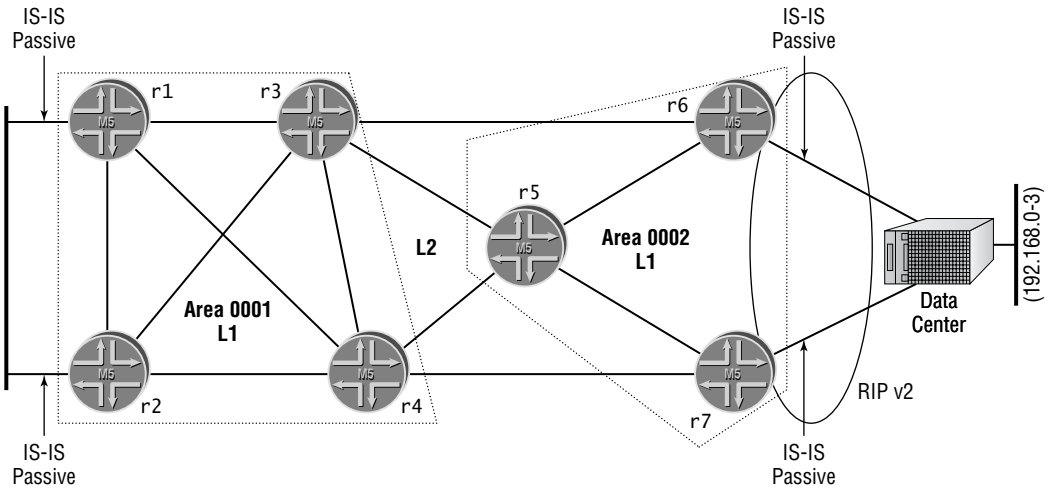
```
 2  10.0.2.1 (10.0.2.1)  0.811 ms  1.054 ms  0.798 ms
 3  10.0.9.7 (10.0.9.7)  0.705 ms  0.648 ms  0.405 ms

[edit]
lab@r1# run traceroute 192.168.0.1
traceroute to 192.168.0.1 (192.168.0.1), 30 hops max, 40 byte packets
 1  10.0.4.13 (10.0.4.13)  0.400 ms  0.293 ms  0.250 ms
 2  10.0.2.13 (10.0.2.13)  0.157 ms  0.153 ms  0.133 ms
 3  192.168.0.1 (192.168.0.1)  0.260 ms  0.231 ms  0.209 ms

[edit]
lab@r1# run traceroute 130.130.0.1
traceroute to 130.130.0.1 (130.130.0.1), 30 hops max, 40 byte packets
 1  10.0.4.13 (10.0.4.13)  0.392 ms  0.289 ms  0.248 ms
 2  130.130.0.1 (130.130.0.1)  0.172 ms  0.166 ms  0.144 ms

[edit]
lab@r1# run traceroute 220.220.0.1
traceroute to 220.220.0.1 (220.220.0.1), 30 hops max, 40 byte packets
 1  10.0.4.13 (10.0.4.13)  0.388 ms  0.300 ms  0.247 ms
 2  10.0.2.13 (10.0.2.13)  0.160 ms  0.152 ms  0.130 ms
 3  220.220.0.1 (220.220.0.1)  0.246 ms  0.228 ms  0.208 ms

[edit]
lab@r1# run traceroute 200.200.0.1
traceroute to 200.200.0.1 (200.200.0.1), 30 hops max, 40 byte packets
 1  10.0.4.13 (10.0.4.13)  0.408 ms  0.291 ms  0.248 ms
 2  10.0.2.6 (10.0.2.6)  0.309 ms  0.276 ms  0.253 ms
 3  200.200.0.1 (200.200.0.1)  0.178 ms  0.180 ms  0.154 ms
```

The forwarding paths shown have all been optimal, with the exception of the extra hop through `r3` that occurs when `r1` traces routes to C1's destinations. Closer inspection reveals that the extra hop is the result of C1's 200.200/16 route resolving through the 10.0/16 aggregate, coupled with the fact that both `r1` and `r2` prefer the 10.0/16 advertisement from `r3` to that learned from `r4`, due to `r3`'s lower RID. Because extra hops are sometimes inevitable when relying on aggregate or default routing, this condition is considered par for the course and, other than simply noting the condition, no additional actions are taken. With full reachability and optimal forwarding confirmed to all internal and external destinations, you have finished the validation aspects of the IS-IS based IGP discovery case study.

Although not shown here, you should quickly confirm that all IBGP and EBGP sessions are correctly established, and that no hidden route problems exist, before considering your baseline network operational. You can assume that there are no operational problems in the test bed at this time. To complete the IGP discovery case study, you must document your findings.

Figure 1.5 provides a summary of your IGP discovery case study findings. The figure also notes the operational issues that were discovered, and rectified, in this case study.

FIGURE 1.5 Results from IGP discovery case study



Notes:

Multi-level IS-IS, Areas 0001 and 0002 with ISO NET based on router number.

lo0 address of `r3` and `r4` not injected into Area 0001 to ensure optimal forwarding between 10.0.3.3 and 10.0.3.4.

Passive setting on `r5`'s core interfaces for optimal Area 0002-to-core routing.

No authentication or route summarization. Routing policy at `r5` to leak L1 externals (DC routes) to L2.

Redistribution of static default route to data center from both `r6` and `r7`. Redistribution of 192.168.0/24 through 192.168.3/24 routes from RIP into IS-IS by both `r6` and `r7`.

All adjacencies are up, reachability problem discovered at `r1` and `r2` caused by local aggregate definition. Corrected through IBGP policy to effect 10.0/16 route advertisement from `r3` and `r4` to `r1` and `r2`; removed local aggregate from `r1` and `r2`.

Suboptimal routing detected at the data center and at `r1`/`r2` for some locations. This is the result of random nexthop choice for data center's default, and the result of `r1` and `r2`'s preference for `r3`'s RID over `r4` with regard to the 10.0/16 route. This is considered normal behavior, so no corrective actions are taken.

Network Discovery Case Study Configuration

The complete case study configuration for all routers in the test bed is provided next. To keep things interesting, the configuration examples shown in subsequent chapters may use either the OSPF or the IS-IS baseline configuration. Differences between the chapter body and case study configurations, and any changes needed to provide proper IGP operation, are called out with highlights in Listings 1.8 through 1.14.

Listing 1.8: *r1* IS-IS Baseline Configuration

```
lab@r1# show | no-more
version 5.6R1.3;
system {
```

```
    host-name r1;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r1-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/0/0 {
        unit 0 {
```

```
            family inet {
                address 10.0.5.1/24;
            }
            family iso;
        }
    }
    fe-0/0/1 {
        unit 0 {
            family inet {
                address 10.0.4.14/30;
            }
            family iso;
        }
    }
    fe-0/0/2 {
        unit 0 {
            family inet {
                address 10.0.4.5/30;
            }
            family iso;
        }
    }
    fe-0/0/3 {
        unit 0 {
            family inet {
                address 10.0.4.18/30;
            }
            family iso;
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 10.0.1.1/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.6.1/32;
```

```
            }
            family iso {
                address 49.0001.1111.1111.1111.00;
            }
        }
    }
}
routing-options {
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.0.6.1;
            neighbor 10.0.6.2;
            neighbor 10.0.3.3;
            neighbor 10.0.3.4;
            neighbor 10.0.3.5;
            neighbor 10.0.9.6;
            neighbor 10.0.9.7;
        }
        group p1 {
            type external;
            export ebgp-out;
            neighbor 10.0.5.254 {
                peer-as 65050;
            }
        }
    }
    isis {
        level 2 disable;
        interface fe-0/0/0.0 {
            passive;
        }
```

```
        interface fe-0/0/1.0;
        interface fe-0/0/2.0;
        interface fe-0/0/3.0;
        interface lo0.0;
    }
}
policy-options {
    policy-statement ebgp-out {
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then accept;
        }
        term 2 {
            from community transit;
            then reject;
        }
    }
    community transit members 65412:420;
}
```

Note that the 10.0/16 local aggregate has been deleted from the `routing-options` stanza on `r1`, and that the first term in the *`ebgp-out`* policy is no longer needed; the term has been left in place because it is causing no harm.

Listing 1.9: *r2* IS-IS Baseline Configuration

```
[edit]
lab@r2# show | no-more
version 5.6R1.3;
system {
    host-name r2;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
```

```
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r2-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/0/0 {
        unit 0 {
            family inet {
                address 10.0.5.2/24;
            }
            family iso;
        }
    }
    fe-0/0/1 {
        unit 0 {
            family inet {
                address 10.0.4.10/30;
```

```
            }
            family iso;
        }
    }
    fe-0/0/2 {
        speed 100m;
        unit 0 {
            family inet {
                address 10.0.4.2/30;
            }
            family iso;
        }
    }
    fe-0/0/3 {
        unit 0 {
            family inet {
                address 10.0.4.6/30;
            }
            family iso;
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 10.0.1.2/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.6.2/32;
            }
            family iso {
                address 49.0001.2222.2222.2222.00;
            }
        }
    }
}
routing-options {
    static {
```

```
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.0.6.2;
            neighbor 10.0.6.1;
            neighbor 10.0.3.3;
            neighbor 10.0.3.4;
            neighbor 10.0.3.5;
            neighbor 10.0.9.6;
            neighbor 10.0.9.7;
        }
        group p1 {
            type external;
            export ebgp-out;
            neighbor 10.0.5.254 {
                peer-as 65050;
            }
        }
    }
    isis {
        level 2 disable;
        interface fe-0/0/0.0 {
            passive;
        }
        interface fe-0/0/1.0;
        interface fe-0/0/2.0;
        interface fe-0/0/3.0;
        interface lo0.0;
    }
}
policy-options {
    policy-statement ebgp-out {
        term 1 {
```

```
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then accept;
        }
        term 2 {
            from community transit;
            then reject;
        }
    }
    community transit members 65412:420;
}
```

Note that the 10.0/16 local aggregate has been deleted from the `routing-options` stanza on `r2`, and that the first term in the *`ebgp-out`* policy is no longer needed; the term has been left in place because it is causing no harm.

Listing 1.10: *r3* IS-IS Baseline Configuration

```
[edit]
lab@r3# show | no-more
version 5.6R1.3;
system {
    host-name r3;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
```

```
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r3-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/0/0 {
        unit 0 {
            family inet {
                address 10.0.4.13/30;
            }
            family iso;
        }
    }
    fe-0/0/1 {
        unit 0 {
            family inet {
                address 10.0.4.1/30;
            }
            family iso;
        }
    }
    fe-0/0/2 {
        unit 0 {
            family inet {
```

```
            address 172.16.0.13/30;
        }
    }
}
fe-0/0/3 {
    unit 0 {
        family inet {
            address 10.0.2.14/30;
        }
        family iso;
    }
}
at-0/1/0 {
    atm-options {
        vpi 0 {
            maximum-vcs 64;
        }
    }
    unit 0 {
        point-to-point;
        vci 50;
        family inet {
            address 10.0.2.2/30;
        }
        family iso;
    }
}
so-0/2/0 {
    dce;
    encapsulation frame-relay;
    unit 100 {
        dlci 100;
        family inet {
            address 10.0.2.5/30;
        }
        family iso;
    }
}
fxp0 {
    unit 0 {
        family inet {
```

```
                address 10.0.1.3/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.3.3/32;
            }
            family iso {
                address 49.0001.3333.3333.3333.00;
            }
        }
    }
}
routing-options {
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    aggregate {
        route 10.0.0.0/16;
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        advertise-inactive;
        group int {
            type internal;
            local-address 10.0.3.3;
            export nhs;
            neighbor 10.0.6.1 {
                export r1;
            }
            neighbor 10.0.6.2 {
                export r2;
            }
            neighbor 10.0.3.4;
```

```
            neighbor 10.0.3.5;
            neighbor 10.0.9.6;
            neighbor 10.0.9.7;
        }
        group ext {
            import ebgp-in;
            export ebgp-out;
            neighbor 172.16.0.14 {
                peer-as 65222;
            }
        }
    }
    isis {
        interface fe-0/0/0.0 {
            level 2 disable;
        }
        interface fe-0/0/1.0 {
            level 2 disable;
        }
        interface fe-0/0/3.0 {
            level 1 disable;
        }
        interface at-0/1/0.0 {
            level 1 disable;
        }
        interface so-0/2/0.100 {
            level 1 disable;
        }
        interface lo0.0 {
            level 1 disable;
        }
    }
}
policy-options {
    policy-statement nhs {
        term 1 {
            from {
                protocol bgp;
                neighbor 172.16.0.14;
            }
            then {
```

```
            next-hop self;
        }
    }
}
policy-statement ebgp-out {
    term 1 {
        from {
            protocol aggregate;
            route-filter 10.0.0.0/16 exact;
        }
        then accept;
    }
}
policy-statement ebgp-in {
    term 1 {
        from {
            protocol bgp;
            neighbor 172.16.0.14;
        }
        then {
            community add transit;
        }
    }
}
policy-statement r1 {
    term 1 {
        from {
            protocol aggregate;
            route-filter 10.0.0.0/16 exact;
        }
        then {
            next-hop 10.0.4.13;
            accept;
        }
    }
    term 2 {
        from {
            protocol bgp;
            neighbor 172.16.0.14;
        }
        then {
```

```
                next-hop self;
            }
        }
    }
    policy-statement r2 {
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then {
                next-hop 10.0.4.1;
                accept;
            }
        }
        term 2 {
            from {
                protocol bgp;
                neighbor 172.16.0.14;
            }
            then {
                next-hop self;
            }
        }
    }
    community transit members 65412:420;
}
```

Note that IBGP export policy changes were made on `r3` to allow advertisement of the 10.0/16 aggregate to `r1` and `r2` through IBGP with the next hop set to an area 0001 address.

Listing 1.11: *r4* IS-IS Baseline Configuration

```
[edit]
lab@r4# show | no-more
version 5.6R1.3;
system {
    host-name r4;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
```

```
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r4-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/0/0 {
        unit 0 {
            family inet {
                address 172.16.0.5/30;
            }
        }
    }
```

```
fe-0/0/1 {
    unit 0 {
        family inet {
            address 10.0.4.9/30;
        }
        family iso;
    }
}
fe-0/0/2 {
    unit 0 {
        family inet {
            address 10.0.4.17/30;
        }
        family iso;
    }
}
fe-0/0/3 {
    unit 0 {
        family inet {
            address 10.0.2.18/30;
        }
        family iso;
    }
}
so-0/1/0 {
    encapsulation frame-relay;
    unit 100 {
        dlci 100;
        family inet {
            address 10.0.2.6/30;
        }
        family iso;
    }
}
so-0/1/1 {
    encapsulation ppp;
    unit 0 {
        family inet {
            address 10.0.2.10/30;
        }
```

```
            family iso;
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 10.0.1.4/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.3.4/32;
            }
            family iso {
                address 49.0001.4444.4444.4444.00;
            }
        }
    }
}
routing-options {
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    aggregate {
        route 10.0.0.0/16;
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        advertise-inactive;
        group int {
            type internal;
            local-address 10.0.3.4;
            export nhs;
```

```
            neighbor 10.0.6.1 {
                export r1;
            }
            neighbor 10.0.6.2 {
                export r2;
            }
            neighbor 10.0.3.3;
            neighbor 10.0.3.5;
            neighbor 10.0.9.6;
            neighbor 10.0.9.7;
        }
        group c1 {
            type external;
            export ebgp-out;
            neighbor 172.16.0.6 {
                peer-as 65010;
            }
        }
    }
    isis {
        interface fe-0/0/1.0 {
            level 2 disable;
        }
        interface fe-0/0/2.0 {
            level 2 disable;
        }
        interface fe-0/0/3.0 {
            level 1 disable;
        }
        interface so-0/1/0.100 {
            level 1 disable;
        }
        interface so-0/1/1.0 {
            level 1 disable;
        }
        interface lo0.0 {
            level 1 disable;
        }
    }
}
policy-options {
```

```
policy-statement ebgp-out {
    term 1 {
        from {
            protocol aggregate;
            route-filter 10.0.0.0/16 exact;
        }
        then accept;
    }
}
policy-statement nhs {
    term 1 {
        from {
            protocol bgp;
            neighbor 172.16.0.6;
        }
        then {
            next-hop self;
        }
    }
}
policy-statement r1 {
    term 1 {
        from {
            protocol aggregate;
            route-filter 10.0.0.0/16 exact;
        }
        then {
            next-hop 10.0.4.17;
            accept;
        }
    }
    term 2 {
        from {
            protocol bgp;
            neighbor 172.16.0.6;
        }
        then {
            next-hop self;
        }
    }
}
```

```
    policy-statement r2 {
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then {
                next-hop 10.0.4.9;
                accept;
            }
        }
        term 2 {
            from {
                protocol bgp;
                neighbor 172.16.0.6;
            }
            then {
                next-hop self;
            }
        }
    }
}
```

Note that IBGP export policy changes were made on r4 to allow advertisement of the 10.0/16 aggregate to r1 and r2 through IBGP with the next hop set to an area 0001 address.

Listing 1.12: *r5* IS-IS Baseline Configuration

```
[edit]
lab@r5# show | no-more
version 5.6R1.3;
system {
    host-name r5;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
```

```
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r5-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/0/0 {
        unit 0 {
            family inet {
                address 10.0.8.6/30;
            }
            family iso;
        }
    }
    fe-0/0/1 {
        unit 0 {
            family inet {
```

```
            address 10.0.8.9/30;
        }
        family iso;
    }
}
so-0/1/0 {
    encapsulation ppp;
    unit 0 {
        family inet {
            address 10.0.2.9/30;
        }
        family iso;
    }
}
at-0/2/1 {
    atm-options {
        vpi 0 {
            maximum-vcs 64;
        }
    }
    unit 0 {
        point-to-point;
        vci 50;
        family inet {
            address 10.0.2.1/30;
        }
        family iso;
    }
}
fxp0 {
    unit 0 {
        family inet {
            address 10.0.1.5/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.0.3.5/32;
```

```
            }
            family iso {
                address 49.0002.5555.5555.5555.00;
            }
        }
    }
}
routing-options {
    static {
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.0.3.5;
            neighbor 10.0.6.1;
            neighbor 10.0.6.2;
            neighbor 10.0.3.3;
            neighbor 10.0.3.4;
            neighbor 10.0.9.6;
            neighbor 10.0.9.7;
        }
    }
    isis {
        export l1-l2;
        interface fe-0/0/0.0 {
            level 2 disable;
        }
        interface fe-0/0/1.0 {
            level 2 disable;
        }
        interface so-0/1/0.0 {
            level 1 passive;
        }
```

```
        interface at-0/2/1.0 {
            level 1 passive;
        }
        interface lo0.0;
    }
}
policy-options {
    policy-statement l1-l2 {
        term 1 {
            from {
                protocol isis;
                level 1;
                route-filter 192.168.0.0/22 longer;
            }
            to level 2;
            then accept;
        }
    }
}
```

Listing 1.13: *r6* IS-IS Baseline Configuration

```
[edit]
lab@r6# show | no-more
version 5.6R1.3;
system {
    host-name r6;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
```

```
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r6-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/1/0 {
        unit 0 {
            family inet {
                address 10.0.8.5/30;
            }
            family iso;
        }
    }
    fe-0/1/1 {
        unit 0 {
            family inet {
                address 10.0.2.13/30;
            }
            family iso;
        }
    }
```

```
    fe-0/1/2 {
        unit 0 {
            family inet {
                address 10.0.8.2/30;
            }
            family iso;
        }
    }
    fe-0/1/3 {
        unit 0 {
            family inet {
                address 172.16.0.9/30;
            }
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 10.0.1.6/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.9.6/32;
            }
            family iso {
                address 49.0002.6666.6666.6666.00;
            }
        }
    }
}
routing-options {
    static {
        route 0.0.0.0/0 reject;
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
```

```
    aggregate {
        route 10.0.0.0/16;
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.0.9.6;
            export nhs;
            neighbor 10.0.6.1;
            neighbor 10.0.6.2;
            neighbor 10.0.3.3;
            neighbor 10.0.3.4;
            neighbor 10.0.3.5;
            neighbor 10.0.9.7;
        }
        group c2 {
            type external;
            export ebgp-out;
            neighbor 172.16.0.10 {
                peer-as 65020;
            }
        }
    }
    isis {
        export rip-isis;
        interface fe-0/1/0.0 {
            level 2 disable;
        }
        interface fe-0/1/1.0 {
            level 1 disable;
        }
        interface fe-0/1/2.0 {
            level 2 disable;
            level 1 passive;
        }
        interface lo0.0;
    }
```

```
    rip {
        group dc {
            export static-rip;
            neighbor fe-0/1/2.0;
        }
    }
}
policy-options {
    policy-statement static-rip {
        term 1 {
            from {
                protocol static;
                route-filter 0.0.0.0/0 exact;
            }
            then accept;
        }
    }
    policy-statement rip-isis {
        term 1 {
            from {
                protocol rip;
                route-filter 192.168.0.0/22 longer;
            }
            then accept;
        }
    }
    policy-statement ebgp-out {
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then accept;
        }
        term 2 {
            from {
                route-filter 192.168.0.0/22 upto /24;
            }
            then accept;
        }
    }
```

```
    policy-statement nhs {
        term 1 {
            from {
                protocol bgp;
                neighbor 172.16.0.10;
            }
            then {
                next-hop self;
            }
        }
        term 2 {
            from {
                route-filter 192.168.0.0/22 longer;
            }
            then accept;
        }
    }
}
```

Listing 1.14: *r7* IS-IS Baseline Configuration

```
[edit]
lab@r7# show | no-more
version 5.6R1.3;
system {
    host-name r7;
    authentication-order [ radius password ];
    ports {
        console type vt100;
    }
    root-authentication {
        encrypted-password "$1$RTyGDGYG$ukqr37VGRgtohedSlruOk/"; # SECRET-DATA
    }
    radius-server {
        10.0.1.201 secret "$9$jvkmT69pRhrz3hrev7Nik."; # SECRET-DATA
    }
    login {
        user lab {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password "$1$L6ZKKWYI$GxEI/7YzXes2JXDcHJvz7/";
                   # SECRET-DATA
```

```
            }
        }
    }
    services {
        ssh;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file r7-cli {
            interactive-commands any;
            archive files 5;
        }
    }
}
interfaces {
    fe-0/3/0 {
        unit 0 {
            family inet {
                address 10.0.8.14/30;
            }
            family iso;
        }
    }
    fe-0/3/1 {
        unit 0 {
            family inet {
                address 10.0.8.10/30;
            }
            family iso;
        }
    }
    fe-0/3/2 {
        unit 0 {
```

```
            family inet {
                address 172.16.0.1/30;
            }
        }
    }
    fe-0/3/3 {
        unit 0 {
            family inet {
                address 10.0.2.17/30;
            }
            family iso;
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 10.0.1.7/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.0.9.7/32;
            }
            family iso {
                address 49.0002.7777.7777.7777.00;
            }
        }
    }
}
routing-options {
    static {
        route 0.0.0.0/0 reject;
        route 10.0.200.0/24 {
            next-hop 10.0.1.102;
            no-readvertise;
        }
    }
    aggregate {
```

```
        route 10.0.0.0/16;
    }
    autonomous-system 65412;
}
protocols {
    bgp {
        group int {
            type internal;
            local-address 10.0.9.7;
            export nhs;
            neighbor 10.0.6.1;
            neighbor 10.0.6.2;
            neighbor 10.0.3.3;
            neighbor 10.0.3.4;
            neighbor 10.0.3.5;
            neighbor 10.0.9.6;
        }
        group c1 {
            type external;
            export ebgp-out;
            neighbor 172.16.0.2 {
                peer-as 65010;
            }
        }
    }
    isis {
        export rip-isis;
        interface fe-0/3/0.0 {
            level 2 disable;
            level 1 passive;
        }
        interface fe-0/3/1.0 {
            level 2 disable;
        }
        interface fe-0/3/3.0 {
            level 1 disable;
        }
        interface lo0.0;
    }
    rip {
        group dc {
```

```
            export static-rip;
            neighbor fe-0/3/0.0;
        }
    }
}
policy-options {
    policy-statement static-rip {
        term 1 {
            from {
                protocol static;
                route-filter 0.0.0.0/0 exact;
            }
            then accept;
        }
    }
    policy-statement rip-isis {
        term 1 {
            from {
                protocol rip;
                route-filter 192.168.0.0/22 longer;
            }
            then accept;
        }
    }
    policy-statement ebgp-out {
        term 1 {
            from {
                protocol aggregate;
                route-filter 10.0.0.0/16 exact;
            }
            then accept;
        }
        term 2 {
            from {
                route-filter 192.168.0.0/22 upto /24;
            }
            then accept;
        }
    }
    policy-statement nhs {
```

```
        term 1 {
            from {
                protocol bgp;
                neighbor 172.16.0.2;
            }
            then {
                next-hop self;
            }
        }
        term 2 {
            from {
                route-filter 192.168.0.0/22 longer;
            }
            then accept;
        }
    }
}
```

Spot the Issues: Review Questions

1. Referring back to Figure 1.5, and the configuration snippet below, describe the number and type of adjacencies that you expect to find on r5. You may assume that r3, r4, r6, and r7 have a similar configuration:

```
[edit protocols isis]
lab@r5# show
export l1-l2;
level 1 disable;
interface fe-0/0/0.0;
interface fe-0/0/1.0;
interface so-0/1/0.0;
interface at-0/2/1.0;
interface lo0.0;
```

2. r5 has no IS-IS adjacencies. Can you spot the problems from the following configuration snippets?

```
[edit]
lab@r5# run show isis interface
IS-IS interface database:
Interface       L CirID Level 1 DR         Level 2 DR        L1/L2 Metric
at-0/2/1.0      2   0x1 Disabled           Point to Point         10/10
fe-0/0/0.0      1   0x2 0000.0000.0000.02  Disabled               10/10
fe-0/0/1.0      1   0x3 0000.0000.0000.03  Disabled               10/10
lo0.0           0   0x1 Passive            Passive                 0/0
so-0/1/0.0      2   0x1 Disabled           Point to Point         10/10

[edit]
lab@r5# run show isis adjacency

[edit]
lab@r5# show interfaces
fe-0/0/0 {
    unit 0 {
        family inet {
            address 10.0.8.6/30;
        }
        family iso;
    }
}
```

```
fe-0/0/1 {
    unit 0 {
        family inet {
            address 10.0.8.9/30;
        }
        family iso;
    }
}
so-0/1/0 {
    encapsulation ppp;
    unit 0 {
        family inet {
            address 10.0.2.9/30;
        }
        family iso;
    }
}
at-0/2/1 {
    atm-options {
        vpi 0 {
            maximum-vcs 64;
        }
    }
    unit 0 {
        point-to-point;
        vci 50;
        family inet {
            address 10.0.2.1/30;
        }
        family iso;
    }
}
fxp0 {
    unit 0 {
        family inet {
            address 10.0.1.5/24;
        }
    }
}
lo0 {
```

```
    unit 0 {
        family inet {
            address 10.0.3.5/32;
        }
        family iso;
    }
}
[edit]
lab@r5# show protocols isis
export l1-l2;
interface fe-0/0/0.0 {
    level 2 disable;
}
interface fe-0/0/1.0 {
    level 2 disable;
}
interface so-0/1/0.0 {
    level 1 disable;
}
interface at-0/2/1.0 {
    level 1 disable;
}
interface lo0.0;
```

3. Based on the topology demonstrated in this chapter and the output shown next, do you expect that all traffic generated by the data center router will follow an optimal path?

```
lab@dc> show route 0/0

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

0.0.0.0/0          *[RIP/100] 01:23:29, metric 2, tag 0
                      to 10.0.8.2 via fe-0/0/0.0
                    > to 10.0.8.14 via fe-0/0/1.0
```

4. Explain why the locally defined 10.0/16 aggregate on r1 and r2 worked fine with OSPF but not with IS-IS in the baseline topology.

5. Make at least three observations from the OSPF stanza shown next:

```
[edit protocols ospf]
lab@r3# show
area 0.0.0.0 {
```

```
        area-range 10.0.2.0/23;
        area-range 10.0.3.3/32 restrict;
        authentication-type md5; # SECRET-DATA
        interface at-0/1/0.0 {
            authentication-key "$9$zKS-n9peK8X7V"; # SECRET-DATA
        }
        interface lo0.0;
    }
    area 0.0.0.1 {
        nssa;
        interface all;
    }
```

Spot the Issues: Answers to Review Questions

1. Because Level 1 has been disabled in the IS-IS instance, you should expect to see a total of four Level 2 adjacencies at `r5`.

2. The problem with `r5`'s IS-IS configuration is the lack of a configured ISO NET. Although lo0 is considered an IS-IS interface, and is defined in the IS-IS stanza, a NET is required for proper IS-IS operation.

3. No. With the DC router receiving two equal-cost next hops for the default route, you should expect to see that some traffic incurs extra hops through either `r6` or `r7`, depending on which next hop is installed at any given time.

4. The key to the differing behaviors lies in the fact that network summaries (LSA Type 3s) were permitted in the OSPF stub area. The network summaries resulted in `r1` and `r2` learning about specific routes to all in-use 10.0/16 addresses. The 10.0/16 aggregate was never used due to the presence of the more specific routes, and therefore the presence of the aggregate did not cause any black holes. IS-IS, on the other hand, does not support the concept of network summaries, which makes an IS-IS Level 1 area function like an OSPF stub area with no summaries. Lacking summary routes, the 10.0/16 aggregate became the longest match for destinations outside of the Level 1 area. The reject next hop associated with the 10.0/16 aggregate route therefore resulted in a black hole.

5. Based on this OSPF stanza, you can determine the following:

   - `r3` is an ABR serving areas 0 and 1.
   - MD5-based authentication is in place in area 0 but not area 1.
   - Area 0 routes matching 10.0.2.0/23 will be presented to area 1 as a single network summary.
   - Area 1 will not have a summary route for the 10.0.3.3 loopback address of `r3` due to the `restrict` keyword.
   - All interfaces on `r3`, except the at-0/1/0 and the lo0 interfaces, have been placed into area 1.
   - `r3` will not generate a default route into the NSSA.