


# Chapter 8: Single-Area OSPF

## Objectives:

- Configuring and verifying basic OSPF with a single-area design.
- Configuring and verifying basic OSPF with a multi-area design.
- Configuring OSPFv2 with subcommands, OSPF default routes, OSPF tuning and load balancing.

## 1. OSPF Concepts

The Open Shortest Path First (OSPF) protocol, defined in [RFC 2328](#) , is an Interior Gateway Protocol used to distribute routing information within a single Autonomous System. This paper examines how OSPF works and how it can be used to design and build large and complicated networks.

### Background Information

OSPF protocol was developed due to a need in the internet community to introduce a high functionality non-proprietary Internal Gateway Protocol (IGP) for the TCP/IP protocol family. The discussion of the creation of a common interoperable IGP for the Internet started in 1988 and did not get formalized until 1991. At that time the OSPF Working Group requested that OSPF be considered for advancement to Draft Internet Standard.

The OSPF protocol is based on link-state technology, which is a departure from the Bellman-Ford vector based algorithms used in traditional Internet routing protocols such as RIP. OSPF has introduced new concepts such as authentication of routing updates, Variable Length Subnet Masks (VLSM), route summarization, and so forth.

These chapter discuss the OSPF terminology, algorithm and the pros and cons of the protocol in designing the large and complicated networks of today.

### OSPF versus RIP

The rapid growth and expansion of today's networks has pushed RIP to its limits. RIP has certain limitations that can cause problems in large networks:

- RIP has a limit of 15 hops. A RIP network that spans more than 15 hops (15 routers) is considered unreachable.
- RIP cannot handle Variable Length Subnet Masks (VLSM). Given the shortage of IP addresses and the flexibility VLSM gives in the efficient assignment of IP addresses, this is considered a major flaw.
- Periodic broadcasts of the full routing table consume a large amount of bandwidth. This is a major problem with large networks especially on slow links and WAN clouds.

- RIP converges slower than OSPF. In large networks convergence gets to be in the order of minutes. RIP routers go through a period of a hold-down and garbage collection and slowly time-out information that has not been received recently. This is inappropriate in large environments and could cause routing inconsistencies.
- RIP has no concept of network delays and link costs. Routing decisions are based on hop counts. The path with the lowest hop count to the destination is always preferred even if the longer path has a better aggregate link bandwidth and less delays.
- RIP networks are flat networks. There is no concept of areas or boundaries. With the introduction of classless routing and the intelligent use of aggregation and summarization, RIP networks seem to have fallen behind.

Some enhancements were introduced in a new version of RIP called RIP2. RIP2 addresses the issues of VLSM, authentication, and multicast routing updates. RIP2 is not a big improvement over RIP (now called RIP 1) because it still has the limitations of hop counts and slow convergence which are essential in today's large networks.

OSPF, on the other hand, addresses most of the issues previously presented:

- With OSPF, there is no limitation on the hop count.
- The intelligent use of VLSM is very useful in IP address allocation.
- OSPF uses IP multicast to send link-state updates. This ensures less processing on routers that are not listening to OSPF packets. Also, updates are only sent in case routing changes occur instead of periodically. This ensures a better use of bandwidth.
- OSPF has better convergence than RIP. This is because routing changes are propagated instantaneously and not periodically.
- OSPF allows for better load balancing.
- OSPF allows for a logical definition of networks where routers can be divided into areas. This limits the explosion of link state updates over the whole network. This also provides a mechanism for aggregating routes and cutting down on the unnecessary propagation of subnet information.
- OSPF allows for routing authentication by using different methods of password authentication.
- OSPF allows for the transfer and tagging of external routes injected into an Autonomous System. This keeps track of external routes injected by exterior protocols such as BGP.

This of course leads to more complexity in the configuration and troubleshooting of OSPF networks. Administrators that are used to the simplicity of RIP are challenged with the amount of new information they have to learn in order to keep up with OSPF networks. Also, this introduces more overhead in memory allocation and CPU utilization. Some of the routers running RIP might have to be upgraded in order to handle the overhead caused by OSPF.

### **What Do We Mean by Link-States?**

OSPF is a link-state protocol. We could think of a link as being an interface on the router. The state of the link is a description of that interface and of its relationship to its neighboring routers. A description of the interface would include, for example, the IP address of the interface, the mask, the type of network it is connected to, the routers connected to that network and so on. The collection of all these link-states would form a link-state database.

## Shortest Path First Algorithm

OSPF uses a shortest path first algorithm in order to build and calculate the shortest path to all known destinations. The shortest path is calculated with the use of the Dijkstra algorithm. The algorithm by itself is quite complicated. This is a very high level, simplified way of looking at the various steps of the algorithm:

1. Upon initialization or due to any change in routing information, a router generates a link-state advertisement. This advertisement represents the collection of all link-states on that router.
2. All routers exchange link-states by means of flooding. Each router that receives a link-state update should store a copy in its link-state database and then propagate the update to other routers.
3. After the database of each router is completed, the router calculates a Shortest Path Tree to all destinations. The router uses the Dijkstra algorithm in order to calculate the shortest path tree. The destinations, the associated cost and the next hop to reach those destinations form the IP routing table.
4. In case no changes in the OSPF network occur, such as cost of a link or a network being added or deleted, OSPF should be very quiet. Any changes that occur are communicated through link-state packets, and the Dijkstra algorithm is recalculated in order to find the shortest path.

The algorithm places each router at the root of a tree and calculates the shortest path to each destination based on the cumulative cost required to reach that destination. Each router will have its own view of the topology even though all the routers will build a shortest path tree using the same link-state database. The following sections indicate what is involved in building a shortest path tree.

## OSPF Cost

The cost (also called metric) of an interface in OSPF is an indication of the overhead required to send packets across a certain interface. The cost of an interface is inversely proportional to the bandwidth of that interface. A higher bandwidth indicates a lower cost. There is more overhead (higher cost) and time delays involved in crossing a 56k serial line than crossing a 10M ethernet line. The formula used to calculate the cost is:

- $\text{cost} = 100000000 / \text{bandwidth in bps}$

For example, it will cost  $10^8 / 10^7 = 10$  to cross a 10M Ethernet line and will cost  $10^8 / 1544000 = 64$  to cross a T1 line.

By default, the cost of an interface is calculated based on the bandwidth; you can force the cost of an interface with the **ip ospf cost <value>** interface subconfiguration mode command.

## What is OSPF and How Does it Work?

OSPF is a [Link State protocol](#) that's considered may be the most famous protocol among the Interior Gateway Protocol (IGP) family, developed in the mid 1980's by the OSPF working group of the IETF.

When configured, OSPF will listen to neighbors and gather all link state data available to build a topology map of all available paths in its network and then save the information in its topology database, also known as its **Link-State Database (LSDB)**. Using the information from its topology database. From the information gathered, it will calculate the best shortest path to each reachable subnet/network using an algorithm called **Shortest Path First (SPF)** that was developed by the computer scientist *Edsger W. Dijkstra* in 1956. OSPF will then construct **three tables** to store the following information:

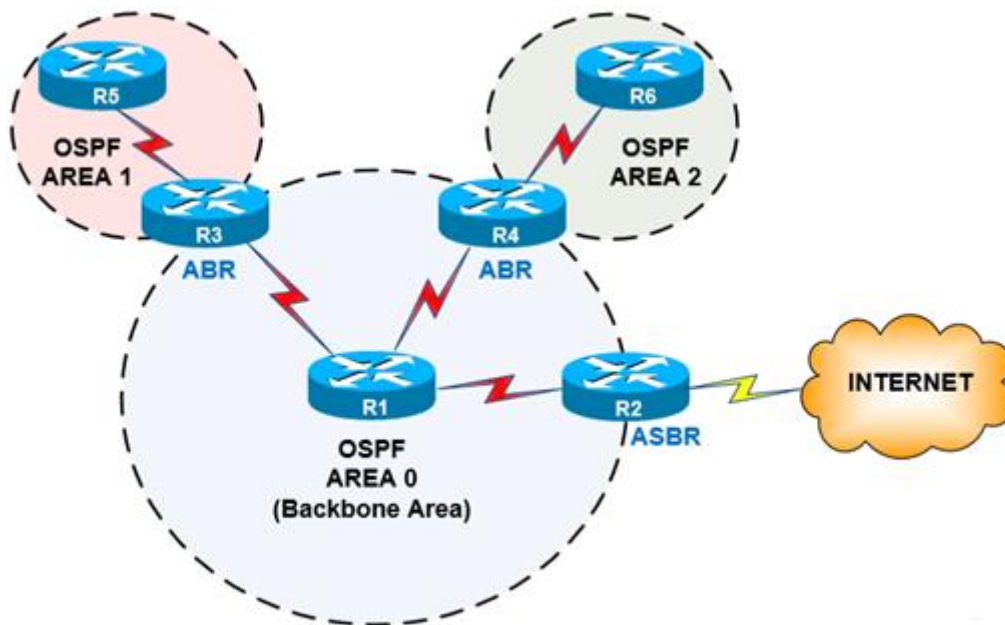
- **Neighbor Table:** Contains all discovered OSPF neighbors with whom routing information will be interchanged
- **Topology Table:** Contains the entire road map of the network with all available OSPF routers and calculated best and alternative paths.
- **Routing Table:** Contain the current working best paths that will be used to forward data traffic between neighbors.

## Understanding OSPF Areas

OSPF offers a very distinguishable feature named: **Routing Areas**. It means dividing routers inside a single autonomous system running OSPF, into areas where each area consists of a group of connected routers.

The idea of dividing the OSPF network into areas is to simplify administration and optimize available resources. Resource optimization is especially important for large enterprise networks with a plethora of network and links. Having many routers exchange the link state database could flood the network and reduce its efficiency – this was the need that led to the creation of concept Areas.

Areas are a logical collection of routers that carry the same **Area ID** or number inside of an OSPF network, the OSPF network itself can contain multiple areas, the first and main Area is called the backbone area “**Area 0**”, all other areas must connect to **Area 0** as shown in the diagram below:



All routers within the same **Area** have the **same topology table -Link State Database-** but different **routing table** as OSPF calculates different best paths for each router depending on its location within the network topology while they will all share the same **Link State topology**.

The goal of having an **Area** is to localize the network as follow:

- The **Area boundaries** will give the opportunity of using **summarization**, as it's not possible to summarize network prefixes in normal link state protocols because routers are supposed to have the same map topology of the entire network coincide in all neighbors.
- **Area boundaries** will also help preventing fault containment by suppressing updates that take place when a change occurs in the network causing a flood of updates between routers. This also happens to be a weakness of link state protocols: When connecting large sized networks it is very difficult to avoid link state database floods.

With **Area boundaries**, updates are kept only **inside the same area**, while other areas remain completely unaware of the update.

### OSPF Link State Packet Types

OSPF routers generate packets of information that are exchanged with neighboring routers. These packets are designed for several purposes such as forming neighbor relations between routers, calculating cost and best path for a specific route and more.

The following is a list of the most frequently used OSPF packets:

**Link State Advertisement (LSA):** The primary mean of communication between OSPF routers, it's the packet that **carries all fundamental information about the topology** and is flooded between areas to perform different functions, there are **11 types of LSA packets** that will be covered in great depth in future OSPF articles.

**Link State DataBase (LSDB):** LSDB packet contains all updated link-state information exchanged among the network, and all routers within the same area have **identical LSDB**, and when two routers form new neighbor adjacency, they sync their **LSDB** to be **fully adjacent**.

**Link State Request (LSR):** Once neighbor adjacency is formed and **LSDB** is exchanged, neighbor routers may locate a missing LSDB information, they then send a request packet to claim the missing piece, neighbors receive this packet and respond with **LSU**.

**Link State Update (LSU):** A response packet sends a specific piece of **LSDB** information requested by an OSPF neighbor via **LSR** packet.

**Link State Acknowledgment (LSAck):** The router that sends the **LSR** packet confirms receiving the **LSU** from neighbor by sending a confirmation packet acknowledging receiving the requested **LSUs**.

### **Working Inside of a Single Area**

Working inside of an Area is hierarchically organized among routers that share this area and are categorized as:

#### **Area Boarder Routers (ABR):**

Routers located on the borders of each **Area** connect to more than one OSPF area, are called **ABR Routers**. **ABR Routers** are responsible for **summarizing IP addresses** of each area and **suppressing updates** among areas to prevent fault containment.

#### **Autonomous System Boundary Router (ASBR):**

An **ASBR** is a router that has interfaces connected to one or more OSPF areas, similarly as the **ABR**, however the difference with an **ASBR** is that it also connects to **other routing systems** such as **BGP, EIGRP, Internet** and others. An **ASBR** router normally advertises routes from other routing systems **into the OSPF area** to which it belongs.

#### **Designated Router (DR):**

A **Designated Router** is elected by the routers on multi-access segments (e.g Local Area Network), based on its priority (Router ID, priority). The **DR** router performs special functions such as generating **Link State Advertisements (LSAs)** and exchanging information with all other routers in the same **Area**. Every router in the same **Area** will create an adjacency with the **DR** and **BDR** (analysed below).

The DR sends updates to all **Area** routers using the **Multicast address 224.0.0.5**. All OSPF routers except the **DR** use **Multicast address 224.0.0.6** to send **Link State Update (LSU)** and **Link State Advertisements (LSAs)** packets to the **DR**.

#### **Backup Designated Router (BDR):**

The **BDR** is a router that becomes the **DR** should the existing **DR** fail. The BDR has the second highest priority (the DR having the highest priority) in the OSPF network. When the **BDR** becomes a **DR**, a new election is made to find a new **BDR**.



## 2. Implementing Single-Area OSPFv2 to the network

OSPF configuration includes only a few required steps, but it has many optional steps. After an OSPF design has been chosen—a task that can be complex in larger IP internetworks—the configuration can be as simple as enabling OSPF on each router interface and placing that interface in the correct OSPF area.

The following is a list of reasons OSPF is considered a better routing protocol than RIP:

- OSPF has no hop count limitations. (RIP has 15 hops only.)
- OSPF understands variable-length subnet masks (VLSMs) and allows for summarization.
- OSPF uses multicasts (not broadcasts) to send updates.
- OSPF converges much faster than RIP, because OSPF propagates changes immediately.
- OSPF allows for load balancing with up to six equal-cost paths.
- OSPF has authentication available. (RIPv2 does also, but RIPv1 does not.)
- OSPF allows for tagging of external routes injected by other autonomous systems.
- OSPF configuration, monitoring, and troubleshooting have a far greater IOS tool base than RIP.



OSPF does have some disadvantages, including the level of difficulty and understanding required to configure, monitor, and troubleshoot it. The other two factors are the memory and Central Processing Unit (CPU) requirements that can affect even high-end router performance. You can configure more than one OSPF process, but you must be mindful that the SPF calculations associated with multiple OSPF processes can consume a considerable amount of CPU and memory.

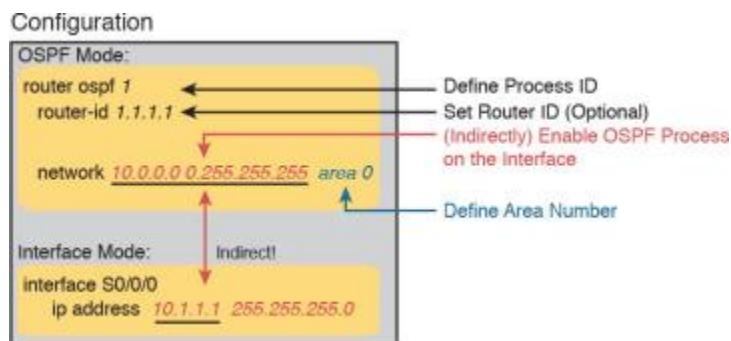
This section shows several configuration examples, all with a single-area OSPF internetwork. Following those examples, the text goes on to cover several of the additional optional configuration settings. For reference, the following list outlines the configuration steps covered in this first major section of the chapter, as well as a brief reference to the required commands:



- **Step 1.** Use the **router ospf** *process-id* global command to enter OSPF configuration mode for a particular OSPF process.
- **Step 2.** (Optional) Configure the OSPF router ID by doing the following:
  - **A.** Use the **router-id** *id-value* router subcommand to define the router ID

- **B.** Use the **interface loopback number** global command, along with an **ip address address mask** command, to configure an IP address on a loopback interface (chooses the highest IP address of all working loopbacks)
- **C.** Rely on an interface IP address (chooses the highest IP address of all working nonloopbacks)
- **Step 3.** Use one or more **network ip-address wildcard-mask area area-id** router subcommands to enable OSPFv2 on any interfaces matched by the configured address and mask, enabling OSPF on the interface for the listed area.
- **Step 4.** (Optional) Use the **passive-interface type number** router subcommand to configure any OSPF interfaces as passive if no neighbors can or should be discovered on the interface.

For a more visual perspective on OSPFv2 configuration, [Figure 8-1](#) shows the relationship between the key OSPF configuration commands. Note that the configuration creates a routing process in one part of the configuration, and then indirectly enables OSPF on each interface. The configuration does not name the interfaces on which OSPF is enabled, instead requiring IOS to apply some logic by comparing the OSPF **network** command to the interface **ip address** commands. The upcoming example discusses more about this logic.

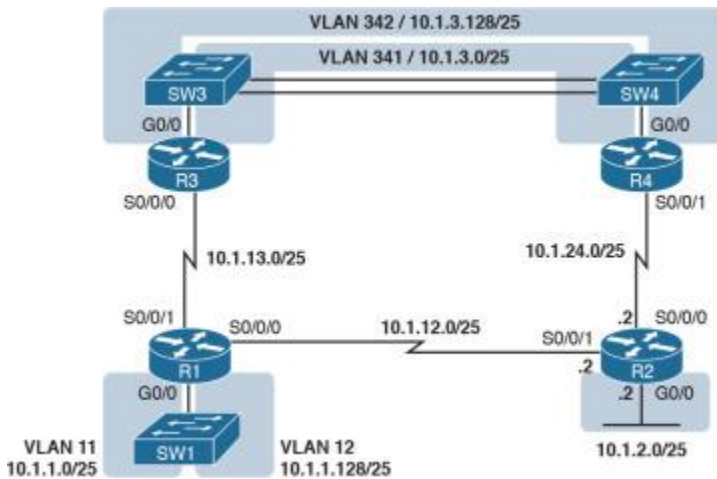


**Figure 8-1** Organization of OSPFv2 Configuration

## OSPF Single-Area Configuration

[Figure 8-1](#) shows a sample network that will be used for the single-area OSPF configuration examples. All links sit in area 0. The design has four routers, each connected to one or two LANs. However, note that Routers R3 and R4, at the top of the figure, connect to the same two VLANs/subnets, so they will form neighbor relationships with each other over each of those VLANs as well. (The two switches at the top of the design are acting as Layer 2 switches.)





**Figure 8-1** Sample Network for OSPF Single-Area Configuration

Example 8-1 shows the IPv4 addressing configuration on Router R3, before getting into the OSPF detail. The configuration enables 802.1Q trunking on R3's G0/0 interface, and assigns an IP address to each subinterface. (Not shown, switch S3 has configured trunking on the other side of that Ethernet link.)

#### Example 8-1 IPv4 Address Configuration on R3 (Including VLAN Trunking)

```
interface GigabitEthernet 0/0.341
 encapsulation dot1q 341
 ip address 10.1.3.1 255.255.255.128
!
interface GigabitEthernet 0/0.342
 encapsulation dot1q 342
 ip address 10.1.3.129 255.255.255.128
!
interface serial 0/0/0
 ip address 10.1.13.3 255.255.255.128
```

The beginning single-area configuration on R3, as shown in Example 8-2, enables OSPF on all the interfaces shown in [Figure 8-2](#). First, the **router ospf 1** global command puts the user in OSPF configuration mode, and sets the OSPF *process-id*. This number just needs to be unique on the local router, allowing the router to support multiple OSPF processes in a single router by using different process IDs. (The **router** command uses the *process-id* to distinguish between the processes.) The *process-id* does not have to match on each router, and it can be any integer between 1 and 65,535.

#### Example 8-2 OSPF Single-Area Configuration on R3 Using One network Command

```
router ospf 1
 network 10.0.0.0 0.255.255.255 area 0
```

Speaking generally rather than about this example, the OSPF **network** command tells a router to find its local interfaces that match the first two parameters on the **network** command. Then, for each matched interface, the

router enables OSPF on those interfaces, discovers neighbors, creates neighbor relationships, and assigns the interface to the area listed in the **network** command. (Note that the area can be configured as either an integer or a dotted-decimal number, but this book makes a habit of configuring the area number as an integer. The integer area numbers range from 0 through 4,294,967,295.)

For the specific command in Example 8-2, any matched interfaces are assigned to area 0. However, the first two parameters—the *ip\_address* and *wildcard\_mask* parameter values of 10.0.0.0 and 0.255.255.255—need some explaining. In this case, the command matches all three interfaces shown for Router R3; the next topic explains why.

### Matching with the OSPF network Command

The key to understanding the traditional OSPFv2 configuration shown in this first example is to understand the OSPF **network** command. The OSPF **network** command compares the first parameter in the command to each interface IP address on the local router, trying to find a match. However, rather than comparing the entire number in the **network** command to the entire IPv4 address on the interface, the router can compare a subset of the octets, based on the wildcard mask, as follows:



- **Wildcard 0.0.0.0:** Compare all 4 octets. In other words, the numbers must exactly match.
- **Wildcard 0.0.0.255:** Compare the first 3 octets only. Ignore the last octet when comparing the numbers.
- **Wildcard 0.0.255.255:** Compare the first 2 octets only. Ignore the last 2 octets when comparing the numbers.
- **Wildcard 0.255.255.255:** Compare the first octet only. Ignore the last 3 octets when comparing the numbers.
- **Wildcard 255.255.255.255:** Compare nothing—this wildcard mask means that all addresses will match the **network** command.

Basically, a wildcard mask value of 0 in an octet tells IOS to compare to see if the numbers match, and a value of 255 tells IOS to ignore that octet when comparing the numbers.

The **network** command provides many flexible options because of the wildcard mask. For example, in Router R3, many **network** commands could be used, with some matching all interfaces, and some matching a subset of interfaces. Table 8-1 shows a sampling of options, with notes.

**Table 8-1 Example OSPF network Commands on R3, with Expected Results**

Command	Logic in Command	Matched Interfaces
<b>network 10.1.0.0 0.0.255.255</b>	Match interface IP addresses that begin with 10.1	G0/0.341 G0/0.342 S0/0/0
<b>network 10.0.0.0 0.255.255.255</b>	Match interface IP addresses that begin with	10 G0/0.341 G0/0.342 S0/0/0
<b>network 0.0.0.0 255.255.255.255</b>	Match all interface IP addresses	G0/0.341 G0/0.342 S0/0/0
<b>network 10.1.13.0 0.0.0.255</b>	Match interface IP addresses that begin with 10.1.13	S0/0/0
<b>network 10.1.3.1 0.0.0.0</b>	Match one IP address: 10.1.3.1	G0/0.341

The wildcard mask gives the local router its rules for matching its own interfaces. For example, Example 8-2 shows R3 using the **network 10.0.0.0 0.255.255.255 area 0** command. However, the wildcard mask allows for many different valid OSPF configurations. For instance, in that same internetwork, Routers R1 and R2 could use the configuration shown in Example 8-3, with two other wildcard masks. In both routers, OSPF is enabled on all the interfaces shown in [Figure 8-2](#).

### Example 8-3 OSPF Configuration on Routers R1 and R2

```
! R1 configuration next - one network command enables OSPF
! on all three interfaces
router ospf 1
 network 10.1.0.0 0.0.255.255 area 0
! R2 configuration next - One network command per interface
router ospf 1
 network 10.1.12.2 0.0.0.0 area 0
 network 10.1.24.2 0.0.0.0 area 0
 network 10.1.2.2 0.0.0.0 area 0
```

Finally, note that other wildcard mask values can be used as well, as long as the wildcard mask in binary is one unbroken string of 0s and another single string of binary 1s. Basically, that includes all wildcard masks that could be used to match all IP addresses in a subnet.



The first two parameters of the **network** command are the address and the wildcard mask. By convention, if the wildcard mask octet is 255, the matching address octet should be configured as a 0. Interestingly, IOS will actually accept a **network** command that breaks this rule, but then IOS will change that octet of the address to a 0 before putting it into the running configuration file. For example, IOS will change a typed command that begins with **network 1.2.3.4 0.0.255.255** to **network 1.2.0.0 0.0.255.255**.

## Verifying OSPFv2 Single Area

As mentioned in Chapter 7, OSPF routers use a three-step process to eventually add OSPF-learned routes to the IP routing table. First, they create neighbor relationships. Then they build and flood LSAs, so each router in the same area has a copy of the same LSDB. Finally, each router independently computes its own IP routes using the SPF algorithm and adds them to its routing table.

The **show ip ospf neighbor**, **show ip ospf database**, and **show ip route** commands display information for each of these three steps, respectively. To verify OSPF, you can use the same sequence. Or, you can just go look at the IP routing table, and if the routes look correct, OSPF probably worked.

For example, first, examine the list of neighbors known on Router R3 from the configuration in Examples 8-1, 8-2, and 8-3. R3 should have one neighbor relationship with R1, over the serial link. It also has two neighbor relationships with R4, over the two different VLANs to which both routers connect. Example 8-4 shows all three.

### Example 8-4 OSPF Neighbors on Router R3 from [Figure 8-2](#)

```
R3# show ip ospf neighbor
```

Neighbor ID	Pri	State	Dead Time	Address	Interface
1.1.1.1	0	FULL/ -	00:00:33	10.1.13.1	Serial0/0/0
10.1.24.4	1	FULL/DR	00:00:35	10.1.3.130	GigabitEthernet0/0.342
10.1.24.4	1	FULL/DR	00:00:36	10.1.3.4	GigabitEthernet0/0.341

The detail in the output mentions several important facts, and for most people, working right to left works best in this case. For example, looking at the headings:

- **Interface:** This is the local router's interface connected to the neighbor. For example, the first neighbor in the list is reachable through R3's S0/0/0 interface.
- **Address:** This is the neighbor's IP address on that link. Again, for this first neighbor, the neighbor, which is R1, uses IP address 10.1.13.1.
- **State:** While many possible states exist, for the details discussed in this chapter, FULL is the correct and fully working state in this case.
- **Neighbor ID:** This is the router ID of the neighbor.

Next, Example 8-5 shows the contents of the LSDB on Router R3. Interestingly, when OSPF is working correctly in an internetwork with a single-area design, all the routers will have the same LSDB contents. So, the **show ip ospf database** command in Example 8-5 should list the same exact information, no matter on which of the four routers it is issued.

### Example 8-5 OSPF Database on Router R3 from [Figure 8-2](#)

R3# **show ip ospf database**

OSPF Router with ID (10.1.13.3) (Process ID 1)

Router Link States (Area 0)

Link ID	ADV Router	Age	Seq#	Checksum	Link count
1.1.1.1	1.1.1.1	498	0x80000006	0x002294	6
2.2.2.2	2.2.2.2	497	0x80000004	0x00E8C6	5
10.1.13.3	10.1.13.3	450	0x80000003	0x001043	4
10.1.24.4	10.1.24.4	451	0x80000003	0x009D7E	4

Net Link States (Area 0)

Link ID	ADV Router	Age	Seq#	Checksum
10.1.3.4	10.1.24.4	451	0x80000001	0x0045F8
10.1.3.130	10.1.24.4	451	0x80000001	0x00546B

For the purposes of this book, do not be concerned about the specifics in the output of this command. However, for perspective, note that the LSDB should list one “Router Link State” (Type 1 Router LSA) for each of the routers in the same area. In this design, all four routers are in the same area, so there are four highlighted Type 1 LSAs listed.

Next, Example 8-6 shows R3’s IPv4 routing table with the **show ip route** command. Note that it lists connected routes as well as OSPF routes. Take a moment to look back at [Figure 8-2](#), and look for the subnets that are not locally connected to R3. Then look for those routes in the output in Example 8-5.

### Example 8-6 IPv4 Routes Added by OSPF on Router R3 from [Figure 8-2](#)

R3# **show ip route**

Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP

D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area

N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2

E1 - OSPF external type 1, E2 - OSPF external type 2

! Legend lines omitted for brevity

```

10.0.0.0/8 is variably subnetted, 11 subnets, 2 masks
O    10.1.1.0/25 [110/65] via 10.1.13.1, 00:13:28, Serial0/0/0
O    10.1.1.128/25 [110/65] via 10.1.13.1, 00:13:28, Serial0/0/0
O    10.1.2.0/25 [110/66] via 10.1.3.130, 00:12:41, GigabitEthernet0/0.342
      [110/66] via 10.1.3.4, 00:12:41, GigabitEthernet0/0.341
C    10.1.3.0/25 is directly connected, GigabitEthernet0/0.341
L    10.1.3.1/32 is directly connected, GigabitEthernet0/0.341
C    10.1.3.128/25 is directly connected, GigabitEthernet0/0.342
L    10.1.3.129/32 is directly connected, GigabitEthernet0/0.342
O    10.1.12.0/25 [110/128] via 10.1.13.1, 00:13:28, Serial0/0/0
C    10.1.13.0/25 is directly connected, Serial0/0/0
L    10.1.13.3/32 is directly connected, Serial0/0/0
O    10.1.24.0/25

```

```
[110/65] via 10.1.3.130, 00:12:41, GigabitEthernet0/0.342
[110/65] via 10.1.3.4, 00:12:41, GigabitEthernet0/0.341
```

First, take a look at the bigger ideas confirmed by this output. The code of “O” on the left identifies a route as being learned by OSPF. The output lists five such IP routes. From [Figure 8-2](#), five subnets exist that are not connected subnets off Router R3. Looking for a quick count of OSPF routes, versus nonconnected routes in the diagram, gives a quick check of whether OSPF learned all routes.

Next, take a look at the first route (to subnet 10.1.1.0/25). It lists the subnet ID and mask, identifying the subnet. It also lists two numbers in brackets. The first, 110, is the administrative distance of the route. All the OSPF routes in this example use the default of 110. The second number, 65, is the OSPF metric for this route.

Additionally, the **show ip protocols** command is also popular as a quick look at how any routing protocol works. This command lists a group of messages for each IPv4 routing protocol running on a router. Example 8-7 shows a sample, this time taken from Router R3.

### Example 8-7 The show ip protocols Command on R3

```
R3# show ip protocols
*** IP Routing is NSF aware ***

Routing Protocol is "ospf 1"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Router ID 10.1.13.3
  Number of areas in this router is 1. 1 normal 0 stub 0 nssa
  Maximum path: 4
  Routing for Networks:
    10.0.0.0 0.255.255.255 area 0
  Routing Information Sources:
    Gateway         Distance      Last Update
    1.1.1.1          110           06:26:17
    2.2.2.2          110           06:25:30
    10.1.24.4        110           06:25:30
  Distance: (default is 110)
```

The output shows several interesting facts. The first highlighted line repeats the parameters on the **router ospf 1** global configuration command. The second highlighted item points out R3’s router ID, as discussed further in the next section. The third highlighted line repeats more configuration, listing the parameters of the **network 10.0.0.0 0.255.255.255 area 0** OSPF subcommand. Finally, the last highlighted item in the example acts as a heading before a list of known OSPF routers, by router ID.

### Configuring the OSPF Router ID

While OSPF has many other optional features, most enterprise networks that use OSPF choose to configure each router’s OSPF router ID. OSPF-speaking routers must have a router ID (RID) for proper operation. By default, routers will choose an interface IP address to use as the RID. However, many network engineers prefer to choose each router’s router ID, so command output from commands like **show ip ospf neighbor** lists more recognizable router IDs.



To choose its RID, a Cisco router uses the following process when the router reloads and brings up the OSPF process. Note that when one of these steps identifies the RID, the process stops.



1. If the **router-id rid** OSPF subcommand is configured, this value is used as the RID.
2. If any loopback interfaces have an IP address configured, and the interface has an interface status of up, the router picks the highest numeric IP address among these loopback interfaces.
3. The router picks the highest numeric IP address from all other interfaces whose interface status code (first status code) is up. (In other words, an interface in up/down state will be included by OSPF when choosing its router ID.)

The first and third criteria should make some sense right away: the RID is either configured or is taken from a working interface's IP address. However, this book has not yet explained the concept of a *loopback interface*, as mentioned in Step 2.

A loopback interface is a virtual interface that can be configured with the **interface loopback interface-number** command, where *interface-number* is an integer. Loopback interfaces are always in an “up and up” state unless administratively placed in a shutdown state. For example, a simple configuration of the command **interface loopback 0**, followed by **ip address 2.2.2.2 255.255.255.0**, would create a loopback interface and assign it an IP address. Because loopback interfaces do not rely on any hardware, these interfaces can be up/up whenever IOS is running, making them good interfaces on which to base an OSPF RID.

Example 8-8 shows the configuration that existed in Routers R1 and R2 before the creation of the **show** command output in Examples 8-4, 8-5, and 8-6. R1 set its router ID using the direct method, while R2 used a loopback IP address.

### Example 8-8 OSPF Router ID Configuration Examples

```
! R1 Configuration first
router ospf 1
  router-id 1.1.1.1
  network 10.1.0.0 0.0.255.255 area 0
! R2 Configuration next
!
interface Loopback2
  ip address 2.2.2.2 255.255.255.255
```

Each router chooses its OSPF RID when OSPF is initialized, which happens when the router boots or when a CLI user stops and restarts the OSPF process (with the **clear ip ospf process** command). So, if OSPF comes up, and later the configuration changes in a way that would impact the OSPF RID, OSPF does not change the RID immediately. Instead, IOS waits until the next time the OSPF process is restarted.

Example 8-9 shows the output of the **show ip ospf** command on R1, after the configuration of Example 8-8 was made, and after the router was reloaded, which made the OSPF router ID change.

### Example 8-9 Confirming the Current OSPF Router ID

```
R1# show ip ospf
  Routing Process "ospf 1" with ID 1.1.1.1
  ! lines omitted for brevity
```

## OSPF Passive Interfaces

Once OSPF has been enabled on an interface, the router tries to discover neighboring OSPF routers and form a neighbor relationship. To do so, the router sends OSPF Hello messages on a regular time interval (called the Hello Interval). The router also listens for incoming Hello messages from potential neighbors.

Sometimes, a router does not need to form neighbor relationships with neighbors on an interface. Often, no other routers exist on a particular link, so the router has no need to keep sending those repetitive OSPF Hello messages.

When a router does not need to discover neighbors off some interface, the engineer has a couple of configuration options. First, by doing nothing, the router keeps sending the messages, wasting some small bit of CPU cycles and effort. Alternately, the engineer can configure the interface as an OSPF passive interface, telling the router to do the following:



**Fast Fact**

- Quit sending OSPF Hellos on the interface.
- Ignore received Hellos on the interface.
- Do not form neighbor relationships over the interface.

By making an interface passive, OSPF does not form neighbor relationships over the interface, but it does still advertise about the subnet connected to that interface. That is, the OSPF configuration enables OSPF on the interface (using the **network** router subcommand), and then makes the interface passive (using the **passive-interface** router subcommand).

To configure an interface as passive, two options exist. First, you can add the following command to the configuration of the OSPF process, in router configuration mode:

- **passive-interface** *type number*

Alternately, the configuration can change the default setting so that all interfaces are passive by default, and then add a **no passive-interface** command for all interfaces that need to not be passive:

- **passive-interface default**
- **no passive interface** *type number*

For example, in the sample internetwork in [Figure 8-2](#) (used in the single-area configuration examples), Router R1, at the bottom left of the figure, has a LAN interface configured for VLAN trunking. The only router connected to both VLANs is Router R1, so R1 will never discover an OSPF neighbor on these subnets. Example 8-10 shows two alternative configurations to make the two LAN subinterfaces passive to OSPF.

**Example 8-10 Configuring Passive Interfaces on R1 and R2 from [Figure 8-2](#)**

```
! First, make each subinterface passive directly
router ospf 1
  passive-interface GigabitEthernet0/0.11
  passive-interface GigabitEthernet0/0.12

! Or, change the default to passive, and make the other interfaces
! not be passive

router ospf 1
  passive-interface default
  no passive-interface serial0/0/0
  no passive-interface serial0/0/1
```

In real internetworks, the choice of configuration style reduces to which option requires the least number of commands. For example, a router with 20 interfaces, 18 of which are passive to OSPF, has far fewer configuration commands when using the **passive-interface default** command to change the default to passive. If only two of those 20 interfaces need to be passive, use the default setting, in which all interfaces are not passive, to keep the configuration shorter.

Interestingly, OSPF makes it a bit of a challenge to use **show** commands to find whether or not an interface is passive. The **show running-config** command lists the configuration directly, but if you cannot get into enable mode to use this command, note these two facts:

- The **show ip ospf interface brief** command lists all interfaces on which OSPF is enabled, *including passive interfaces*.
- The **show ip ospf interface** command lists a single line that mentions that the interface is passive.

Example 8-11 shows these two commands on Router R1, with the configuration shown in the top of Example 8-10. Note that subinterfaces G0/0.11 and G0/0.12 both show up in the output of **show ip ospf interface brief**.

### Example 8-11 *Displaying Passive Interfaces*

```
R1# show ip ospf interface brief
```

Interface	PID	Area	IP Address/Mask	Cost	State	Nbrs	F/C
Gi0/0.12	1	0	10.1.1.129/25	1	DR	0/0	
Gi0/0.11	1	0	10.1.1.1/25	1	DR	0/0	
Se0/0/0	1	0	10.1.12.1/25	64	P2P	0/0	
Se0/0/1	1	0	10.1.13.1/25	64	P2P	0/0	

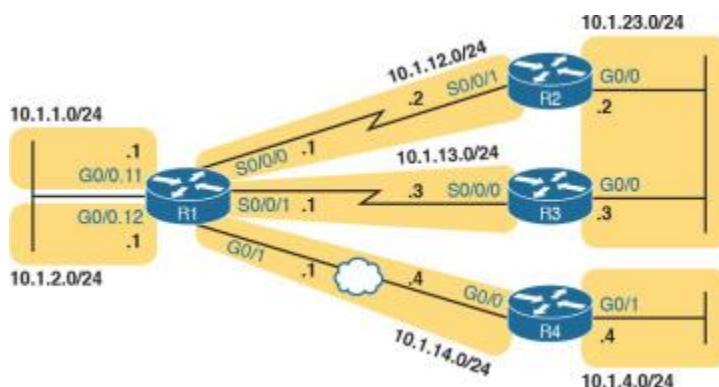
```
R1# show ip ospf interface g0/0.11
```

```
GigabitEthernet0/0.11 is up, line protocol is up
  Internet Address 10.1.1.1/25, Area 0, Attached via Network Statement
  Process ID 1, Router ID 10.1.1.129, Network Type BROADCAST, Cost: 1
  Topology-MTID      Cost      Disabled      Shutdown      Topology Name
    0                  1          no            no            Base
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 10.1.1.129, Interface address 10.1.1.1
  No backup designated router on this network
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    oob-resync timeout 40
  No Hellos (Passive interface)
! Lines omitted for brevity
```

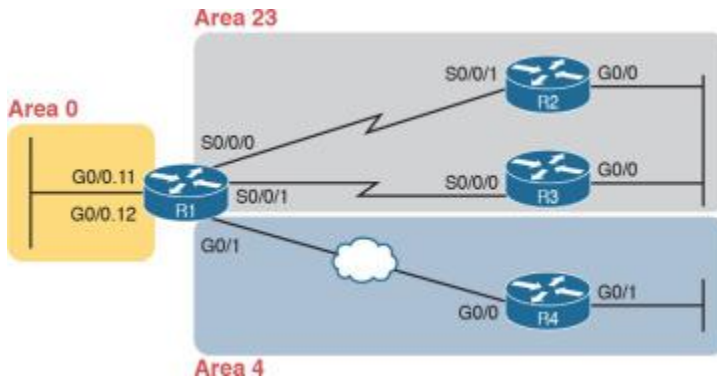
## 3. Implementing Multiarea OSPFv2 to the network

Configuring the routers in a multiarea design is almost just like configuring OSPFv2 for a single area. The only difference is that the configuration places some interfaces on each ABR in different areas. The differences come in the verification and operation of OSPFv2.

This second major section of the chapter provides a second set of configurations to contrast multiarea configuration with single-area configuration. This new scenario shows the configuration for the routers in the multiarea OSPF design based on [Figures 8-2](#) and [8-3](#). [Figure 8-2](#) shows the internetwork topology and subnet IDs, and [Figure 8-3](#) shows the area design. Note that [Figure 8-2](#) lists the last octet of each router's IPv4 address near each interface, rather than the entire IPv4 address, to reduce clutter.



**Figure 8-2** Subnets for a Multiarea OSPF Configuration Example



**Figure 8-3** Area Design for an Example Multiarea OSPF Configuration

Take a moment to think about the area design shown in [Figure 8-3](#), and look for the ABRs. Only R1 connects to the backbone area at all. The other three routers are internal routers in a single area. So, as it turns out, three of the four routers have single-area configurations, with all interfaces in the same area.

Note that the examples in this section use a variety of configuration options just so you can see those options. The options include different ways to set the OSPF RID, different wildcard masks on OSPF **network** commands, and the use of passive interfaces where no other OSPF routers should exist off an interface.

### Single-Area Configurations

Example 8-12 begins the configuration example by showing the OSPF and IP address configuration on R2. Note that R2 acts as an internal router in area 23, meaning that the configuration will refer to only one area (23). The configuration sets R2's RID to 2.2.2.2 directly with the **router-id** command. And, because R2 should find neighbors on both its two interfaces, neither can reasonably be made passive, so R2's configuration lists no passive interfaces.

#### Example 8-12 OSPF Configuration on R2, Placing Two Interfaces into Area 23

```
interface GigabitEthernet0/0
 ip address 10.1.23.2 255.255.255.0
!
interface serial 0/0/1
 ip address 10.1.12.2 255.255.255.0
!
router ospf 1
 network 10.0.0.0 0.255.255.255 area 23
 router-id 2.2.2.2
```

Example 8-13 continues reviewing a few commands with the configuration for both R3 and R4. R3 puts both its interfaces into area 23, per its **network** command, sets its RID to 3.3.3.3 by using a loopback interface, and, like R2, cannot make either of its interfaces passive. The R4 configuration is somewhat different, with both interfaces placed into area 4, setting its RID based on a nonloopback interface (G0/0, for OSPF RID 10.1.14.4),

and making R4's G0/1 interface passive, because no other OSPF routers sit on that link. (Note that the choice to use one method over another to set the OSPF RID is simply to show the variety of configuration options.)

### Example 8-13 OSPF Single-Area Configuration on R3 and R4

```
! First, on R3
interface GigabitEthernet0/0
 ip address 10.1.23.3 255.255.255.0
!
interface serial 0/0/0
 ip address 10.1.13.3 255.255.255.0
!
interface loopback 0
 ip address 3.3.3.3 255.255.255.0
!
router ospf 1
 network 10.0.0.0 0.255.255.255 area 23
! Next, on R4
interface GigabitEthernet0/0
 description R4 will use this interface for its OSPF RID
 ip address 10.1.14.4 255.255.255.0
!
interface GigabitEthernet0/1
 ip address 10.1.4.4 255.255.255.0
!
router ospf 1
 network 10.0.0.0 0.255.255.255 area 4
 passive-interface GigabitEthernet0/1
```

### Multiarea Configuration

The only router that has a multiarea config is an ABR, by virtue of the configuration referring to more than one area. In this design (as shown in [Figure 8-4](#)), only Router R1 acts as an ABR, with interfaces in three different areas. Example 8-14 shows R1's OSPF configuration. Note that the configuration does not state anything about R1 being an ABR; instead, it uses multiple **network** commands, some placing interfaces into area 0, some into area 23, and some into area 4.

### Example 8-14 OSPF Multiarea Configuration on Router R1

```
interface GigabitEthernet0/0.11
 encapsulation dot1q 11
 ip address 10.1.1.1 255.255.255.0
!
interface GigabitEthernet0/0.12
 encapsulation dot1q 12
 ip address 10.1.2.1 255.255.255.0
!
interface GigabitEthernet0/1
 ip address 10.1.14.1 255.255.255.0
!
```



```
interface serial 0/0/0
 ip address 10.1.12.1 255.255.255.0
!
interface serial 0/0/1
 ip address 10.1.13.1 255.255.255.0
!
router ospf 1
 network 10.1.1.1 0.0.0.0 area 0
 network 10.1.2.1 0.0.0.0 area 0
 network 10.1.12.1 0.0.0.0 area 23
 network 10.1.13.1 0.0.0.0 area 23
 network 10.1.14.1 0.0.0.0 area 4
 router-id 1.1.1.1
 passive-interface GigabitEthernet0/0.11
 passive-interface GigabitEthernet0/0.12
```

Focus on the highlighted **network** commands in the example. All five commands happen to use a wildcard mask of 0.0.0.0, so that each command requires a specific match of the listed IP address. If you compare these **network** commands to the various interfaces on Router R1, you can see that the configuration enables OSPF, for area 0, on subinterfaces G0/0.11 and G0/0.12, area 23 for the two serial interfaces, and area 4 for R1's G0/1 interface.

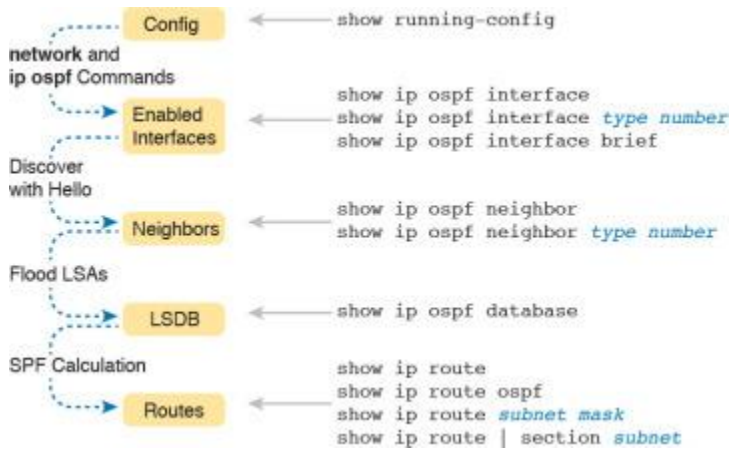
Many networks make a habit of using a 0.0.0.0 wildcard mask on OSPF **network** commands, requiring an exact match of each interface IP address, as shown in Example 8-14. This style of configuration makes it more obvious exactly which interfaces match which **network** command.

Finally, note that R1's configuration also sets its RID directly and makes its two LAN subinterfaces passive.

So, what's the big difference between single-area and multiarea OSPF configuration? Practically nothing. The only difference is that with multiarea, the ABR's **network** commands list different areas.

### Verifying the Multiarea Configuration

The next few pages look at how to verify a few of the new OSPF features introduced in this chapter. [Figure 8-4](#) summarizes the most important OSPF verification commands for reference.



**Figure 8-4** OSPF Verification Commands

\This section looks at the following topics:

- Verifying the ABR interfaces are in the correct (multiple) areas
- Finding which router is DR and BDR on multiaccess links
- A brief look at the LSDB
- Displaying IPv4 routes

#### Verifying the Correct Areas on Each Interface on an ABR

The easiest place to make a configuration oversight with a multiarea configuration is to place an interface into the wrong OSPF area. Several commands mention the OSPF area. The **show ip protocols** command basically relists the OSPF **network** configuration commands, which indirectly identify the interfaces and areas. Also, the **show ip ospf interface** and **show ip ospf interface brief** commands directly show the area configured for an interface; Example 8-15 shows an example of the briefer version of these commands.

#### Example 8-15 Listing the OSPF-Enabled Interfaces and the Matching OSPF Areas

```

R1# show ip ospf interface brief
Interface    PID    Area      IP Address/Mask    Cost    State  Nbrs  F/C
Gi0/0.12     1      0         10.1.2.1/24        1       DR     0/0
Gi0/0.11     1      0         10.1.1.1/24        1       DR     0/0
Gi0/1        1      4         10.1.14.1/24       1       BDR    1/1
Se0/0/1      1      23        10.1.13.1/24       64      P2P    1/1
Se0/0/0      1      23        10.1.12.1/24       64      P2P    1/1

```

In the output, to correlate the areas, just look at the interface in the first column and the area in the third column. Also, for this example, double-check this information with [Figures 8-3](#) and [8-4](#) to confirm that the configuration matches the design.

### Verifying Which Router Is DR and BDR

Several **show** commands identify the DR and BDR in some way, as well. In fact, the **show ip ospf interface brief** command output, just listed in Example 8-15, lists the local router's state, showing that R1 is DR on two subinterfaces and BDR on its G0/1 interface.

Example 8-16 shows two other examples that identify the DR and BDR, but with a twist. The **show ip ospf interface** command lists detailed output about OSPF settings, per interface. Those details include the RID and interface address of the DR and BDR. At the same time, the **show ip ospf neighbor** command lists shorthand information about the neighbor's DR or BDR role as well; this command does not say anything about the local router's role.

### Example 8-16 Discovering the DR and BDR on the R1–R4 Ethernet (from R4)

```
R4# show ip ospf interface gigabitEthernet 0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet Address 10.1.14.4/24, Area 4, Attached via Network Statement
  Process ID 1, Router ID 10.1.14.4, Network Type BROADCAST, Cost: 1
  Topology-MTID      Cost      Disabled      Shutdown      Topology Name
    0                  1          no            no            Base
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 10.1.14.4, Interface address 10.1.14.4
  Backup Designated router (ID) 1.1.1.1, Interface address 10.1.14.1
  !
  ! Lines omitted for brevity
R4# show ip ospf neighbor
```

Neighbor ID	Pri	State	Dead Time	Address	Interface
1.1.1.1	1	FULL/BDR	00:00:33	10.1.14.1	GigabitEthernet0/0

First, focus on the highlighted lines from the **show ip ospf interface** command output. It lists the DR as RID 10.1.14.4, which is R4. It also lists the BDR as 1.1.1.1, which is R1.

The end of the example shows the **show ip ospf neighbor** command on R4, listing R4's single neighbor, with Neighbor RID 1.1.1.1 (R1). The command lists R4's concept of its neighbor state with neighbor 1.1.1.1 (R1), with the current state listed as FULL/BDR. The FULL state means that R4 has fully exchanged its LSDB with R1. BDR means that the neighbor (R1) is acting as the BDR, implying that R4 (the only other router on this link) is acting as the DR.

Example 8-16 also shows the results of an DR/BDR election, with the router using the higher RID winning the election. The rules work like this:

- When a link comes up, if two (or more) routers on the subnet send and hear each other's Hello messages, they elect a DR and BDR, with the higher OSPF RID becoming DR, and the second highest RID becoming the BDR.

- Once the election has completed, new routers entering the subnet do not take over the DR or BDR role, even if they have better (higher) RID.

In this case, Routers R1 and R4, on the same Ethernet, heard each other's Hellos. R1, with RID 1.1.1.1, has a lower-value RID than R4's 10.1.14.1. As a result, R4 (10.1.14.1) won the DR election.

### Verifying Interarea OSPF Routes

Finally, all this OSPF theory and all the **show** commands do not matter if the routers do not learn IPv4 routes. To verify the routes, Example 8-17 shows R4's IPv4 routing table.

### Example 8-17 Verifying OSPF Routes on Router R4

```
R4# show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

    10.0.0.0/8 is variably subnetted, 9 subnets, 2 masks
O IA   10.1.1.0/24 [110/2] via 10.1.14.1, 11:04:43, GigabitEthernet0/0
O IA   10.1.2.0/24 [110/2] via 10.1.14.1, 11:04:43, GigabitEthernet0/0
C       10.1.4.0/24 is directly connected, GigabitEthernet0/1
L       10.1.4.4/32 is directly connected, GigabitEthernet0/1
O IA   10.1.12.0/24 [110/65] via 10.1.14.1, 11:04:43, GigabitEthernet0/0
O IA   10.1.13.0/24 [110/65] via 10.1.14.1, 11:04:43, GigabitEthernet0/0
C       10.1.14.0/24 is directly connected, GigabitEthernet0/0
L       10.1.14.4/32 is directly connected, GigabitEthernet0/0
O IA   10.1.23.0/24 [110/66] via 10.1.14.1, 11:04:43, GigabitEthernet0/0
```

This example shows a couple of new codes that are particularly interesting for OSPF. As usual, a single character on the left identifies the source of the route, with O meaning OSPF. In addition, IOS notes any interarea routes with an IA code as well. (The example does not list any intra-area OSPF routes, but these routes would simply omit the IA code; earlier Example 8-6 lists some intra-area OSPF routes.) Also, note that R4 has routes to all seven subnets in the topology used in this example: two connected routes and five interarea OSPF routes.

## 4. Additional OSPF Features

So far this chapter has focused on the most common OSPF features using the traditional configuration using the OSPF **network** command. This final of three major sections discusses some very popular but optional OSPFv2 configuration features, as listed here in their order of appearance:

- Default routes
- Metrics
- Load balancing
- OSPF interface configuration

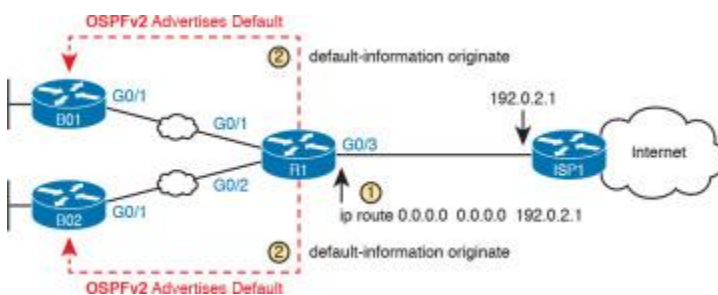
### OSPF Default Routes

In some cases, routers benefit from using a default route. For those exact same reasons, networks that happen to use OSPFv2 can use OSPF to advertise default routes.

The most classic case for using a routing protocol to advertise a default route has to do with an enterprise's connection to the Internet. As a strategy, the enterprise engineer uses these design goals:

- All routers learn specific routes for subnets inside the company; a default route is not needed when forwarding packets to these destinations.
- One router connects to the Internet, and it has a default route that points toward the Internet.
- All routers should dynamically learn a default route, used for all traffic going to the Internet, so that all packets destined to locations in the Internet go to the one router connected to the Internet.

[Figure 8-5](#) shows the idea of how OSPF advertises the default route, with the specific OSPF configuration. In this case, a company connects to an ISP with its Router R1. That router has a static default route (destination 0.0.0.0, mask 0.0.0.0) with a next-hop address of the ISP router. Then, the use of the OSPF **default-information originate** command (Step 2) makes the router advertise a default route using OSPF to the remote routers (B1 and B2).

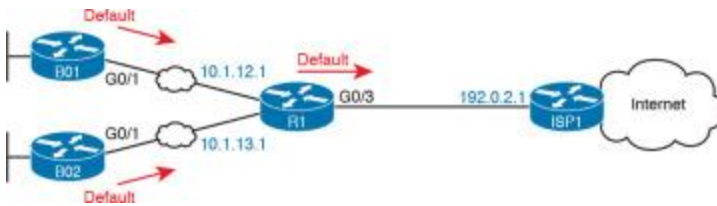


**Figure 8-5** Using OSPF to Create and Flood a Default Route

NOTE

The example in [Figure 8-5](#) uses a static default route, but it could have used a default route as learned from the ISP with DHCP, as well as learning a default route with External BGP (eBGP), as discussed in Chapter 12, “Implementing External BGP.”

[Figure 8-6](#) shows the default routes that result from OSPF’s advertisements in [Figure 8-5](#). On the far left, the branch routers all have OSPF-learned default routes, pointing to R1. R1 itself also needs a default route, pointing to the ISP router, so that R1 can forward all Internet-bound traffic to the ISP.



**Figure 8-6** *Default Routes Resulting from the **default-information originate** Command*

Finally, this feature gives the engineer control over when the router originates this default route. First, R1 needs a default route, either defined as a static default route, learned from the ISP with DHCP, or learned from the ISP with a routing protocol like eBGP. The **default-information originate** command then tells R1 to advertise a default route when its own default route is working, and to advertise the default route as down when its own default route fails.



Interestingly, the **default-information originate always** router subcommand tells the router to always advertise the default route, no matter whether the router’s default route is working or not.

Example 8-18 shows details of the default route on both R1 and branch router B01. Beginning with Router R1, in this case, Router R1 used DHCP to learn its IP address on its G0/3 interface from the ISP. R1 then creates a static default route with the ISP router’s IP address of 192.0.2.1 as the next-hop address, as highlighted in the output of the **show ip route static** command output.

### **Example 8-18** *Default Routes on Routers R1 and B01*

```
! The next command is from Router R1. Note the static code for the default route
R1# show ip route static
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
! Rest of the legend omitted for brevity
```

```
Gateway of last resort is 192.0.2.1 to network 0.0.0.0
```

```
S*    0.0.0.0/0 [254/0] via 192.0.2.1
```

```
! The next command is from router B01; notice the External route code for the default
```

```
B01# show ip route ospf
```

```
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
```



D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area  
 N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2  
 E1 - OSPF external type 1, E2 - OSPF external type 2  
 br/>#! Rest of the legend omitted for brevity

Gateway of last resort is 10.1.12.1 to network 0.0.0.0

```
O*E2  0.0.0.0/0 [110/1] via 10.1.12.1, 00:20:51, GigabitEthernet0/1
      10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
O      10.1.3.0/24 [110/3] via 10.1.12.1, 00:20:51, GigabitEthernet0/1
O      10.1.13.0/24 [110/2] via 10.1.12.1, 00:20:51, GigabitEthernet0/1
```

Keeping the focus on the command on Router R1, note that R1 indeed has a default route, that is, a route to 0.0.0.0/0. The “Gateway of last resort,” which refers to the default route currently used by the router, points to next-hop IP address 192.0.2.1, which is the ISP router’s IP address.

Next look to the bottom half of the example, and router BO1’s OSPF-learned default route. BO1 lists a route for 0.0.0.0/0 as well. The next-hop router in this case is 10.1.12.1, which is Router R1’s IP address on the WAN link. The code on the far left is O\*E2, meaning: an OSPF-learned route, which is a default route, and is specifically an external OSPF route. Finally, BO1’s gateway of last resort setting uses that one OSPF-learned default route, with next-hop router 10.1.12.1.

## Steady-State Operation

If Hello Interval is not received for [dead interval] amount of time, the router believes the neighbor has failed.

- Default dead timer is 4 times the hello interval

(10 second hello, 40 second dead timer)

- Router marks as "down" in its neighbor table
- Runs the dijkstra algorithm to calculate new routes, floods to inform other routers of failed link

## OSPF Metrics (Cost)

Earlier, the Chapter 7 section “Calculating the Best Routes with SPF” discussed how SPF calculates the metric for each route, choosing the route with the best metric for each destination subnet. OSPF routers can influence that choice by changing the OSPF interface cost on any and all interfaces.

Cisco routers allow two different ways to change the OSPF interface cost. The one straightforward way is to set the cost directly, with an interface subcommand: **ip ospf cost x**. The other method is to let IOS choose default costs, based on a formula, but to change the inputs to the formula. This second method requires a little more thought and care and is the focus of this next topic.

## Setting the Cost Based on Interface Bandwidth

The default OSPF cost values can actually cause a little confusion, for a couple of reasons. So, to get through some of the potential confusion, this section begins with some examples.

First, IOS uses the following formula to choose an interface's OSPF cost. IOS puts the interface's bandwidth in the denominator, and a settable OSPF value called the *reference bandwidth* in the numerator:

- $\text{Reference\_bandwidth} / \text{Interface\_bandwidth}$

With this formula, the following sequence of logic happens:

1. A higher interface bandwidth—that is, a faster bandwidth—results in a lower number in the calculation.
2. A lower number in the calculation gives the interface a lower cost.
3. An interface with a lower cost is more likely to be used by OSPF when calculating the best routes.

Now for some examples. Assume a default reference bandwidth, set to 100 Mbps, which is the same as 100,000 Kbps. (The upcoming examples will use a unit of Kbps just to avoid math with fractions.) Assume defaults for interface bandwidth on serial, Ethernet, and Fast Ethernet interfaces, as shown in the output of the **show interfaces** command, respectively, of 1544 Kbps, 10,000 Kbps (meaning 10 Mbps), and 100,000 Kbps (meaning 100 Mbps). Table 8-2 shows the results of how IOS calculates the OSPF cost for some interface examples.

**Table 8-2 OSPF Cost Calculation Examples with Default Bandwidth Settings**

Interface	Interface Default Bandwidth (Kbps)	Formula (Kbps)	OSPF Cost
Serial	1544 Kbps	$100,000/1544$	64
Ethernet	10,000 Kbps	$100,000/10,000$	10
Fast Ethernet	100,000 Kbps	$100,000/100,000$	1

Example 8-19 shows the cost settings on R1's OSPF interfaces, all based on default OSPF (reference bandwidth) and default interface bandwidth settings.

### Example 8-19 Confirming OSPF Interface Costs

```
R1# show ip ospf interface brief
Interface    PID    Area    IP Address/Mask    Cost    State    Nbrs    F/C
Gi0/0.12     1      0       10.1.2.1/24        1       DR       0/0
Gi0/0.11     1      0       10.1.1.1/24        1       DR       0/0
Gi0/1        1      4       10.1.14.1/24       1       BDR      1/1
Se0/0/1      1      23      10.1.13.1/24       64      P2P      1/1
Se0/0/0      1      23      10.1.12.1/24       64      P2P      1/1
```

To change the OSPF cost on these interfaces, the engineer simply needs to use the **bandwidth speed** interface subcommand to set the bandwidth on an interface. The interface bandwidth does not change the Layer 1 transmission speed at all; instead, it is used for other purposes, including routing protocol metric calculations.

For instance, if you add the **bandwidth 10000** command to a serial interface, with a default reference bandwidth, the serial interface's OSPF cost could be calculated as  $100,000 / 10,000 = 10$ .

Note that if the calculation of the default metric results in a fraction, OSPF rounds down to the nearest integer. For instance, the example shows the cost for interface S0/0/0 as 64. The calculation used the default serial interface bandwidth of 1.544 Mbps, with reference bandwidth 100 (Mbps), with the  $100 / 1.544$  calculation resulting in 64.7668394. OSPF rounds down to 64.

### The Need for a Higher Reference Bandwidth

This default calculation works nicely as long as the fastest link in the network runs at 100 Mbps. The default reference bandwidth is set to 100, meaning 100 Mbps, the equivalent of 100,000 Kbps. As a result, with default settings, faster router interfaces end up with the same OSPF cost, as shown in Table 8-3, because the lowest allowed OSPF cost is 1.

**Table 8-3 Faster Interfaces with Equal OSPF Costs**

Interface	Interface Default Bandwidth (Kbps)	Formula (Kbps)	OSPF Cost
Fast Ethernet	100,000 Kbps	$100,000/100,000$	1
Gigabit Ethernet	1,000,000 Kbps	$100,000/1,000,000$	1
10 Gigabit Ethernet	10,000,000 Kbps	$100,000/10,000,000$	1
100 Gigabit Ethernet	100,000,000 Kbps	$100,000/100,000,000$	1

To avoid this issue, and change the default cost calculation, you can change the reference bandwidth with the **auto-cost reference-bandwidth speed** OSPF mode subcommand. This command sets a value in a unit of megabits per second (Mbps). To avoid the issue shown in Table 8-3, set the reference bandwidth value to match the fastest link speed in the network. For instance, **auto-cost reference-bandwidth 10000** accommodates links up to 10 Gbps in speed.

#### NOTE

Cisco recommends making the OSPF reference bandwidth setting the same on all OSPF routers in an enterprise network.

For convenient study, the following list summarizes the rules for how a router sets its OSPF interface costs:



1. Set the cost explicitly, using the **ip ospf cost** *x* interface subcommand, to a value between 1 and 65,535, inclusive.
2. Change the interface bandwidth with the **bandwidth** *speed* command, with *speed* being a number in kilobits per second (Kbps).
3. Change the reference bandwidth, using router OSPF subcommand **auto-cost reference-bandwidth** *ref-bw*, with a unit of megabits per second (Mbps).

## OSPF Load Balancing

When a router uses SPF to calculate the metric for each of several routes to reach one subnet, one route may have the lowest metric, so OSPF puts that route in the routing table. However, when the metrics tie for multiple routes to the same subnet, the router can put multiple equal-cost routes in the routing table (the default is four different routes) based on the setting of the **maximum-paths** *number* router subcommand. For example, if an internetwork has six possible paths between some parts of the network, and the engineer wants all routes to be used, the routers can be configured with the **maximum-paths 6** subcommand under **router ospf**.

The more challenging concept relates to how the routers use those multiple routes. A router could load balance the packets on a per-packet basis. For example, if the router has three equal-cost OSPF routes for the same subnet in the routing table, the router could send the one packet over the first route, the next packet over the second route, the next packet over the third route, and then start over with the first route for the next packet. Alternatively, the load balancing could be on a per-destination IP address basis.

Note that the default setting of **maximum-paths** varies by router platform.

## OSPFv2 Interface Configuration

The newer interface-style OSPF configuration works mostly like the old style, for almost all features, with one important exception. The interface configuration enables OSPF directly on the interface with the **ip ospf** interface subcommand, while the traditional OSPFv2 configuration enables OSPFv2 on an interface, but indirectly, using the **network** command in OSPF configuration mode. The rest of the OSPF features discussed throughout this chapter are not changed by the use of OSPFv2 interface configuration.

Basically, instead of matching interfaces with indirect logic using **network** commands, you directly enable OSPFv2 on interfaces by configuring an interface subcommand on each interface.

### OSPFv2 Interface Configuration Example

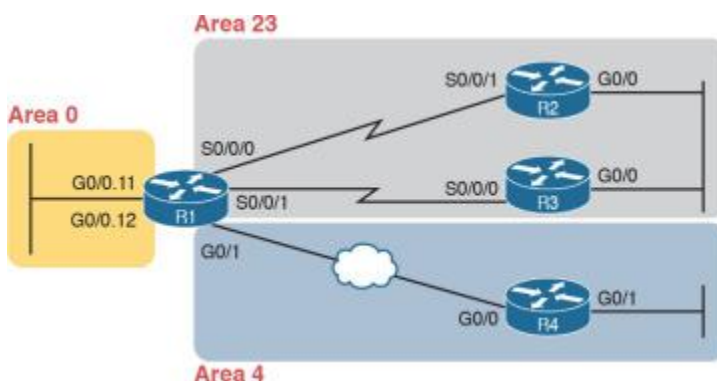
To show how OSPF interface configuration works, this example basically repeats the example shown earlier in the book using the traditional OSPFv2 configuration with **network** commands. So, before looking at the OSPFv2 interface configuration, take a moment to look back at [Figures 8-3](#) and [8-4](#), along with Examples 8-12, 8-13, and 8-14. Once reviewed, for easier reference, [Figure 8-8](#) repeats [Figure 8-4](#) for reference in the upcoming interface configuration examples.

To convert from the old-style configuration in Examples 8-12, 8-13, and 8-14, simply do the following:



- **Step 1.** Use the **no network network-id area area-id** subcommands in OSPF configuration mode to remove the **network** commands.
- **Step 2.** Add one **ip ospf process-id area area-id** command in interface configuration mode under each interface on which OSPF should operate, with the correct OSPF process (*process-id*) and the correct OSPF area number.

For example, Example 8-12 had a single **network** command that enabled OSPF on two interfaces on Router R2, putting both in area 23. Example 8-20 shows the replacement newer style of configuration.



**Figure 8-7** Area Design Used in the Upcoming OSPF Example

### Example 8-20 New-Style Configuration on Router R2

```
interface GigabitEthernet0/0
 ip address 10.1.23.2 255.255.255.0
 ip ospf 1 area 23
!
interface serial 0/0/1
 ip address 10.1.12.2 255.255.255.0
 ip ospf 1 area 23

router ospf 1
 router-id 2.2.2.2
! Notice - no network commands here!
```

### Verifying OSPFv2 Interface Configuration

OSPF operates the same way whether you use the new style or old style of configuration. The OSPF area design works the same, neighbor relationships form the same way, routers negotiate to become the DR and BDR the same way, and so on. However, you can see a few small differences in command output when using the newer OSPFv2 configuration if you look closely.

The **show ip protocols** command relists most of the routing protocol configuration, just in slightly different format, as shown in Example 8-21. With the newer-style configuration, the output lists the phrase “Interfaces Configured Explicitly,” with the list of interfaces configured with the new **ip ospf process-id area area-id** commands, as highlighted in the example. With the old configuration, the output lists the contents of all the **network** commands, just leaving out the “network” word itself. Note that in the next two examples, R2 has been reconfigured to use OSPF interface configuration as shown in the previous example (Example 8-20), while Router R3 still uses the older-style **network** commands per earlier configuration Example 8-13.

### Example 8-21 Differences in show ip protocols Output: Old- and New-Style OSPFv2 Configuration

```
R2# show ip protocols
*** IP Routing is NSF aware ***

Routing Protocol is "ospf 1"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Router ID 2.2.2.2
  Number of areas in this router is 1. 1 normal 0 stub 0 nssa
  Maximum path: 4
  Routing for Networks:
  Routing on Interfaces Configured Explicitly (Area 23):
    Serial0/0/1
    GigabitEthernet0/0
  Routing Information Sources:
    Gateway         Distance      Last Update
    3.3.3.3          110          00:04:59
    1.1.1.1          110          00:04:43
  Distance: (default is 110)
```



```
! Below, showing only the part that differs on R3:
R3# show ip protocols
! ... beginning lines omitted for brevity
  Routing for Networks:
    10.0.0.0 0.255.255.255 area 23
! ... ending line omitted for brevity
```

Basically, the **show ip protocols** command output differs depending on the style of configuration, either relisting the interfaces when using interface configuration or relisting the network commands if using **network** commands.

Next, the **show ip ospf interface [interface]** command lists details about OSPF settings for the interface(s) on which OSPF is enabled. The output also makes a subtle reference to whether that interface was enabled for OSPF with the old or new configuration style. As seen in Example 8-22, R2’s new-style interface configuration results in the highlighted text, “Attached via Interface Enable,” whereas R3’s old-style configuration lists “Attached via Network Statement.”

### Example 8-22 Differences in show ip ospf interface Output with OSPFv2 Interface Configuration

```
R2# show ip ospf interface g0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet Address 10.1.23.2/24, Area 23, Attached via Interface Enable
  Process ID 1, Router ID 22.2.2.2, Network Type BROADCAST, Cost: 1
  Topology-MTID      Cost      Disabled      Shutdown      Topology Name
    0                1          no            no            Base
  Enabled by interface config, including secondary ip addresses
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 2.2.2.2, Interface address 10.1.23.2
  Backup Designated router (ID) 3.3.3.3, Interface address 10.1.23.3
! Showing only the part that differs on R3:
R3# show ip ospf interface g0/0
GigabitEthernet0/0 is up, line protocol is up
  Internet Address 10.1.23.3/24, Area 23, Attached via Network Statement
! ... ending line omitted for brevity
```

Note that the briefer version of this command, the **show ip ospf interface brief** command, does not change whether the configuration uses traditional **network** commands or the alternative interface configuration with the **ip ospf** interface subcommand.

## 5. OSPF Implementation Summary

- An OSPF **network** command is used to match the IP addresses that are configured on the interfaces. Those that match are inserted into the OSPF process.
- An OSPF **network** command uses wildcard masks to control which bits in an octet are matched.
- The **show ip ospf neighbor** command can be used to find information about any OSPF neighborships, including the interface, the state, the neighbor’s address, and the neighbor’s router ID.

- To select a router ID for OSPF, a router goes through a process. When a router ID has been found, the process stops. The process is any value configured with the **router-id** command; the highest configured IPv4 address of any enabled loopback interface; and the highest configured IPv4 address of any physically up (up/up or up/down) physical interface.
- An OSPF interface configured as passive will quit sending OSPF Hello messages, will ignore any received Hello messages, and will not form any neighborships.
- The only OSPF router configured into multiple areas is an Area Border Router (ABR).
- The **show ip ospf interface** [*type number* | **brief**] command can be used to display which interfaces are enabled into the OSPF process.
- The **show ip ospf neighbor** [*type number*] command can be used to display any OSPF neighborships.
- The **show ip ospf database** command can be used to display the OSPF LSDB.
- The **show ip route** [**ospf** | *subnet mask*] command can be used to display OSPF routes in the current routing table.
- The **show ip protocols** and **show ip ospf interface** [**brief**] commands can be used to display which areas are configured on a device.
- An OSPF **default-information originate** command is used along with a configured static default route to advertise a default route into OSPF.
- OSPF uses three rules to set interface costs: setting the cost explicitly with the **ip ospf cost** *cost* command, changing the interface bandwidth with the **bandwidth** *bandwidth* command, or changing the reference bandwidth with the **auto-cost reference-bandwidth** *reference-bandwidth* command.
- The output of the **show ip protocols** and **show ip ospf interface** commands will differ depending on whether OSPF was configured with the old (**network**) or new (interface commands) configuration style.

## Reference:

- <http://www.ciscopress.com/articles/article.asp?p=2738311>