



Implementing OSPF

Open Shortest Path First (OSPF) is an Interior Gateway Protocol (IGP) developed by the OSPF working group of the Internet Engineering Task Force (IETF). Designed expressly for IP networks, OSPF supports IP subnetting and tagging of externally derived routing information. OSPF also allows packet authentication when sending and receiving packets.

OSPF Version 3 (OSPFv3) expands on OSPF Version 2, providing support for IPv6 routing prefixes.

This module describes the concepts and tasks you need to implement both versions of OSPF on your software. The term “OSPF” implies both versions of the routing protocol, unless otherwise noted.



Note GTSM TTL Security is not supported.

- [Prerequisites for Implementing OSPF](#) , on page 2
- [Enable OSPF](#) , on page 2
- [Verify OSPF Configuration and Operation](#) , on page 5
- [Stub Area](#) , on page 7
- [Neighbors and Adjacency for OSPF](#) , on page 10
- [Authentication Strategies](#) , on page 14
- [Control Frequency That Same LSA Is Originated or Accepted for OSPF](#) , on page 18
- [Virtual Link and Transit Area for OSPF](#) , on page 20
- [Summarize Subnetwork LSAs on OSPF ABR](#) , on page 25
- [Route Redistribution for OSPF](#) , on page 28
- [OSPF Shortest Path First Throttling](#) , on page 31
- [Graceful Restart for OSPFv3](#) , on page 34
- [OSPFv2OSPF SPF Prefix Prioritization](#) , on page 37
- [Multi-Area Adjacency for OSPF Version 2](#) , on page 41
- [Label Distribution Protocol IGP Auto-configuration for OSPF](#) , on page 44
- [IGP link state](#) , on page 48
- [OSPF Authentication Message Digest Management](#) , on page 48
- [OSPF Link-State Database Overload Protection](#) , on page 52
- [References for OSPF](#) , on page 53

Prerequisites for Implementing OSPF

The following are prerequisites for implementing OSPF:

- Configuration tasks for OSPFv3 assume that you are familiar with IPv6 addressing and basic configuration. See the *Implementing Network Stack IPv4 and IPv6* in the *Cisco IP Addresses and Services Configuration Guide IP Addresses and Services Configuration Guide for Cisco NCS 5000 Series Routers* for information on IPv6 routing and addressing.
- Before you enable OSPFv3 on an interface, you must perform the following tasks:
 - Complete the OSPF network strategy and planning for your IPv6 network. For example, you must decide whether multiple areas are required.
 - Enable IPv6 on the interface.
- Configuring authentication (IP Security) is an optional task. If you choose to configure authentication, you must first decide whether to configure plain text or Message Digest 5 (MD5) authentication, and whether the authentication applies to an entire area or specific interfaces.

Enable OSPF

This task explains how to perform the minimum OSPF configuration on your router that is to enable an OSPF process with a router ID, configure a backbone or nonbackbone area, and then assign one or more interfaces on which OSPF runs.

Before you begin

Although you can configure OSPF before you configure an IP address, no OSPF routing occurs until at least one IP address is configured.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. **interface** *type interface-path-id*
6. Repeat Step 5 for each interface that uses OSPF.
7. **log adjacency changes** [**detail**] [**enable** | **disable**]
8. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/RP0/CPU0:router(config)# router ospf 1
```

or

```
RP/0/RP0/CPU0:router(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IP address as the router ID.

Step 4 **area** *area-id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters area configuration mode and configures an area for the OSPF process.

- Backbone areas have an area ID of 0.
- Nonbackbone areas have a nonzero area ID.
- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 5 **interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface TenGigE 0/1/0/3
```

Enters interface configuration mode and associates one or more interfaces for the area configured in Step 4.

Step 6 Repeat Step 5 for each interface that uses OSPF.

—

Step 7 **log adjacency changes** [**detail**] [**enable** | **disable**]

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# log adjacency changes detail
```

(Optional) Requests notification of neighbor changes.

- By default, this feature is enabled.
- The messages generated by neighbor changes are considered notifications, which are categorized as severity Level 5 in the **logging console** command. The **logging console** command controls which severity level of messages are sent to the console. By default, all severity level messages are sent.

Step 8 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Enable OSPF: Example

OSPF areas must be explicitly configured, and interfaces configured under the area configuration mode are explicitly bound to that area. In this example, interface 10.1.2.0/24 is bound to area 0 and interface 10.1.3.0/24 is bound to area 1.

```
interface TenGigE 0/3/0/0
 ip address 10.1.2.1 255.255.255.0
 negotiation auto
!
interface TenGigE 0/3/0/1
 ip address 10.1.3.1 255.255.255.0
 negotiation auto
!
router ospf 1
 router-id 10.2.3.4
 area 0
  interface TenGigE 0/3/0/0
!
 area 1
  interface TenGigE 0/3/0/1
!
!
```

Verify OSPF Configuration and Operation

This task explains how to verify the configuration and operation of OSPF.

SUMMARY STEPS

1. **show** { **ospf** | **ospfv3** } [*process-name*]
2. **show** { **ospf** | **ospfv3** } [*process-name*] **border-routers** [*router-id*]
3. **show** { **ospf** | **ospfv3** } [*process-name*] **database**
4. **show** { **ospf** | **ospfv3** } [*process-name*] [*area-id*] **flood-list interface** *type interface-path-id*
5. **show** { **ospf** | **ospfv3** } [*process-name*] [**vrf** *vrf-name*] [*area-id*] **interface** [*type interface-path-id*]
6. **show** { **ospf** | **ospfv3** } [*process-name*] [*area-id*] **neighbor** [*type interface-path-id*] [*neighbor-id*] [**detail**]
7. **clear** { **ospf** | **ospfv3** } [*process-name*] **process**
8. **clear** { **ospf** | **ospfv3** } [*process-name*] **redistribution**
9. **clear** { **ospf** | **ospfv3** } [*process-name*] **routes**
10. **clear** { **ospf** | **ospfv3** } [*process-name*] **vrf** [*vrf-name* | **all**] { **process** | **redistribution** | **routes** | **statistics** [**interface** *type interface-path-id* | **message-queue** | **neighbor**] }
11. **clear** { **ospf** | **ospfv3** } [*process-name*] **statistics** [**neighbor** [*type interface-path-id*] [*ip-address*]]

DETAILED STEPS

Step 1 **show** { **ospf** | **ospfv3** } [*process-name*]

Example:

```
RP/0/RP0/CPU0:router# show ospf group1
```

(Optional) Displays general information about OSPF routing processes.

Step 2 **show** { **ospf** | **ospfv3** } [*process-name*] **border-routers** [*router-id*]

Example:

```
RP/0/RP0/CPU0:router# show ospf group1 border-routers
```

(Optional) Displays the internal OSPF routing table entries to an ABR and ASBR.

Step 3 **show** { **ospf** | **ospfv3** } [*process-name*] **database**

Example:

```
RP/0/RP0/CPU0:router# show ospf group2 database
```

(Optional) Displays the lists of information related to the OSPF database for a specific router.

- The various forms of this command deliver information about different OSPF LSAs.

Step 4 **show** { **ospf** | **ospfv3** } [*process-name*] [*area-id*] **flood-list interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router# show ospf 100 flood-list interface TenGigE 0/3/0/0
```

(Optional) Displays a list of OSPF LSAs waiting to be flooded over an interface.

Step 5 **show** { **ospf** | **ospfv3** } [*process-name*] [**vrf** *vrf-name*] [*area-id*] **interface** [*type interface-path-id*]

Example:

```
RP/0/RP0/CPU0:router# show ospf 100 interface TenGigE 0/3/0/0
```

(Optional) Displays OSPF interface information.

Step 6 **show** { **ospf** | **ospfv3** } [*process-name*] [*area-id*] **neighbor** [*type interface-path-id*] [*neighbor-id*] [**detail**]

Example:

```
RP/0/RP0/CPU0:router# show ospf 100 neighbor
```

(Optional) Displays OSPF neighbor information on an individual interface basis.

Step 7 **clear** { **ospf** | **ospfv3** } [*process-name*] **process**

Example:

```
RP/0/  
/CPU0:router# clear ospf 100 process
```

(Optional) Resets an OSPF router process without stopping and restarting it.

Step 8 **clear**{**ospf|ospfv3**[*process-name*] **redistribution**

Example:

```
RP/0/RP0/CPU0:router#clear ospf 100 redistribution
```

Clears OSPF route redistribution.

Step 9 **clear**{**ospf|ospfv3**[*process-name*] **routes**

Example:

```
RP/0/RP0/CPU0:router#clear ospf 100 routes
```

Clears OSPF route table.

Step 10 **clear**{**ospf|ospfv3**[*process-name*] **vrf** [*vrf-name|all*] {**process|redistribution|routes|statistics** [*interface type interface-path-id|message-queue|neighbor*]}

Example:

```
RP/0/RP0/CPU0:router#clear ospf 100 vrf vrf_1 process
```

Clears OSPF route table.

Step 11 **clear** { **ospf** | **ospfv3** } [*process-name*] **statistics** [**neighbor** [*type interface-path-id*] [*ip-address*]]

Example:

```
RP/0/RP0/CPU0:router# clear ospf 100 statistics
```

(Optional) Clears the OSPF statistics of neighbor state transitions.

Stub Area

A stub area is an area that does not accept route advertisements or detailed network information external to the area. A stub area typically has only one router that interfaces the area to the rest of the autonomous system. The stub ABR advertises a single default route to external destinations into the stub area. Routers within a stub area use this route for destinations outside the area and the autonomous system. This relationship conserves LSA database space that would otherwise be used to store external LSAs flooded into the area.

Not-so-Stubby Area

A Not-so-Stubby Area (NSSA) is similar to the stub area. NSSA does not flood Type 5 external LSAs from the core into the area, but can import autonomous system external routes in a limited fashion within the area.

NSSA allows importing of Type 7 autonomous system external routes within an NSSA area by redistribution. These Type 7 LSAs are translated into Type 5 LSAs by NSSA ABRs, which are flooded throughout the whole routing domain. Summarization and filtering are supported during the translation.

Use NSSA to simplify administration if you are a network administrator that must connect a central site using OSPF to a remote site that is using a different routing protocol.

Before NSSA, the connection between the corporate site border router and remote router could not be run as an OSPF stub area because routes for the remote site could not be redistributed into a stub area, and two routing protocols needed to be maintained. A simple protocol like RIP was usually run and handled the redistribution. With NSSA, you can extend OSPF to cover the remote connection by defining the area between the corporate router and remote router as an NSSA. Area 0 cannot be an NSSA.

Configure Stub and Not-So-Stubby Area Types

This task explains how to configure the stub area and the NSSA for OSPF.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. Do one of the following:
 - **stub** [**no-summary**]
 - **nssa** [**no-redistribution**] [**default-information-originate**] [**no-summary**]
6. Do one of the following:
 - **stub**

- **nssa**

7. **default-cost** *cost*
8. Use the **commit** or **end** command.
9. Repeat this task on all other routers in the stub area or NSSA.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/RP0/CPU0:router(config)# router ospf 1
```

or

```
RP/0/RP0/CPU0:router(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IP address as the router ID.

Step 4 **area** *area-id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 1
```

Enters area configuration mode and configures a nonbackbone area for the OSPF process.

- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 5 Do one of the following:

- **stub** [**no-summary**]
- **nssa** [**no-redistribution**] [**default-information-originate**] [**no-summary**]

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# stub no summary
```

or

```
RP/0/RP0/CPU0:router(config-ospf-ar)# nssa no-redistribution
```

Defines the nonbackbone area as a stub area.

- Specify the **no-summary** keyword to further reduce the number of LSAs sent into a stub area. This keyword prevents the ABR from sending summary link-state advertisements (Type 3) in the stub area.

or

Defines an area as an NSSA.

Step 6 Do one of the following:

- **stub**
- **nssa**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# stub
```

or

```
RP/0/RP0/CPU0:router(config-ospf-ar)# nssa
```

(Optional) Turns off the options configured for stub and NSSA areas.

- If you configured the stub and NSSA areas using the optional keywords (**no-summary** , **no-redistribution** , **default-information-originate** , and **no-summary**) in Step 5, you must now reissue the **stub** and **nssa** commands without the keywords—rather than using the **no** form of the command.
- For example, the **no nssa default-information-originate** form of the command changes the NSSA area into a normal area that inadvertently brings down the existing adjacencies in that area.

Step 7 **default-cost** *cost*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)#default-cost 15
```

(Optional) Specifies a cost for the default summary route sent into a stub area or an NSSA.

- Use this command only on ABRs attached to the NSSA. Do not use it on any other routers in the area.
- The default cost is 1.

Step 8 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 9 Repeat this task on all other routers in the stub area or NSSA.

Configuring a Stub area: example

The following example shows that area 1 is configured as a stub area:

```
router ospfv3 1
router-id 10.0.0.217
area 0
interface TenGigE 0/2/0/1
area 1
stub
interface TenGigE 0/2/0/0
```

Neighbors and Adjacency for OSPF

Routers that share a segment (Layer 2 link between two interfaces) become neighbors on that segment. OSPF uses the hello protocol as a neighbor discovery and keep alive mechanism. The hello protocol involves receiving and periodically sending hello packets out each interface. The hello packets list all known OSPF neighbors on the interface. Routers become neighbors when they see themselves listed in the hello packet of the neighbor. After two routers are neighbors, they may proceed to exchange and synchronize their databases, which creates an adjacency. On broadcast and NBMA networks all neighboring routers have an adjacency.

Configure Neighbors for Nonbroadcast Networks

This task explains how to configure neighbors for a nonbroadcast network. This task is optional.

Before you begin

Configuring NBMA networks as either broadcast or nonbroadcast assumes that there are virtual circuits from every router to every router or fully meshed network.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*

3. **router-id** { *router-id* }
4. **area** *area-id*
5. **network** { **broadcast** | **non-broadcast** }
6. **dead-interval** *seconds*
7. **hello-interval** *seconds*
8. **interface** *type interface-path-id*
9. Do one of the following:
 - **neighbor** *ip-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*]
 - **neighbor** *ipv6-link-local-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*] [**database-filter** [**all**]]
10. Repeat Step 9 for all neighbors on the interface.
11. **exit**
12. **interface** *type interface-path-id*
13. Do one of the following:
 - **neighbor** *ip-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*] [**database-filter** [**all**]]
 - **neighbor** *ipv6-link-local-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*] [**database-filter** [**all**]]
14. Repeat Step 13 for all neighbors on the interface.
15. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/RP0/CPU0:router(config)# router ospf 1
```

or

```
RP/0/RP0/CPU0:router(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }**Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IP address as the router ID.

Step 4 **area** *area-id***Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters area configuration mode and configures an area for the OSPF process.

- The example configures a backbone area.
- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 5 **network** { **broadcast** | **non-broadcast** }**Example:**

```
RP/0/RP0/CPU0:router(config-ospf-ar)# network non-broadcast
```

Configures the OSPF network type to a type other than the default for a given medium.

- The example sets the network type to NBMA.

Step 6 **dead-interval** *seconds***Example:**

```
RP/0/RP0/CPU0:router(config-ospf-ar)# dead-interval 40
```

(Optional) Sets the time to wait for a hello packet from a neighbor before declaring the neighbor down.

Step 7 **hello-interval** *seconds***Example:**

```
RP/0/RP0/CPU0:router(config-ospf-ar)# hello-interval 10
```

(Optional) Specifies the interval between hello packets that OSPF sends on the interface.

Step 8 **interface** *type interface-path-id***Example:**

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface TenGigE 0/2/0/0
```

Enters interface configuration mode and associates one or more interfaces for the area configured in Step 4.

- In this example, the interface inherits the nonbroadcast network type and the hello and dead intervals from the areas because the values are not set at the interface level.

Step 9 Do one of the following:

- **neighbor** *ip-address* [**priority** *number*] [**poll-interval** *seconds*][**cost** *number*]
- **neighbor** *ipv6-link-local-address* [**priority** *number*] [**poll-interval** *seconds*][**cost** *number*] [**database-filter** [**all**]]

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# neighbor 10.20.20.1 priority 3 poll-interval 15
```

or

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# neighbor fe80::3203:a0ff:fe9d:f3fe
```

Configures the IPv4 address of OSPF neighbors interconnecting to nonbroadcast networks.

or

Configures the link-local IPv6 address of OSPFv3 neighbors.

- The *ipv6-link-local-address* argument must be in the form documented in RFC 2373 in which the address is specified in hexadecimal using 16-bit values between colons.
- The **priority** keyword notifies the router that this neighbor is eligible to become a DR or BDR. The priority value should match the actual priority setting on the neighbor router. The neighbor priority default value is zero.
- Neighbors with no specific cost configured assumes the cost of the interface, based on the **cost** command.
- The **database-filter** keyword filters outgoing LSAs to an OSPF neighbor. If you specify the **all** keyword, incoming and outgoing LSAs are filtered.

Step 10 Repeat Step 9 for all neighbors on the interface.

—

Step 11 exit

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# exit
```

Enters area configuration mode.

Step 12 **interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface TenGigE 0/3/0/0
```

Enters interface configuration mode and associates one or more interfaces for the area configured in Step 4.

- In this example, the interface inherits the nonbroadcast network type and the hello and dead intervals from the areas because the values are not set at the interface level.

Step 13 Do one of the following:

- **neighbor** *ip-address* [**priority** *number*] [**poll-interval** *seconds*][**cost** *number*] [**database-filter** [**all**]]
- **neighbor** *ipv6-link-local-address* [**priority** *number*] [**poll-interval** *seconds*][**cost** *number*] [**database-filter** [**all**]]

Example:

```
RP/0/
/CPU0:router(config-ospf-ar)# neighbor 10.34.16.6
```

or

```
RP/0/
/CPU0:router(config-ospf-ar)# neighbor fe80::3203:a0ff:fe9d:f3f
```

Configures the IPv4 address of OSPF neighbors interconnecting to nonbroadcast networks.

or

Configures the link-local IPv6 address of OSPFv3 neighbors.

- The *ipv6-link-local-address* argument must be in the form documented in RFC 2373 in which the address is specified in hexadecimal using 16-bit values between colons.
- The **priority** keyword notifies the router that this neighbor is eligible to become a DR or BDR. The priority value should match the actual priority setting on the neighbor router. The neighbor priority default value is zero.
- Neighbors with no specific cost configured assumes the cost of the interface, based on the **cost** command.
- The **database-filter** keyword filters outgoing LSAs to an OSPF neighbor. If you specify the **all** keyword, incoming and outgoing LSAs are filtered. Use with extreme caution since filtering may cause the routing topology to be seen as entirely different between two neighbors, resulting in traffic disruption or routing loops.

Step 14 Repeat Step 13 for all neighbors on the interface.

—

Step 15 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Authentication Strategies

Authentication can be specified for an entire process or area, or on an interface or a virtual link. An interface or virtual link can be configured for only one type of authentication, not both. Authentication configured for an interface or virtual link overrides authentication configured for the area or process.

If you intend for all interfaces in an area to use the same type of authentication, you can configure fewer commands if you use the **authentication** command in the area configuration submenu (and specify the **message-digest** keyword if you want the entire area to use MD5 authentication). This strategy requires fewer commands than specifying authentication for each interface.

Configure Authentication at Different Hierarchical Levels for OSPF Version 2

This task explains how to configure MD5 (secure) authentication on the OSPF router process, configure one area with plain text authentication, and then apply one interface with clear text (null) authentication.



Note Authentication configured at the interface level overrides authentication configured at the area level and the router process level. If an interface does not have authentication specifically configured, the interface inherits the authentication parameter value from a higher hierarchical level.

Before you begin

If you choose to configure authentication, you must first decide whether to configure plain text or MD5 authentication, and whether the authentication applies to all interfaces in a process, an entire area, or specific interfaces. See [OSPF Hierarchical CLI and CLI Inheritance, on page 55](#) for information about each type of authentication and when you should use a specific method for your network.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. **authentication** [**message-digest** | **null**]
5. **message-digest-key** *key-id* **md5** { *key* | **clear** *key* | **encrypted** *key* | **LINE** }
6. **area** *area-id*
7. **interface** *type interface-path-id*
8. Repeat Step 7 for each interface that must communicate, using the same authentication.
9. **exit**
10. **area** *area-id*
11. **authentication** [**message-digest** | **null**]
12. **interface** *type interface-path-id*
13. Repeat Step 12 for each interface that must communicate, using the same authentication.
14. **interface** *type interface-path-id*
15. **authentication** [**message-digest** | **null**]
16. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 2 **router ospf** *process-name*

Example:

```
RP/0/RP0/CPU0:router(config)# router ospf 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Step 4 **authentication** [**message-digest** | **null**]

Example:

```
RP/0/RP0/CPU0:router(config-ospf)#authentication message-digest
```

Enables MD5 authentication for the OSPF process.

- This authentication type applies to the entire router process unless overridden by a lower hierarchical level such as the area or interface.

Step 5 **message-digest-key** *key-id* **md5** { *key* | **clear** *key* | **encrypted** *key* | **LINE** }

Example:

```
RP/0/RP0/CPU0:router(config-ospf)#message-digest-key 4 md5 yourkey
```

Specifies the MD5 authentication key for the OSPF process.

- The neighbor routers must have the same key identifier.

Step 6 **area** *area-id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters area configuration mode and configures a backbone area for the OSPF process.

Step 7 **interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface TenGigE 0/1/0/3
```

Enters interface configuration mode and associates one or more interfaces to the backbone area.

- All interfaces inherit the authentication parameter values specified for the OSPF process (Step 4, Step 5, and Step 6).

Step 8 Repeat Step 7 for each interface that must communicate, using the same authentication.

—

Step 9 **exit**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# exit
```

Enters area OSPF configuration mode.

Step 10 **area** *area-id***Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# area 1
```

Enters area configuration mode and configures a nonbackbone area 1 for the OSPF process.

- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 11 **authentication** [**message-digest** | **null**]**Example:**

```
RP/0/RP0/CPU0:router(config-ospf-ar)# authentication
```

Enables Type 1 (plain text) authentication that provides no security.

- The example specifies plain text authentication (by not specifying a keyword). Use the **authentication-key** command in interface configuration mode to specify the plain text password.

Step 12 **interface** *type interface-path-id***Example:**

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface TenGigE 0/1/0/0
```

Enters interface configuration mode and associates one or more interfaces to the nonbackbone area 1 specified in Step 7.

- All interfaces configured inherit the authentication parameter values configured for area 1.

Step 13 Repeat Step 12 for each interface that must communicate, using the same authentication.

—

Step 14 **interface** *type interface-path-id***Example:**

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface TenGigE 0/3/0/0
```

Enters interface configuration mode and associates one or more interfaces to a different authentication type.

Step 15 **authentication** [**message-digest** | **null**]**Example:**

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# authentication null
```

Specifies no authentication on TenGigE 0/3/0/0, overriding the plain text authentication specified for area 1.

- By default, all of the interfaces configured in the same area inherit the same authentication parameter values of the area.

Step 16 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Control Frequency That Same LSA Is Originated or Accepted for OSPF

This task explains how to tune the convergence time of OSPF routes in the routing table when many LSAs need to be flooded in a very short time interval.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. Perform Step 5 or Step 6 or both to control the frequency that the same LSA is originated or accepted.
5. **timers lsa refresh** *seconds*
6. **timers lsa min-arrival** *seconds*
7. **timers lsa group-pacing** *seconds*
8. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 2 Do one of the following:

- **router ospf** *process-name*

- **router ospfv3** *process-name*

Example:

```
RP/0/RP0/CPU0:router:router(config)# router ospf 1
```

or

```
RP/0/RP0/CPU0:router(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }**Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IP address as the router ID.

Step 4 Perform Step 5 or Step 6 or both to control the frequency that the same LSA is originated or accepted.

—

Step 5 **timers lsa refresh** *seconds***Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# timers lsa refresh 1800
```

Sets how often self-originated LSAs should be refreshed, in seconds.

- The default is 1800 seconds for both OSPF and OSPFv3.

Step 6 **timers lsa min-arrival** *seconds***Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# timers lsa min-arrival 2
```

Limits the frequency that new processes of any particular OSPF Version 2 LSA can be accepted during flooding.

- The default is 1 second.

Step 7 **timers lsa group-pacing** *seconds***Example:**

```
RP/0/  
/CPU0:router(config-ospf)# timers lsa group-pacing 1000
```

Changes the interval at which OSPF link-state LSAs are collected into a group for flooding.

- The default is 240 seconds.

Step 8 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Virtual Link and Transit Area for OSPF

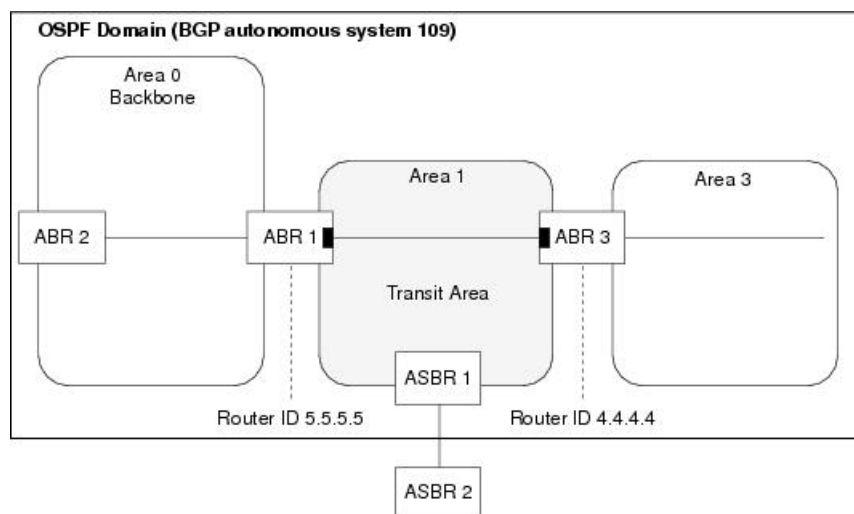
In OSPF, routing information from all areas is first summarized to the backbone area by ABRs. The same ABRs, in turn, propagate such received information to their attached areas. Such hierarchical distribution of routing information requires that all areas be connected to the backbone area (Area 0). Occasions might exist for which an area must be defined, but it cannot be physically connected to Area 0. Examples of such an occasion might be if your company makes a new acquisition that includes an OSPF area, or if Area 0 itself is partitioned.

In the case in which an area cannot be connected to Area 0, you must configure a virtual link between that area and Area 0. The two endpoints of a virtual link are ABRs, and the virtual link must be configured in both routers. The common nonbackbone area to which the two routers belong is called a transit area. A virtual link specifies the transit area and the router ID of the other virtual endpoint (the other ABR).

A virtual link cannot be configured through a stub area or NSSA.

Figure 1: Virtual Link to Area 0

This figure illustrates a virtual link from Area 3 to Area 0.



Create Virtual Link

This task explains how to create a virtual link to your backbone (area 0) and apply MD5 authentication. You must perform the steps described on both ABRs, one at each end of the virtual link.



Note After you explicitly configure area parameter values, they are inherited by all interfaces bound to that area—unless you override the values and configure them explicitly for the interface.

Before you begin

The following prerequisites must be met before creating a virtual link with MD5 authentication to area 0:

- You must have the router ID of the neighbor router at the opposite end of the link to configure the local router. You can execute the **show ospf** or **show ospfv3** command on the remote router to get its router ID.
- For a virtual link to be successful, you need a stable router ID at each end of the virtual link. You do not want them to be subject to change, which could happen if they are assigned by default. . Therefore, we recommend that you perform one of the following tasks before configuring a virtual link:
 - Use the **router-id** command to set the router ID. This strategy is preferable.
 - Configure a loopback interface so that the router has a stable router ID.
- Before configuring your virtual link for OSPF Version 2, you must decide whether to configure plain text authentication, MD5 authentication, or no authentication (which is the default). Your decision determines whether you need to perform additional tasks related to authentication.

SUMMARY STEPS

1. Do one of the following:
 - **show ospf** [*process-name*]
 - **show ospfv3** [*process-name*]
2. **configure**
3. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
4. **router-id** { *router-id* }
5. **area** *area-id*
6. **virtual-link** *router-id*
7. **authentication message-digest**
8. **message-digest-key** *key-id* **md5** { *key* | **clear** *key* | **encrypted** *key* }
9. Repeat all of the steps in this task on the ABR that is at the other end of the virtual link. Specify the same key ID and key that you specified for the virtual link on this router.
10. Use the **commit** or **end** command.
11. Do one of the following:

- **show ospf** [*process-name*] [*area-id*] **virtual-links**
- **show ospfv3** [*process-name*] **virtual-links**

DETAILED STEPS

Step 1 Do one of the following:

- **show ospf** [*process-name*]
- **show ospfv3** [*process-name*]

Example:

```
RP/0//CPU0:router# show ospf
```

or

```
RP/0//CPU0:router# show ospfv3
```

(Optional) Displays general information about OSPF routing processes.

- The output displays the router ID of the local router. You need this router ID to configure the other end of the link.

Step 2 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 3 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0//CPU0:router(config)# router ospf 1
```

or

```
RP/0//CPU0:router(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 4 **router-id** { *router-id* }

Example:

```
RP/0//CPU0:router(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IPv4 address as the router ID.

Step 5 **area** *area-id*

Example:

```
RP/0//CPU0:router(config-ospf)# area 1
```

Enters area configuration mode and configures a nonbackbone area for the OSPF process.

- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 6 **virtual-link** *router-id*

Example:

```
RRP/0//CPU0:router(config-ospf-ar)# virtual-link 10.3.4.5
```

Defines an OSPF virtual link.

- See .

Step 7 **authentication message-digest**

Example:

```
RP/0//CPU0:router(config-ospf-ar-vl)#authentication message-digest
```

Selects MD5 authentication for this virtual link.

Step 8 **message-digest-key** *key-id* **md5** { *key* | **clear** *key* | **encrypted** *key* }

Example:

```
RP/0//CPU0:router(config-ospf-ar-vl)#message-digest-key 4 md5 yourkey
```

Defines an OSPF virtual link.

- See to understand a virtual link.
- The *key-id* argument is a number in the range from 1 to 255. The *key* argument is an alphanumeric string of up to 16 characters. The routers at both ends of the virtual link must have the same key identifier and key to be able to route OSPF traffic.
- The **authentication-key** *key* command is not supported for OSPFv3.
- Once the key is encrypted it must remain encrypted.

Step 9 Repeat all of the steps in this task on the ABR that is at the other end of the virtual link. Specify the same key ID and key that you specified for the virtual link on this router.

—

Step 10 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 11 Do one of the following:

- **show ospf** [*process-name*] [*area-id*] **virtual-links**
- **show ospfv3** [*process-name*] **virtual-links**

Example:

```
RP/0//CPU0:router# show ospf 1 2 virtual-links
```

or

```
RP/0//CPU0:router# show ospfv3 1 virtual-links
```

(Optional) Displays the parameters and the current state of OSPF virtual links.

Creating virtual link- example

ABR 1 Configuration

ABR 2 Configuration

In the following example, the **show ospfv3 virtual links** command verifies that the OSPF_VL0 virtual link to the OSPFv3 neighbor is up, the ID of the virtual link interface is 2, and the IPv6 address of the virtual link endpoint is 2003:3000::1.

```
show ospfv3 virtual-links
```

```
Virtual Links for OSPFv3 1
```

```
Virtual Link OSPF_VL0 to router 10.0.0.3 is up
  Interface ID 2, IPv6 address 2003:3000::1
  Run as demand circuit
  DoNotAge LSA allowed.
  Transit area 0.1.20.255, via interface TenGigE 0/1/0/1 Cost of using 2
  Transmit Delay is 5 sec,
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
  Hello due in 00:00:02
  Adjacency State FULL (Hello suppressed)
  Index 0/2/3, retransmission queue length 0, number of retransmission 1
  First 0(0)/0(0)/0(0) Next 0(0)/0(0)/0(0)
  Last retransmission scan length is 1, maximum is 1
  Last retransmission scan time is 0 msec, maximum is 0 msec
```

```
Check for lines:
```

```
Virtual Link OSPF_VL0 to router 10.0.0.3 is up
  Adjacency State FULL (Hello suppressed)
```



```
State is up and Adjacency State is FULL
```

This example shows how to set up a virtual link to connect the backbone through area 1 for the OSPFv3 topology that consists of areas 0 and 1 and virtual links 10.0.0.217 and 10.0.0.212:

```
router ospfv3 1
router-id 10.0.0.217
area 0
interface TenGigE 0/2/0/1
area 1
virtual-link 10.0.0.212
interface TenGigE 0/2/0/0
```

```
router ospfv3 1
router-id 10.0.0.212
area 0
interface TenGigE 0/3/0/1
area 1
virtual-link 10.0.0.217
interface TenGigE 0/2/0/0
```

In this example, all interfaces on router ABR1 use MD5 authentication:

```
router ospf ABR1
router-id 10.10.10.10
authentication message-digest
message-digest-key 100 md5 0 cisco
area 0
interface TenGigE 0/2/0/1
interface TenGigE 0/3/0/0
area 1
interface TenGigE 0/2/0/0
virtual-link 10.10.5.5
!
!
```

In this example, only area 1 interfaces on router ABR3 use MD5 authentication:

```
router ospf ABR2
router-id 10.10.5.5
area 0
area 1
authentication message-digest
message-digest-key 100 md5 0 cisco
interface TenGigE 0/9/0/1
virtual-link 10.10.10.10
area 3
interface Loopback 0
interface TenGigE 0/9/0/0
!
```

Summarize Subnetwork LSAs on OSPF ABR

If you configured two or more subnetworks when you assigned your IP addresses to your interfaces, you might want the software to summarize (aggregate) into a single LSA all of the subnetworks that the local area advertises to another area. Such summarization would reduce the number of LSAs and thereby conserve

network resources. This summarization is known as interarea route summarization. It applies to routes from within the autonomous system. It does not apply to external routes injected into OSPF by way of redistribution.

This task configures OSPF to summarize subnetworks into one LSA, by specifying that all subnetworks that fall into a range are advertised together. This task is performed on an ABR only.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. Do one of the following:
 - **range** *ip-address mask* [**advertise** | **not-advertise**]
 - **range** *ipv6-prefix / prefix-length* [**advertise** | **not-advertise**]
6. **interface** *type interface-path-id*
7. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
Enters mode.
```

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/RP0/CPU0:router(config)# router ospf 1
or
```

```
RP/0/RP0/CPU0:router(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IPv4 address as the router ID.

Step 4 `area area-id`**Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# area
```

Enters area configuration mode and configures a nonbackbone area for the OSPF process.

- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 5 Do one of the following:

- **range** *ip-address mask* [**advertise** | **not-advertise**]
- **range** *ipv6-prefix / prefix-length* [**advertise** | **not-advertise**]

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# range 192.168.0.0 255.255.0.0 advertise
```

or

```
RP/0/RP0/CPU0:router(config-ospf-ar)# range 4004:f000::/32 advertise
```

Consolidates and summarizes OSPF routes at an area boundary.

- The **advertise** keyword causes the software to advertise the address range of subnetworks in a Type 3 summary LSA.
- The **not-advertise** keyword causes the software to suppress the Type 3 summary LSA, and the subnetworks in the range remain hidden from other areas.
- In the first example, all subnetworks for network 192.168.0.0 are summarized and advertised by the ABR into areas outside the backbone.
- In the second example, two or more IPv4 interfaces are covered by a 192.x.x network.

Step 6 `interface type interface-path-id`**Example:**

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface TenGigE 0/0/0/0
```

Enters interface configuration mode and associates one or more interfaces to the area.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** — Exits the configuration session without committing the configuration changes.
- **Cancel** — Remains in the configuration session, without committing the configuration changes.

Example

The following example shows the prefix range 2300::/16 summarized from area 1 into the backbone:

```
router ospfv3 1
  router-id 192.168.0.217
  area 0
    interface TenGigE 0/0/0/0
  area 1
    range 2300::/16
    interface TenGigE 0/0/0/0
```

Route Redistribution for OSPF

Redistribution allows different routing protocols to exchange routing information. This technique can be used to allow connectivity to span multiple routing protocols. It is important to remember that the **redistribute** command controls redistribution into an OSPF process and not from OSPF.

Redistribute Routes into OSPF

This task redistributes routes from an IGP (could be a different OSPF process) into OSPF.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **redistribute** *protocol* [*process-id*] { **level-1** | **level-1-2** | **level-2** } [**metric** *metric-value*] [**metric-type** *type-value*] [**match** { **external** [**1** | **2**] }] [**tag** *tag-value*] [**route-policy** *policy-name*]
5. Do one of the following:
 - **summary-prefix** *address mask* [**not-advertise**] [**tag** *tag*]
 - **summary-prefix** *ipv6-prefix / prefix-length* [**not-advertise**] [**tag** *tag*]
6. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
Enters mode.
```

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/RP0/CPU0:router(config)# router ospf 1
or
```

```
RP/0/RP0/CPU0:router(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RRP/0/RP0/CPU0:router(config-ospf)# router-id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IPv4 address as the router ID.

Step 4 **redistribute** *protocol* [*process-id*] { **level-1** | **level-1-2** | **level-2** } [**metric** *metric-value*] [**metric-type** *type-value*] [**match** { **external** [**1** | **2**] }] [**tag** *tag-value*] [**route-policy** *policy-name*]

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# redistribute bgp 100
```

or

```
RP/0/RP0/CPU0:router(config-router)# redistribute bgp 110
```

Redistributes OSPF routes from one routing domain to another routing domain.

or

Redistributes OSPFv3 routes from one routing domain to another routing domain.

- This command causes the router to become an ASBR by definition.

- OSPF tags all routes learned through redistribution as external.
- The protocol and its process ID, if it has one, indicate the protocol being redistributed into OSPF.
- The metric is the cost you assign to the external route. The default is 20 for all protocols except BGP, whose default metric is 1.
- The OSPF example redistributes BGP autonomous system 1, Level 1 routes into OSPF as Type 2 external routes.
- The OSPFv3 example redistributes BGP autonomous system 1, Level 1 and 2 routes into OSPF. The external link type associated with the default route advertised into the OSPFv3 routing domain is the Type 1 external route.

Note RPL is not supported for OSPFv3.

Step 5 Do one of the following:

- **summary-prefix** *address mask* [**not-advertise**] [**tag tag**]
- **summary-prefix** *ipv6-prefix / prefix-length* [**not-advertise**] [**tag tag**]

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# summary-prefix 10.1.0.0 255.255.0.0
```

or

```
RP/0/RP0/CPU0:router(config-router)# summary-prefix 2010:11:22::/32
```

(Optional) Creates aggregate addresses for OSPF.

or

(Optional) Creates aggregate addresses for OSPFv3.

- This command provides external route summarization of the non-OSPF routes.
- External ranges that are being summarized should be contiguous. Summarization of overlapping ranges from two different routers could cause packets to be sent to the wrong destination.
- This command is optional. If you do not specify it, each route is included in the link-state database and advertised in LSAs.
- In the OSPFv2 example, the summary address 10.1.0.0 includes address 10.1.1.0, 10.1.2.0, 10.1.3.0, and so on. Only the address 10.1.0.0 is advertised in an external LSA.
- In the OSPFv3 example, the summary address 2010:11:22::/32 has addresses such as 2010:11:22:0:1000::1, 2010:11:22:0:2000:679:1, and so on. Only the address 2010:11:22::/32 is advertised in the external LSA.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.

- **Cancel**—Remains in the configuration session, without committing the configuration changes.

Example

The following example uses prefix lists to limit the routes redistributed from other protocols.

Only routes with 9898:1000 in the upper 32 bits and with prefix lengths from 32 to 64 are redistributed from BGP 42. Only routes *not* matching this pattern are redistributed from BGP 1956.

```
ipv6 prefix-list list1
 seq 10 permit 9898:1000::/32 ge 32 le 64
ipv6 prefix-list list2
 seq 10 deny 9898:1000::/32 ge 32 le 64
 seq 20 permit ::/0 le 128
router ospfv3 1
 router-id 10.0.0.217
 redistribute bgp 42
 redistribute bgp 1956
 distribute-list prefix-list list1 out bgp 42
 distribute-list prefix-list list2 out bgp 1956
 area 1
 interface TenGigE 0/2/0/0
```

OSPF Shortest Path First Throttling

OSPF SPF throttling makes it possible to configure SPF scheduling in millisecond intervals and to potentially delay SPF calculations during network instability. SPF is scheduled to calculate the Shortest Path Tree (SPT) when there is a change in topology. One SPF run may include multiple topology change events.

The interval at which the SPF calculations occur is chosen dynamically and based on the frequency of topology changes in the network. The chosen interval is within the boundary of the user-specified value ranges. If network topology is unstable, SPF throttling calculates SPF scheduling intervals to be longer until topology becomes stable.

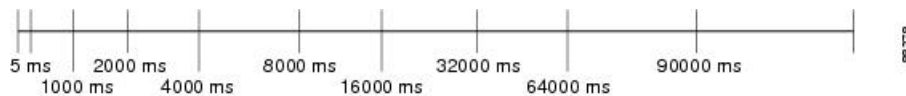
SPF calculations occur at the interval set by the **timers throttle spf** command. The wait interval indicates the amount of time to wait until the next SPF calculation occurs. Each wait interval after that calculation is twice as long as the previous interval until the interval reaches the maximum wait time specified.

The SPF timing can be better explained using an example. In this example, the start interval is set at 5 milliseconds (ms), initial wait interval at 1000 ms, and maximum wait time at 90,000 ms.

```
timers spf 5 1000 90000
```

Figure 2: SPF Calculation Intervals Set by the timers spf Command

This figure shows the intervals at which the SPF calculations occur as long as at least one topology change event is received in a given wait interval.

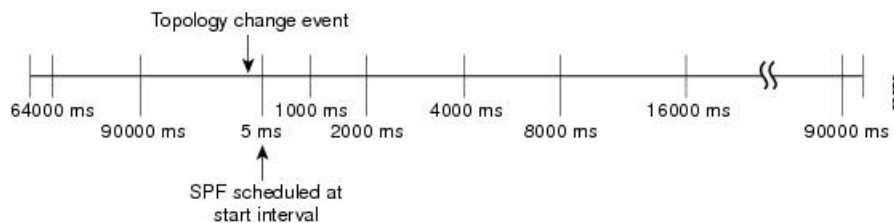


Notice that the wait interval between SPF calculations doubles when at least one topology change event is received during the previous wait interval. After the maximum wait time is reached, the wait interval remains the same until the topology stabilizes and no event is received in that interval.

If the first topology change event is received after the current wait interval, the SPF calculation is delayed by the amount of time specified as the start interval. The subsequent wait intervals continue to follow the dynamic pattern.

If the first topology change event occurs after the maximum wait interval begins, the SPF calculation is again scheduled at the start interval and subsequent wait intervals are reset according to the parameters specified in the **timers throttle spf** command. Notice in [Figure 3: Timer Intervals Reset After Topology Change Event, on page 32](#) that a topology change event was received after the start of the maximum wait time interval and that the SPF intervals have been reset.

Figure 3: Timer Intervals Reset After Topology Change Event



Configure OSPF Shortest Path First Throttling

This task explains how to configure SPF scheduling in millisecond intervals and potentially delay SPF calculations during times of network instability. This task is optional.

SUMMARY STEPS

1. **configure**
2. Do one of the following:
 - **router ospf** *process-name*
 - **router ospfv3** *process-name*
3. **router-id** { *router-id* }
4. **timers throttle spf** *spf-start* *spf-hold* *spf-max-wait*
5. **area** *area-id*
6. **interface** *type interface-path-id*
7. Use the **commit** or **end** command.
8. Do one of the following:
 - **show ospf** [*process-name*]
 - **show ospfv3** [*process-name*]

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
Enters mode.
```

Step 2 Do one of the following:

- **router ospf** *process-name*
- **router ospfv3** *process-name*

Example:

```
RP/0/RP0/CPU0:router(config)# router ospf 1
or
RP/0/RP0/CPU0:router(config)# router ospfv3 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

or

Enables OSPFv3 routing for the specified routing process and places the router in router ospfv3 configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# router-id 192.168.4.3
Configures a router ID for the OSPF process.
```

Note We recommend using a stable IPv4 address as the router ID.

Step 4 **timers throttle spf** *spf-start spf-hold spf-max-wait*

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# timers throttle spf 10 4800 90000
Sets SPF throttling timers.
```

Step 5 **area** *area-id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
Enters area configuration mode and configures a backbone area.
```

- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 6 `interface` *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface TenGigE 0/0/0/0
```

Enters interface configuration mode and associates one or more interfaces to the area.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 8 Do one of the following:

- **show ospf** [*process-name*]
- **show ospfv3** [*process-name*]

Example:

```
RP/0/RP0/CPU0:router# show ospf 1
```

or

```
RP/0/RP0/CPU0:router# RP/0/RP0/CPU0:router# show ospfv3 2
```

(Optional) Displays SPF throttling timers.

Graceful Restart for OSPFv3

The OSPFv3 Graceful Shutdown feature preserves the data plane capability in these circumstances:

- Planned OSPFv3 process restart, such as a restart resulting from a software upgrade or downgrade
- Unplanned OSPFv3 process restart, such as a restart resulting from a process crash

In addition, OSPFv3 will unilaterally shutdown and enter the exited state when a critical memory event, indicating the processor is critically low on available memory, is received from the sysmon watch dog process.

This feature supports nonstop data forwarding on established routes while the OSPFv3 routing protocol restarts. Therefore, this feature enhances high availability of IPv6 forwarding.

Configure OSPFv3 Graceful Restart

This task explains how to configure a graceful restart for an OSPFv3 process. This task is optional.

SUMMARY STEPS

1. **configure**
2. **router ospfv3** *process-name*
3. **graceful-restart**
4. **graceful-restart lifetime**
5. **graceful-restart interval** *seconds*
6. **graceful-restart helper disable**
7. Use the **commit** or **end** command.
8. **show ospfv3** [*process-name* [*area-id*]] **database grace**

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 2 **router ospfv3** *process-name*

Example:

```
RP/0/RP0/CPU0:router(config)# router ospfv3 test
```

Enters router configuration mode for OSPFv3. The process name is a WORD that uniquely identifies an OSPF routing process. The process name is any alphanumeric string no longer than 40 characters without spaces.

Step 3 **graceful-restart**

Example:

```
RP/0/RP0/CPU0:router(config-ospfv3)#graceful-restart
```

Enables graceful restart on the current router.

Step 4 **graceful-restart lifetime**

Example:

```
RP/0/RP0/CPU0:router(config-ospfv3)# graceful-restart lifetime 120
```

Specifies a maximum duration for a graceful restart.

- The default lifetime is 95 seconds.
- The range is 90 to 3600 seconds.

Step 5 **graceful-restart interval** *seconds*

Example:

```
RP/0/RP0/CPU0:router(config-ospfv3)# graceful-restart interval 120
```

Specifies the interval (minimal time) between graceful restarts on the current router.

- The default value for the interval is 90 seconds.
- The range is 90 to 3600 seconds.

Step 6 **graceful-restart helper disable**

Example:

```
RP/0/RP0/CPU0:router(config-ospfv3)# graceful-restart helper disable
```

Disables the helper capability.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 8 **show ospfv3 [process-name [area-id]] database grace**

Example:

```
RP/0/RP0/CPU0:router# show ospfv3 1 database grace
```

Displays the state of the graceful restart link.

Display Information About Graceful Restart

This section describes the tasks you can use to display information about a graceful restart.

- To see if the feature is enabled and when the last graceful restart ran, use the **show ospf** command. To see details for an OSPFv3 instance, use the **show ospfv3 process-name [area-id] database grace** command.

Displaying the State of the Graceful Restart Feature

The following screen output shows the state of the graceful restart capability on the local router:

```
RP/0/RP0/CPU0:router# show ospfv3 1 database grace

Routing Process "ospfv3 1" with ID 2.2.2.2
Initial SPF schedule delay 5000 msec
Minimum hold time between two consecutive SPFs 10000 msec
Maximum wait time between two consecutive SPFs 10000 msec
Initial LSA throttle delay 0 msec
Minimum hold time for LSA throttle 5000 msec
Maximum wait time for LSA throttle 5000 msec
Minimum LSA arrival 1000 msec
```

```

LSA group pacing timer 240 secs
Interface flood pacing timer 33 msec
Retransmission pacing timer 66 msec
Maximum number of configured interfaces 255
Number of external LSA 0. Checksum Sum 00000000
Number of areas in this router is 1. 1 normal 0 stub 0 nssa
Graceful Restart enabled, last GR 11:12:26 ago (took 6 secs)
Area BACKBONE(0)
  Number of interfaces in this area is 1
  SPF algorithm executed 1 times
  Number of LSA 6. Checksum Sum 0x0268a7
  Number of DCbitless LSA 0
  Number of indication LSA 0
  Number of DoNotAge LSA 0
  Flood list length 0

```

OSPFv2OSPF SPF Prefix Prioritization

The OSPFv2 OSPF SPF Prefix Prioritization feature enables an administrator to converge, in a faster mode, important prefixes during route installation.

When a large number of prefixes must be installed in the Routing Information Base (RIB) and the Forwarding Information Base (FIB), the update duration between the first and last prefix, during SPF, can be significant.

In networks where time-sensitive traffic (for example, VoIP) may transit to the same router along with other traffic flows, it is important to prioritize RIB and FIB updates during SPF for these time-sensitive prefixes.

The OSPFv2OSPF SPF Prefix Prioritization feature provides the administrator with the ability to prioritize important prefixes to be installed, into the RIB during SPF calculations. Important prefixes converge faster among prefixes of the same route type per area. Before RIB and FIB installation, routes and prefixes are assigned to various priority batch queues in the OSPF local RIB, based on specified route policy. The RIB priority batch queues are classified as "critical," "high," "medium," and "low," in the order of decreasing priority.

When enabled, prefix alters the sequence of updating the RIB with this prefix priority:

Critical > High > Medium > Low

As soon as prefix priority is configured, /32 prefixes are no longer preferred by default; they are placed in the low-priority queue, if they are not matched with higher-priority policies. Route policies must be devised to retain /32s in the higher-priority queues (high-priority or medium-priority queues).

Priority is specified using route policy, which can be matched based on IP addresses or route tags. During SPF, a prefix is checked against the specified route policy and is assigned to the appropriate RIB batch priority queue.

These are examples of this scenario:

- If only high-priority route policy is specified, and no route policy is configured for a medium priority:
 - Permitted prefixes are assigned to a high-priority queue.
 - Unmatched prefixes, including /32s, are placed in a low-priority queue.
- If both high-priority and medium-priority route policies are specified, and no maps are specified for critical priority:
 - Permitted prefixes matching high-priority route policy are assigned to a high-priority queue.

- Permitted prefixes matching medium-priority route policy are placed in a medium-priority queue.
 - Unmatched prefixes, including /32s, are moved to a low-priority queue.
- If both critical-priority and high-priority route policies are specified, and no maps are specified for medium priority:
 - Permitted prefixes matching critical-priority route policy are assigned to a critical-priority queue.
 - Permitted prefixes matching high-priority route policy are assigned to a high-priority queue.
 - Unmatched prefixes, including /32s, are placed in a low-priority queue.
 - If only medium-priority route policy is specified and no maps are specified for high priority or critical priority:
 - Permitted prefixes matching medium-priority route policy are assigned to a medium-priority queue.
 - Unmatched prefixes, including /32s, are placed in a low-priority queue.

Use the **[no] spf prefix-priority route-policy** *rpl* command to prioritize OSPFv2OSPF prefix installation into the global RIB during SPF.

SPF prefix prioritization is disabled by default. In disabled mode, /32 prefixes are installed into the global RIB, before other prefixes. If SPF prioritization is enabled, routes are matched against the route-policy criteria and are assigned to the appropriate priority queue based on the SPF priority set. Unmatched prefixes, including /32s, are placed in the low-priority queue.

If all /32s are desired in the high-priority queue or medium-priority queue, configure this single route map:

```
prefix-set ospf-medium-prefixes
 0.0.0.0/0 ge 32
end-set
```

Configure OSPFv2 OSPF SPF Prefix Prioritization

Perform this task to configure OSPFv2 OSPF SPF (shortest path first) prefix prioritization.

SUMMARY STEPS

1. **configure**
2. **prefix-set** *prefix-set name*
3. **route-policy** *route-policy name* **if destination in** *prefix-set name* **then set** **spf-priority** {critical | high | medium} **endif**
4. Use one of these commands:
 - **router ospf** *ospf-name*
 - **router ospfv3** *ospfv3-name*
5. **router ospf** *ospf name*
6. **spf prefix-priority route-policy** *route-policy name*
7. Use the **commit** or **end** command.
8. **show rpl route-policy** *route-policy name* **detail**

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 2 **prefix-set** *prefix-set name*

Example:

```
RP/0/RP0/CPU0:router(config)#prefix-set ospf-critical-prefixes
RP/0/RP0/CPU0:router(config-pfx)#66.0.0.0/16
RP/0/RP0/CPU0:router(config-pfx)#end-set
```

Configures the prefix set.

Step 3 **route-policy** *route-policy name* **if destination in** *prefix-set name* **then set** **spf-priority** {critical | high | medium} **endif**

Example:

```
RP/0/RP0/CPU0:router#route-policy ospf-spf-priority
RP/0/RP0/CPU0:router(config-rpl)#if destination in ospf-critical-prefixes then
  set spf-priority critical
endif
RP/0/RP0/CPU0:router(config-rpl)#end-policy
```

Configures route policy and sets OSPF SPF priority.

Step 4 Use one of these commands:

- **router ospf** *ospf-name*
- **router ospfv3** *ospfv3-name*

Example:

```
RP/0/RP0/CPU0:router# router ospf 1
```

Or

```
RP/0/RP0/CPU0:router# router ospfv3 1
```

Enters Router OSPF configuration mode.

Step 5 **router ospf** *ospf name*

Example:

```
RP/0/RP0/CPU0:router# router ospf 1
```

Enters Router OSPF configuration mode.

Step 6 **spf prefix-priority route-policy** *route-policy name*

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# spf prefix-priority route-policy ospf-spf-priority
```

Or

```
RP/0/RP0/CPU0:router(config-ospfv3)#spf prefix-priority route-policy ospf3-spf-priority
```

Configures SPF prefix-priority for the defined route policy.

Note Configure the **spf prefix-priority** command under router OSPF.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 8 **show rpl route-policy route-policy name detail**

Example:

```
RP/0/RP0/CPU0:router#show rpl route-policy ospf-spf-priority detail
prefix-set ospf-critical-prefixes
 66.0.0.0/16
end-set
!
route-policy ospf-spf-priority
 if destination in ospf-critical-prefixes then
   set spf-priority critical
 endif
end-policy
!
```

Displays the set SPF prefix priority.

OSPFv2

OSPFv3

This example shows how to configure /32 prefixes as medium-priority, in general, in addition to placing some /32 and /24 prefixes in critical-priority and high-priority queues:

```
prefix-set ospf-critical-prefixes
 192.41.5.41/32,
 11.1.3.0/24,
 192.168.0.44/32
end-set
!
prefix-set ospf-high-prefixes
 44.4.10.0/24,
 192.41.4.41/32,
```



```
41.4.41.41/32
end-set
!
prefix-set ospf-medium-prefixes
0.0.0.0/0 ge 32
end-set
!

route-policy ospf-priority
  if destination in ospf-high-prefixes then
    set spf-priority high
  else
    if destination in ospf-critical-prefixes then
      set spf-priority critical
    else
      if destination in ospf-medium-prefixes then
        set spf-priority medium
      endif
    endif
  endif
end-policy

router ospf 1
  spf prefix-priority route-policy ospf-priority
  area 0
    interface TenGigE 0/3/0/0
    !
  !
  area 3
    interface TenGigE 0/2/0/0
    !
  !
  area 8
    interface TenGigE 0/2/0/0

router ospfv3 1
  spf prefix-priority route-policy ospf-priority
  area 0
    interface TenGigE 0/3/0/0
    !
  !
  area 3
    interface TenGigE 0/2/0/0
    !
  !
  area 8
    interface TenGigE 0/2/0/0
```

Multi-Area Adjacency for OSPF Version 2

The multi-area adjacency feature for OSPFv2 allows a link to be configured on the primary interface in more than one area so that the link could be considered as an intra-area link in those areas and configured as a preference over more expensive paths.

This feature establishes a point-to-point unnumbered link in an OSPF area. A point-to-point link provides a topological path for that area, and the primary adjacency uses the link to advertise the link consistent with draft-ietf-ospf-multi-area-adj-06.

The following are multi-area interface attributes and limitations:

- Exists as a logical construct over an existing primary interface for OSPF; however, the neighbor state on the primary interface is independent of the multi-area interface.
- Establishes a neighbor relationship with the corresponding multi-area interface on the neighboring router. A mixture of multi-area and primary interfaces is not supported.
- Advertises an unnumbered point-to-point link in the router link state advertisement (LSA) for the corresponding area when the neighbor state is full.
- Created as a point-to-point network type. You can configure multi-area adjacency on any interface where only two OSPF speakers are attached. In the case of native broadcast networks, the interface must be configured as an OSPF point-to-point type using the **network point-to-point** command to enable the interface for a multi-area adjacency.
- Inherits the Bidirectional Forwarding Detection (BFD) characteristics from its primary interface. BFD is not configurable under a multi-area interface; however, it is configurable under the primary interface.

Configure Multi-area Adjacency

This task explains how to create multiple areas on an OSPF primary interface.

Before you begin



Note You can configure multi-area adjacency on any interface where only two OSPF speakers are attached. In the case of native broadcast networks, the interface must be configured as an OSPF point-to-point type using the **network point-to-point** command to enable the interface for a multi-area adjacency.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **area** *area-id*
4. **interface** *type interface-path-id*
5. **area** *area-id*
6. **multi-area-interface** *type interface-path-id*
7. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 2 `router ospf` *process-name***Example:**

```
RP/0/RP0/CPU0:router(config)# router ospf 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 `area` *area-id***Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# area 0
```

Enters area configuration mode and configures a backbone area.

- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 4 `interface` *type interface-path-id***Example:**

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface Serial 0/1/0/3
```

Enters interface configuration mode and associates one or more interfaces to the area.

Step 5 `area` *area-id***Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# area 1
```

Enters area configuration mode and configures an area used for multiple area adjacency.

- The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 6 `multi-area-interface` *type interface-path-id***Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# multi-area-interface Serial 0/1/0/3
```

Enables multiple adjacencies for different OSPF areas and enters multi-area interface configuration mode

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.

- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Example

The multi-area interface inherits the interface characteristics from its primary interface, but some interface characteristics can be configured under the multi-area interface configuration mode as shown below:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# multi-area-interface TenGigE 0/0/0/0
RP/0/RP0/CPU0:router(config-ospf-ar-mif)# ?
 authentication          Enable authentication
 authentication-key      Authentication password (key)
 cost                   Interface cost
 cost-fallback           Cost when cumulative bandwidth goes below the threshold
 database-filter        Filter OSPF LSA during synchronization and flooding
 dead-interval          Interval after which a neighbor is declared dead
 distribute-list         Filter networks in routing updates
 hello-interval         Time between HELLO packets
 message-digest-key     Message digest authentication password (key)
 mtu-ignore             Enable/Disable ignoring of MTU in DBD packets
 packet-size            Customize size of OSPF packets upto MTU
 retransmit-interval    Time between retransmitting lost link state advertisements
 transmit-delay         Estimated time needed to send link-state update packet

RP/0/RP0/CPU0:router(config-ospf-ar-mif)#
```

Label Distribution Protocol IGP Auto-configuration for OSPF

Label Distribution Protocol (LDP) Interior Gateway Protocol (IGP) auto-configuration simplifies the procedure to enable LDP on a set of interfaces used by an IGP instance, such as OSPF. LDP IGP auto-configuration can be used on a large number of interfaces (for example, when LDP is used for transport in the core) and on multiple OSPF instances simultaneously.

This feature supports the IPv4 unicast address family for the default VPN routing and forwarding (VRF) instance.

LDP IGP auto-configuration can also be explicitly disabled on an individual interface basis under LDP using the **igp auto-config disable** command. This allows LDP to receive all OSPF interfaces minus the ones explicitly disabled.

Configure Label Distribution Protocol IGP Auto-configuration for OSPF

This task explains how to configure LDP auto-configuration for an OSPF instance.

Optionally, you can configure this feature for an area of an OSPF instance.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*

3. **mpls ldp auto-config**
4. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure**

Example:

```
RP/0/RP0/CPU0:router# configure
Enters mode.
```

Step 2 **router ospf** *process-name*

Example:

```
RP/0/RP0/CPU0:router(config)# router ospf 1
Enables OSPF routing for the specified routing process and places the router in router configuration mode.
```

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **mpls ldp auto-config**

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# mpls ldp auto-config
Enables LDP IGP interface auto-configuration for an OSPF instance.
```

- Optionally, this command can be configured for an area of an OSPF instance.

Step 4 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configure LDP IGP Synchronization: OSPF

Perform this task to configure LDP IGP Synchronization under OSPF.



Note By default, there is no synchronization between LDP and IGPs.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. (Optional) **vrf** *vrf-name*
4. Use one of the following commands:
 - **mpls ldp sync**
 - **area** *area-id* **mpls ldp sync**
 - **area** *area-id* **interface** *name* **mpls ldp sync**
5. (Optional) Use one of the following commands:
 - **mpls ldp sync**
 - **area** *area-id* **mpls ldp sync**
 - **area** *area-id* **interface** *name* **mpls ldp sync**
6. Use the **commit** or **end** command.
7. (Optional) **show mpls ldp vrf** *vrf-name* **ipv4 igp sync**
8. (Optional) **show mpls ldp vrf all ipv4 igp sync**
9. (Optional) **show mpls ldp { ipv4 | ipv6 } igp sync**

DETAILED STEPS**Step 1** **configure****Example:**

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 2 **router ospf** *process-name***Example:**

```
RP/0/RP0/CPU0:router(config)# router ospf 100
```

Identifies the OSPF routing process and enters OSPF configuration mode.

Step 3 (Optional) **vrf** *vrf-name***Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# vrf red
```

Specifies the non-default VRF.

Step 4 Use one of the following commands:

- **mpls ldp sync**
- **area** *area-id* **mpls ldp sync**
- **area** *area-id* **interface** *name* **mpls ldp sync**

Example:

```
RP/0/RP0/CPU0:router(config-ospf)# mpls ldp sync
```

Enables LDP IGP synchronization on an interface.

Step 5 (Optional) Use one of the following commands:

- **mpls ldp sync**
- **area *area-id* mpls ldp sync**
- **area *area-id* interface *name* mpls ldp sync**

Example:

```
RP/0/RP0/CPU0:router(config-ospf-vrf)# mpls ldp sync
```

```
RP/0/RP0/CPU0:router(config-ospf-vrf)# area 1 mpls ldp sync
```

Enables LDP IGP synchronization on an interface for the specified VRF.

Step 6 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Step 7 (Optional) **show mpls ldp vrf *vrf-name* ipv4 igp sync**

Example:

```
RP/0/RP0/CPU0:router# show mpls ldp vrf red ipv4 igp sync
```

Displays the LDP IGP synchronization information for the specified VRF for address family IPv4.

Step 8 (Optional) **show mpls ldp vrf all ipv4 igp sync**

Example:

```
RP/0/RP0/CPU0:router# show mpls ldp vrf all ipv4 igp sync
```

Displays the LDP IGP synchronization information for all VRFs for address family IPv4.

Step 9 (Optional) **show mpls ldp { ipv4 | ipv6 } igp sync**

Example:

```
RP/0/RP0/CPU0:router# show mpls ldp ipv4 igp sync
```

```
RP/0/RP0/CPU0:router# show mpls ldp ipv6 igp sync
```

Displays the LDP IGP synchronization information for IPv4 or IPv6 address families.

Example

The example shows how to configure LDP IGP synchronization for OSPF.

```
router ospf 100
mpls ldp sync
!
mpls ldp
  igp sync delay 30
!
```

IGP link state

IGP Link-State Database Distribution

A given BGP node may have connections to multiple, independent routing domains. IGP link-state database distribution into BGP-LS is supported for both OSPF and IS-IS protocols in order to distribute this information on to controllers or applications that desire to build paths spanning or including these multiple domains.

To distribute OSPFv2 link-state data using BGP-LS, use the **distribute link-state** command in router configuration mode.

```
Router# configure
Router(config)# router ospf 100
Router(config-ospf)# distribute link-state instance-id 32
```

OSPF Authentication Message Digest Management

All OSPF routing protocol exchanges are authenticated and the method used can vary depending on how authentication is configured. When using cryptographic authentication, the OSPF routing protocol uses the Message Digest 5 (MD5) authentication algorithm to authenticate packets transmitted between neighbors in the network. For each OSPF protocol packet, a key is used to generate and verify a message digest that is appended to the end of the OSPF packet. The message digest is a one-way function of the OSPF protocol packet and the secret key. Each key is identified by the combination of interface used and the key identification. An interface may have multiple keys active at any time.

To manage the rollover of keys and enhance MD5 authentication for OSPF, you can configure a container of keys called a *keychain* with each key comprising the following attributes: generate/accept time, key identification, and authentication algorithm.

Configure Authentication Message Digest Management for OSPF

This task explains how to manage authentication of a keychain on the OSPF interface.

Before you begin

A valid keychain must be configured before this task can be attempted.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **router-id** { *router-id* }
4. **area** *area-id*
5. **interface** *type interface-path-id*
6. **authentication** [**message-digest** *keychain* | **null**]
7. Use the **commit** or **end** command.

DETAILED STEPS

Step 1 **configure****Example:**

```
RP/0/RP0/CPU0:router# configure
```

Enters mode.

Step 2 **router ospf** *process-name***Example:**

```
RP/0/RP0/CPU0:router(config)# router ospf 1
```

Enables OSPF routing for the specified routing process and places the router in router configuration mode.

Note The *process-name* argument is any alphanumeric string no longer than 40 characters.

Step 3 **router-id** { *router-id* }**Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# router id 192.168.4.3
```

Configures a router ID for the OSPF process.

Note We recommend using a stable IPv4 address as the router ID.

Step 4 **area** *area-id***Example:**

```
RP/0/RP0/CPU0:router(config-ospf)# area 1
```

Enters area configuration mode.

The *area-id* argument can be entered in dotted-decimal or IPv4 address notation, such as area 1000 or area 0.0.3.232. However, you must choose one form or the other for an area. We recommend using the IPv4 address notation.

Step 5 **interface** *type interface-path-id*

Example:

```
RP/0/RP0/CPU0:router(config-ospf-ar)# interface TenGigE 0/0/0/0
```

Enters interface configuration mode and associates one or more interfaces to the area.

Step 6 authentication [message-digest keychain | null]

Configures an MD5 keychain.

Example:

The following example shows the configuration for message-digest authentication.

```
RP/0/RP0/CPU0:router(config-ospf-ar-if)# authentication message-digest keychain ospf_int1
```

Note In the above example, the *ospf_int1* keychain must be configured before you attempt this step.

Step 7 Use the **commit** or **end** command.

commit —Saves the configuration changes and remains within the configuration session.

end —Prompts user to take one of these actions:

- **Yes** — Saves configuration changes and exits the configuration session.
- **No** —Exits the configuration session without committing the configuration changes.
- **Cancel** —Remains in the configuration session, without committing the configuration changes.

Examples

The following example shows how to configure the keychain *ospf_intf_1* that contains five key IDs. Each key ID is configured with different **send-lifetime** values; however, all key IDs specify the same text string for the key.

```
key chain ospf_intf_1
key 1
send-lifetime 11:30:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
key 2
send-lifetime 11:40:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
key 3
send-lifetime 11:50:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
key 4
send-lifetime 12:00:30 May 1 2007 duration 600
cryptographic-algorithm MD5
key-string clear ospf_intf_1
key 5
send-lifetime 12:10:30 May 1 2007 duration 600
cryptographic-algorithm MD5
```

```
key-string clear ospf_intf_1
```

The following example shows that keychain authentication is enabled on the TenGigE 0/0/0/0 interface:

```
show ospf 1 interface TenGigE 0/0/0/0
```

```
TenGigE 0/0/0/0 is up, line protocol is up
  Internet Address 100.10.10.2/24, Area 0
  Process ID 1, Router ID 2.2.2.1, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State DR, Priority 1
  Designated Router (ID) 2.2.2.1, Interface address 100.10.10.2
  Backup Designated router (ID) 1.1.1.1, Interface address 100.10.10.1
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:02
  Index 3/3, flood queue length 0
  Next 0(0)/0(0)
  Last flood scan length is 2, maximum is 16
  Last flood scan time is 0 msec, maximum is 0 msec
  Neighbor Count is 1, Adjacent neighbor count is 1
    Adjacent with neighbor 1.1.1.1 (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
Keychain-based authentication enabled
  Key id used is 3
  Multi-area interface Count is 0
```

The following example shows output for configured keys that are active:

```
show key chain ospf_intf_1
```

```
Key-chain: ospf_intf_1/ -

Key 1 -- text "0700325C4836100B0314345D"
  cryptographic-algorithm -- MD5
  Send lifetime: 11:30:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
Key 2 -- text "10411A0903281B051802157A"
  cryptographic-algorithm -- MD5
  Send lifetime: 11:40:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
Key 3 -- text "06091C314A71001711112D5A"
  cryptographic-algorithm -- MD5
  Send lifetime: 11:50:30, 01 May 2007 - (Duration) 600 [Valid now]
  Accept lifetime: Not configured
Key 4 -- text "151D181C0215222A3C350A73"
  cryptographic-algorithm -- MD5
  Send lifetime: 12:00:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
Key 5 -- text "151D181C0215222A3C350A73"
  cryptographic-algorithm -- MD5
  Send lifetime: 12:10:30, 01 May 2007 - (Duration) 600
  Accept lifetime: Not configured
```

OSPF Link-State Database Overload Protection

The OSPF Link-State Database Overload Protection feature allows you to protect the OSPF routing process from the large number of received link-state advertisements (LSAs) that can result from a misconfiguration on another router in the OSPF domain (for example, the redistribution of a large number of IP prefixes to OSPF). In this feature, the router monitors the count of all nonself-generated LSAs it receives. When the total number of LSAs reaches the value set as a threshold in the configuration, the router logs a warning message. When the total LSAs exceed the max number in configuration, the router stops accepting new LSAs. Therefore, the OSPF Link-State Database Overload Protection feature allows you to limit the number of nonself-generated LSAs for a given OSPF process. Excessive LSAs generated by other routers in the OSPF domain can substantially drain the CPU and memory resources of the router.

If the count of received LSAs is higher than the configured max number after one minute, the OSPF process disables all adjacencies in the given context and clears the OSPF database. This state is called the ignore state. In this state, the router ignores all OSPF packets received on all interfaces belonging to the OSPF instance and stops OSPF packet generation on its interfaces. The OSPF process remains in ignore state for the duration of the configured ignore time. When the ignore time expires, the OSPF process returns to a normal operation state and starts building adjacencies on all its interfaces.

To prevent the OSPF instance from endlessly oscillating between the normal state and ignore state, as a result of the LSA counts immediately exceeding the max number again after it returns from the ignore state, the OSPF instance keeps a count of how many times it has been in ignore state. This counter is called the ignore count. If the ignore count exceeds its configured value, the OSPF instance remains in ignore state permanently.

To return the OSPF instance to its normal state, use the **clear IP OSPF process** command. The ignore count is reset to zero if the LSA count doesn't exceed the max number again during the time configured by the reset-time keyword.



Note There are no default values for the **max-lsa** command. The router considers the max-lsa limits only if it's available in the configuration. After configuring the OSPF Link-State Database Overload Protection feature, the default values for other OSPF parameters are as follows:

- Default threshold percentage to log warning: 75%.
 - Default ignore count value: 5
 - Default ignore time in minutes: 5 minutes
 - Default time to reset ignore count: 10 minutes
-

In the event of LSA count reaches or exceeds the configured threshold value, router displays the following logs:

```
Router:Oct 25 11:42:49.756 IST: ospf[1046]: %ROUTING-OSPF-4-MAX_LSA_THR : Threshold for
maximum number of non self-generated LSAs has been reached "default" - 9000 LSAs
Router:Oct 25 11:42:49.756 IST: ospf[1046]: %ROUTING-OSPF-4-MAX_LSA : Maximum number of non
self-generated LSAs has been exceeded "default" - 12001 LSAs
```

```
Router:Oct 25 11:42:49.756 IST: ospf[1046]: %ROUTING-OSPF-4-MAX_LSA_THR : Threshold for
maximum number of non self-generated LSAs has been reached "V1" - 750 LSAs
Router:Oct 25 11:42:49.756 IST: ospf[1046]: %ROUTING-OSPF-4-MAX_LSA : Maximum number of non
self-generated LSAs has been exceeded "V1" - 1001 LSA
```

Configuration Example

The following example shows how to configure the OSPF instance to accept 12000 nonself-generated LSAs in the global routing table and 1000 nonself-generated LSAs in VRF V1:

```
Router# configure
Router(config)# router ospf 0
Router(config-ospf)# max-lsa 12000
Router(config-ospf)# vrf V1
Router(config-ospf)# max-lsa 1000
```

Running Configuration

The following example shows how to display the current status of the OSPF instance:

```
Router# show ospf 0
Routing Process "ospf 0" with ID 10.0.0.2
NSR (Non-stop routing) is Disabled
Supports only single TOS(TOS0) routes
Supports opaque LSA
It is an area border router
Maximum number of non self-generated LSA allowed 12000
Current number of non self-generated LSA 1
Threshold for warning message 75%
Ignore-time 5 minutes, reset-time 10 minutes
Ignore-count allowed 5, current ignore-count 0

Router# show ospf vrf V1
VRF V1 active in Routing Process "ospf 0" with ID 10.0.0.2
Role: Primary Active
NSR (Non-stop routing) is Enabled
Supports only single TOS(TOS0) routes
Supports opaque LSA
It is an area border router
Maximum number of non self-generated LSA allowed 1000
Current number of non self-generated LSA 1
Threshold for warning message 75%
Ignore-time 5 minutes, reset-time 10 minutes
Ignore-count allowed 5, current ignore-count 0
Router is not originating router-LSAs with maximum metric
```

References for OSPF

To implement OSPF you need to understand the following concepts:

OSPF Functional Overview

OSPF is a routing protocol for IP. It is a link-state protocol, as opposed to a distance-vector protocol. A link-state protocol makes its routing decisions based on the states of the links that connect source and destination machines. The state of the link is a description of that interface and its relationship to its neighboring networking devices. The interface information includes the IP address of the interface, network mask, type of network to which it is connected, routers connected to that network, and so on. This information is propagated in various types of link-state advertisements (LSAs).

A router stores the collection of received LSA data in a link-state database. This database includes LSA data for the links of the router. The contents of the database, when subjected to the Dijkstra algorithm, extract data to create an OSPF routing table. The difference between the database and the routing table is that the database

contains a complete collection of raw data; the routing table contains a list of shortest paths to known destinations through specific router interface ports.

OSPF is the IGP of choice because it scales to large networks. It uses areas to partition the network into more manageable sizes and to introduce hierarchy in the network. A router is attached to one or more areas in a network. All of the networking devices in an area maintain the same complete database information about the link states in their area only. They do not know about all link states in the network. The agreement of the database information among the routers in the area is called convergence.

At the intradomain level, OSPF can import routes learned using Intermediate System-to-Intermediate System (IS-IS). OSPF routes can also be exported into IS-IS. At the interdomain level, OSPF can import routes learned using Border Gateway Protocol (BGP). OSPF routes can be exported into BGP.

Unlike Routing Information Protocol (RIP), OSPF does not provide periodic routing updates. On becoming neighbors, OSPF routers establish an adjacency by exchanging and synchronizing their databases. After that, only changed routing information is propagated. Every router in an area advertises the costs and states of its links, sending this information in an LSA. This state information is sent to all OSPF neighbors one hop away. All the OSPF neighbors, in turn, send the state information unchanged. This flooding process continues until all devices in the area have the same link-state database.

To determine the best route to a destination, the software sums all of the costs of the links in a route to a destination. After each router has received routing information from the other networking devices, it runs the shortest path first (SPF) algorithm to calculate the best path to each destination network in the database.

The networking devices running OSPF detect topological changes in the network, flood link-state updates to neighbors, and quickly converge on a new view of the topology. Each OSPF router in the network soon has the same topological view again. OSPF allows multiple equal-cost paths to the same destination. Since all link-state information is flooded and used in the SPF calculation, multiple equal cost paths can be computed and used for routing.

On broadcast and nonbroadcast multiaccess (NBMA) networks, the designated router (DR) or backup DR performs the LSA flooding.

OSPF runs directly on top of IP; it does not use TCP or User Datagram Protocol (UDP). OSPF performs its own error correction by means of checksums in its packet header and LSAs.

In OSPFv3, the fundamental concepts are the same as OSPF Version 2, except that support is added for the increased address size of IPv6. New LSA types are created to carry IPv6 addresses and prefixes, and the protocol runs on an individual link basis rather than on an individual IP-subnet basis.

OSPF typically requires coordination among many internal routers: Area Border Routers (ABRs), which are routers attached to multiple areas, and Autonomous System Border Routers (ASBRs) that export reroutes from other sources (for example, IS-IS, BGP, or static routes) into the OSPF topology. At a minimum, OSPF-based routers or access servers can be configured with all default parameter values, no authentication, and interfaces assigned to areas. If you intend to customize your environment, you must ensure coordinated configurations of all routers.

Comparison of Cisco IOS XR Software OSPFv3 and OSPFv2

Much of the OSPFv3 protocol is the same as in OSPFv2. OSPFv3 is described in RFC 2740.

The key differences between the Cisco IOS XR Software OSPFv3 and OSPFv2 protocols are as follows:

- OSPFv3 expands on OSPFv2 to provide support for IPv6 routing prefixes and the larger size of IPv6 addresses.

- When using an NBMA interface in OSPFv3, users must manually configure the router with the list of neighbors. Neighboring routers are identified by the link local address of the attached interface of the neighbor.
- Unlike in OSPFv2, multiple OSPFv3 processes can be run on a link.
- LSAs in OSPFv3 are expressed as “prefix and prefix length” instead of “address and mask.”
- The router ID is a 32-bit number with no relationship to an IPv6 address.

OSPF Hierarchical CLI and CLI Inheritance

Hierarchical CLI is the grouping of related network component information at defined hierarchical levels such as at the router, area, and interface levels. Hierarchical CLI allows for easier configuration, maintenance, and troubleshooting of OSPF configurations. When configuration commands are displayed together in their hierarchical context, visual inspections are simplified. Hierarchical CLI is intrinsic for CLI inheritance to be supported.

With CLI inheritance support, you need not explicitly configure a parameter for an area or interface. In the software, the parameters of interfaces in the same area can be exclusively configured with a single command, or parameter values can be inherited from a higher hierarchical level—such as from the area configuration level or the router ospf configuration levels.

For example, the hello interval value for an interface is determined by this precedence “IF” statement:

If the **hello interval** command is configured at the interface configuration level, then use the interface configured value, else

If the **hello interval** command is configured at the area configuration level, then use the area configured value, else

If the **hello interval** command is configured at the router ospf configuration level, then use the router ospf configured value, else

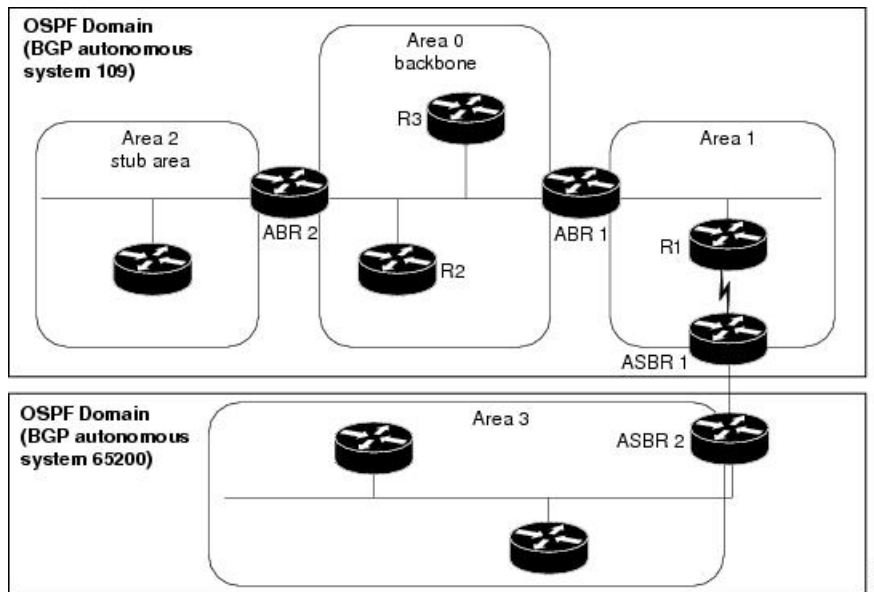
Use the default value of the command.

OSPF Routing Components

Before implementing OSPF, you must know what the routing components are and what purpose they serve. They consist of the autonomous system, area types, interior routers, ABRs, and ASBRs.

Figure 4: OSPF Routing Components

This figure illustrates the routing components in an OSPF network topology.



Autonomous Systems

The autonomous system is a collection of networks, under the same administrative control, that share routing information with each other. An autonomous system is also referred to as a routing domain. *Figure 1: OSPF Routing Components* shows two autonomous systems: 109 and 65200. An autonomous system can consist of one or more OSPF areas.

Areas

Areas allow the subdivision of an autonomous system into smaller, more manageable networks or sets of adjacent networks. As shown in the *Figure 1: OSPF Routing Components*, autonomous system 109 consists of three areas: Area 0, Area 1, and Area 2.

OSPF hides the topology of an area from the rest of the autonomous system. The network topology for an area is visible only to routers inside that area. When OSPF routing is within an area, it is called *intra-area routing*. This routing limits the amount of link-state information flood into the network, reducing routing traffic. It also reduces the size of the topology information in each router, conserving processing and memory requirements in each router.

Also, the routers within an area cannot see the detailed network topology outside the area. Because of this restricted view of topological information, you can control traffic flow between areas and reduce routing traffic when the entire autonomous system is a single routing domain.

Backbone Area

A backbone area is responsible for distributing routing information between multiple areas of an autonomous system. OSPF routing occurring outside of an area is called *interarea routing*.

The backbone itself has all properties of an area. It consists of ABRs, routers, and networks only on the backbone. As shown in *Figure 1: OSPF Routing Components*, Area 0 is an OSPF backbone area. Any OSPF backbone area has a reserved area ID of 0.0.0.0.

Routers

The OSPF network is composed of ABRs, ASBRs, and interior routers.

Area Border Routers

An area border routers (ABR) is a router with multiple interfaces that connect directly to networks in two or more areas. An ABR runs a separate copy of the OSPF algorithm and maintains separate routing data for each area that is attached to, including the backbone area. ABRs also send configuration summaries for their attached areas to the backbone area, which then distributes this information to other OSPF areas in the autonomous system. In *Figure 1: OSPF Routing Components* section, there are two ABRs. ABR 1 interfaces Area 1 to the backbone area. ABR 2 interfaces the backbone Area 0 to Area 2, a stub area.

Autonomous System Boundary Routers (ASBR)

An autonomous system boundary router (ASBR) provides connectivity from one autonomous system to another system. ASBRs exchange their autonomous system routing information with boundary routers in other autonomous systems. Every router inside an autonomous system knows how to reach the boundary routers for its autonomous system.

ASBRs can import external routing information from other protocols like BGP and redistribute them as AS-external (ASE) Type 5 LSAs to the OSPF network. If the Cisco IOS XR router is an ASBR, you can configure it to advertise VIP addresses for content as autonomous system external routes. In this way, ASBRs flood information about external networks to routers within the OSPF network.

ASBR routes can be advertised as a Type 1 or Type 2 ASE. The difference between Type 1 and Type 2 is how the cost is calculated. For a Type 2 ASE, only the external cost (metric) is considered when multiple paths to the same destination are compared. For a Type 1 ASE, the combination of the external cost and cost to reach the ASBR is used. Type 2 external cost is the default and is always more costly than an OSPF route and used only if no OSPF route exists.

Interior Routers

An interior router (such as R1 in *Figure 1: OSPF Routing Components*) is attached to one area (for example, all the interfaces reside in the same area).

OSPF Process and Router ID

An OSPF process is a logical routing entity running OSPF in a physical router. This logical routing entity should not be confused with the logical routing feature that allows a system administrator to partition the physical box into separate routers.

A physical router can run multiple OSPF processes, although the only reason to do so would be to connect two or more OSPF domains. Each process has its own link-state database. The routes in the routing table are calculated from the link-state database. One OSPF process does not share routes with another OSPF process unless the routes are redistributed.

Each OSPF process is identified by a router ID. The router ID must be unique across the entire routing domain. OSPF obtains a router ID from the following sources, in order of decreasing preference:

- By default, when the OSPF process initializes, it checks if there is a router-id in the checkpointing database.

- The 32-bit numeric value specified by the OSPF router-id command in router configuration mode. (This value can be any 32-bit value. It is not restricted to the IPv4 addresses assigned to interfaces on this router, and need not be a routable IPv4 address.)
- The ITAL selected router-id.
- The primary IPv4 address of an interface over which this OSPF process is running. The first interface address in the OSPF interface is selected.

We recommend that the router ID be set by the **router-id** command in router configuration mode. Separate OSPF processes could share the same router ID, in which case they cannot reside in the same OSPF routing domain.

Supported OSPF Network Types

OSPF classifies different media into the following types of networks:

- NBMA networks
- Broadcast networks

You can configure your network as either a broadcast or an NBMA network. Using this feature, you can configure broadcast networks as NBMA networks when, for example, you have routers in your network that do not support multicast addressing.

Route Authentication Methods for OSPF

OSPF Version 2 supports two types of authentication: plain text authentication and MD5 authentication. By default, no authentication is enabled (referred to as null authentication in RFC 2178).

OSPF Version 3 supports all types of authentication except key rollover.

Plain Text Authentication

Plain text authentication (also known as Type 1 authentication) uses a password that travels on the physical medium and is easily visible to someone that does not have access permission and could use the password to infiltrate a network. Therefore, plain text authentication does not provide security. It might protect against a faulty implementation of OSPF or a misconfigured OSPF interface trying to send erroneous OSPF packets.

MD5 Authentication

MD5 authentication provides a means of security. No password travels on the physical medium. Instead, the router uses MD5 to produce a message digest of the OSPF packet plus the key, which is sent on the physical medium. Using MD5 authentication prevents a router from accepting unauthorized or deliberately malicious routing updates, which could compromise your network security by diverting your traffic.



Note MD5 authentication supports multiple keys, requiring that a key number be associated with a key. See the *OSPF Authentication Message Digest Management* section.

Key Rollover

To support the changing of an MD5 key in an operational network without disrupting OSPF adjacencies (and hence the topology), a key rollover mechanism is supported. As a network administrator configures the new key into the multiple networking devices that communicate, some time exists when different devices are using both a new key and an old key. If an interface is configured with a new key, the software sends two copies of the same packet, each authenticated by the old key and new key. The software tracks which devices start using the new key, and the software stops sending duplicate packets after it detects that all of its neighbors are using the new key. The software then discards the old key. The network administrator must then remove the old key from each the configuration file of each router.

OSPF FIB Download Notification

OSPF FIB Download Notification feature minimizes the ingress traffic drop for a prolonged period of time after the line card reloads.

Open Shortest Path First (OSPF) registers with Routing Information Base (RIB) through ITAL which keeps the interface down until all the routes are downloaded to Forwarding Information Base (FIB). OSPF gets the Interface Up notification when all the routes on the reloaded line card are downloaded through RIB/FIB.

RIB provides notification to registered clients when a:

- Node is lost.
- Node is created.
- Node's FIB upload is completed.

Designated Router (DR) for OSPF

On broadcast or NBMA segments only, OSPF minimizes the amount of information being exchanged on a segment by choosing one router to be a DR and one router to be a BDR. Thus, the routers on the segment have a central point of contact for information exchange. Instead of each router exchanging routing updates with every other router on the segment, each router exchanges information with the DR and BDR. The DR and BDR relay the information to the other routers.

The software looks at the priority of the routers on the segment to determine which routers are the DR and BDR. The router with the highest priority is elected the DR. If there is a tie, then the router with the higher router ID takes precedence. After the DR is elected, the BDR is elected the same way. A router with a router priority set to zero is ineligible to become the DR or BDR.

Default Route for OSPF

Type 5 (ASE) LSAs are generated and flooded to all areas except stub areas. For the routers in a stub area to be able to route packets to destinations outside the stub area, a default route is injected by the ABR attached to the stub area.

The cost of the default route is 1 (default) or is determined by the value specified in the **default-cost** command.

Link-State Advertisement Types for OSPF Version 2

Each of the following LSA types has a different purpose:

- Router LSA (Type 1)—Describes the links that the router has within a single area, and the cost of each link. These LSAs are flooded within an area only. The LSA indicates if the router can compute paths based on quality of service (QoS), whether it is an ABR or ASBR, and if it is one end of a virtual link. Type 1 LSAs are also used to advertise stub networks.
- Network LSA (Type 2)—Describes the link state and cost information for all routers attached to a multiaccess network segment. This LSA lists all the routers that have interfaces attached to the network segment. It is the job of the designated router of a network segment to generate and track the contents of this LSA.
- Summary LSA for ABRs (Type 3)—Advertises internal networks to routers in other areas (interarea routes). Type 3 LSAs may represent a single network or a set of networks aggregated into one prefix. Only ABRs generate summary LSAs.
- Summary LSA for ASBRs (Type 4)—Advertises an ASBR and the cost to reach it. Routers that are trying to reach an external network use these advertisements to determine the best path to the next hop. ABRs generate Type 4 LSAs.
- Autonomous system external LSA (Type 5)—Redistributes routes from another autonomous system, usually from a different routing protocol into OSPF.
- Autonomous system external LSA (Type 7)—Provides for carrying external route information within an NSSA. Type 7 LSAs may be originated by and advertised throughout an NSSA. NSSAs do not receive or originate Type 5 LSAs. Type 7 LSAs are advertised only within a single NSSA. They are not flooded into the backbone area or into any other area by border routers.
- Intra-area-prefix LSAs (Type 9)—A router can originate multiple intra-area-prefix LSAs for every router or transit network, each with a unique link-state ID. The link-state ID for each intra-area-prefix LSA describes its association to either the router LSA or network LSA and contains prefixes for stub and transit networks.
- Area local scope (Type 10)—Opaque LSAs are not flooded past the borders of their associated area.
- Link-state (Type 11)—The LSA is flooded throughout the AS. The flooding scope of Type 11 LSAs are equivalent to the flooding scope of AS-external (Type 5) LSAs. Similar to Type 5 LSAs, the LSA is rejected if a Type 11 opaque LSA is received in a stub area from a neighboring router within the stub area. Type 11 opaque LSAs have these attributes:
 - LSAs are flooded throughout all transit areas.
 - LSAs are not flooded into stub areas from the backbone.
 - LSAs are not originated by routers into their connected stub areas.

Link-State Advertisement Types for OSPFv3

Each of the following LSA types has a different purpose:

- Router LSA (Type 1)—Describes the link state and costs of a the router link to the area. These LSAs are flooded within an area only. The LSA indicates whether the router is an ABR or ASBR and if it is one end of a virtual link. Type 1 LSAs are also used to advertise stub networks. In OSPFv3, these LSAs have no address information and are network protocol independent. In OSPFv3, router interface information may be spread across multiple router LSAs. Receivers must concatenate all router LSAs originated by a given router before running the SPF calculation.

- Network LSA (Type 2)—Describes the link state and cost information for all routers attached to a multiaccess network segment. This LSA lists all OSPF routers that have interfaces attached to the network segment. Only the elected designated router for the network segment can generate and track the network LSA for the segment. In OSPFv3, network LSAs have no address information and are network-protocol-independent.
- Interarea-prefix LSA for ABRs (Type 3)—Advertises internal networks to routers in other areas (interarea routes). Type 3 LSAs may represent a single network or set of networks aggregated into one prefix. Only ABRs generate Type 3 LSAs. In OSPFv3, addresses for these LSAs are expressed as “prefix and prefix length” instead of “address and mask.” The default route is expressed as a prefix with length 0.
- Interarea-router LSA for ASBRs (Type 4)—Advertises an ASBR and the cost to reach it. Routers that are trying to reach an external network use these advertisements to determine the best path to the next hop. ABRs generate Type 4 LSAs.
- Autonomous system external LSA (Type 5)—Redistributes routes from another autonomous system, usually from a different routing protocol into OSPF. In OSPFv3, addresses for these LSAs are expressed as “prefix and prefix length” instead of “address and mask.” The default route is expressed as a prefix with length 0.
- Autonomous system external LSA (Type 7)—Provides for carrying external route information within an NSSA. Type 7 LSAs may be originated by and advertised throughout an NSSA. NSSAs do not receive or originate Type 5 LSAs. Type 7 LSAs are advertised only within a single NSSA. They are not flooded into the backbone area or into any other area by border routers.
- Link LSA (Type 8)—Has link-local flooding scope and is never flooded beyond the link with which it is associated. Link LSAs provide the link-local address of the router to all other routers attached to the link or network segment, inform other routers attached to the link of a list of IPv6 prefixes to associate with the link, and allow the router to assert a collection of Options bits to associate with the network LSA that is originated for the link.
- Intra-area-prefix LSAs (Type 9)—A router can originate multiple intra-area-prefix LSAs for every router or transit network, each with a unique link-state ID. The link-state ID for each intra-area-prefix LSA describes its association to either the router LSA or network LSA and contains prefixes for stub and transit networks.

An address prefix occurs in almost all newly defined LSAs. The prefix is represented by three fields: Prefix Length, Prefix Options, and Address Prefix. In OSPFv3, addresses for these LSAs are expressed as “prefix and prefix length” instead of “address and mask.” The default route is expressed as a prefix with length 0.

Inter-area-prefix and intra-area-prefix LSAs carry all IPv6 prefix information that, in IPv4, is included in router LSAs and network LSAs. The Options field in certain LSAs (router LSAs, network LSAs, interarea-router LSAs, and link LSAs) has been expanded to 24 bits to provide support for OSPF in IPv6.

In OSPFv3, the sole function of link-state ID in interarea-prefix LSAs, interarea-router LSAs, and autonomous system external LSAs is to identify individual pieces of the link-state database. All addresses or router IDs that are expressed by the link-state ID in OSPF Version 2 are carried in the body of the LSA in OSPFv3.

Passive Interface

Setting an interface as passive disables the sending of routing updates for the neighbors, hence adjacencies will not be formed in OSPF. However, the particular subnet will continue to be advertised to OSPF neighbors. Use the **passive** command in appropriate mode to suppress the sending of OSPF protocol operation on an interface.

It is recommended to use passive configuration on interfaces that are connecting LAN segments with hosts to the rest of the network, but are not meant to be transit links between routers.

Modes of Graceful Restart Operation

The operational modes that a router can be in for this feature are restart mode and helper mode, helper mode, and protocol shutdown mode. Restart mode occurs when the OSPFv3 process is doing a graceful restart. Helper mode refers to the neighbor routers that continue to forward traffic on established OSPFv3 routes while OSPFv3 is restarting on a neighboring router.

Restart Mode

When the OSPFv3 process starts up, it determines whether it must attempt a graceful restart. The determination is based on whether graceful restart was previously enabled. (OSPFv3 does not attempt a graceful restart upon the first-time startup of the router.) When OSPFv3 graceful restart is enabled, it changes the purge timer in the RIB to a nonzero value.

During a graceful restart, the router does not populate OSPFv3 routes in the RIB. It tries to bring up full adjacencies with the fully adjacent neighbors that OSPFv3 had before the restart. Eventually, the OSPFv3 process indicates to the RIB that it has converged, either for the purpose of terminating the graceful restart (for any reason) or because it has completed the graceful restart.

If OSPFv3 attempts a restart too soon after the most recent restart, the OSPFv3 process is most likely crashing repeatedly, so the new graceful restart stops running. To control the period between allowable graceful restarts, use the **graceful-restart interval** command. When OSPFv3 starts a graceful restart with the first interface that comes up, a timer starts running to limit the duration (or lifetime) of the graceful restart. You can configure this period with the **graceful-restart lifetime** command. On each interface that comes up, a *grace* LSA (Type 11) is flooded to indicate to the neighboring routers that this router is attempting graceful restart. The neighbors enter into helper mode. The designated router and backup designated router check of the hello packet received from the restarting neighbor is bypassed, because it might not be valid.

Helper Mode

Helper mode is enabled by default. When a (helper) router receives a grace LSA (Type 11) from a router that is attempting a graceful restart, the following events occur:

- If helper mode has been disabled through the **graceful-restart helper disable** command, the router drops the LSA packet.
- If helper mode is enabled, the router enters helper mode if all of the following conditions are met:
 - The local router itself is not attempting a graceful restart.
 - The local (helping) router has full adjacency with the sending neighbor.
 - The value of *lsage* (link state age) in the received LSA is less than the requested grace period.
 - The sender of the grace LSA is the same as the originator of the grace LSA.
- Upon entering helper mode, a router performs its helper function for a specific period of time. This time period is the lifetime value from the router that is in restart mode—minus the value of *lsage* in the received grace LSA. If the graceful restart succeeds in time, the helper's timer is stopped before it expires. If the helper's timer does expire, the adjacency to the restarting router is brought down, and normal OSPFv3 functionality resumes.

- The dead timer is not honored by the router that is in helper mode.
- A router in helper mode ceases to perform the helper function in any of the following cases:
 - The helper router is able to bring up a FULL adjacency with the restarting router.
 - The local timer for the helper function expires.

Protocol Shutdown Mode

In this mode the OSPFv3 operation is completely disabled. This is accomplished by flushing self-originated link state advertisements (LSAs), immediately bringing down local OSPFv3-supported interfaces, and clearing the Link State Database (LSDB). The non-local LSDB entries are removed by OSPFv3, These are not flooded (MaxAged).

The protocol shutdown mode can be invoked either manually through the **protocol shutdown** command that disables the protocol instance or when the OSPFv3 process runs out of memory. These events occur when protocol shut down is performed:

- The local Router LSA and all local Link LSAs are flushed. All other LSAs are eventually aged out by other OSPFv3 routers in the domain.
- OSPFv3 neighbors not yet in Full state with the local router are brought down with the Kill_Nbr event.
- After a three second delay, empty Hello packets are immediately sent to each neighbor that has an active adjacency.
 - An empty Hello packet is sent periodically until the dead_interval has elapsed.
 - When the dead_interval elapses, Hello packets are no longer sent.

After a Dead Hello interval delay (4 X Hello Interval), the following events are then performed:

- The LSA database from that OSPFv3 instance is cleared.
- All routes from RIB that were installed by OSPFv3 are purged.

The router will not respond to any OSPF control packets it receives from neighbors while in protocol shutdown state.

Protocol Restoration

The method of restoring the protocol is dependent on the trigger that originally invoked the shut down. If the OSPFv3 was shut down using the **protocol shutdown** command, then use the **no protocol shutdown** command to restore OSPFv3 back to normal operation. If the OSPFv3 was shutdown due to a Critical Memory message from the sysmon, then a Normal Memory message from sysmon, which indicates that sufficient memory has been restored to the processor, restores the OSPFv3 protocol to resume normal operation. When OSPFv3 is shutdown due to the Critical Memory trigger, it must be manually restarted when normal memory levels are restored on the route processor. It will not automatically restore itself.

These events occur when the OSPFv3 is restored:

1. All OSPFv3 interfaces are brought back up using the Hello packets and database exchange.
2. The local router and link LSAs are rebuilt and advertised.
3. The router replies normally to all OSPFv3 control messages received from neighbors.

4. Routes learned from other OSPFv3 routers are installed in RIB.

Load Balancing in OSPF Version 2 and OSPFv3

When a router learns multiple routes to a specific network by using multiple routing processes (or routing protocols), it installs the route with the lowest administrative distance in the routing table. Sometimes the router must select a route from among many learned by using the same routing process with the same administrative distance. In this case, the router chooses the path with the lowest cost (or metric) to the destination. Each routing process calculates its cost differently; the costs may need to be manipulated to achieve load balancing.

OSPF performs load balancing automatically. If OSPF finds that it can reach a destination through more than one interface and each path has the same cost, it installs each path in the routing table. The only restriction on the number of paths to the same destination is controlled by the **maximum-paths** (OSPF) command.

The range for maximum paths is from 1 to 8 and the default number of maximum paths is 8.

Path Computation Element for OSPFv2

A PCE is an entity (component, application, or network node) that is capable of computing a network path or route based on a network graph and applying computational constraints.

PCE is accomplished when a PCE address and client is configured for MPLS-TE. PCE communicates its PCE address and capabilities to OSPF then OSPF packages this information in the PCE Discovery type-length-value (TLV) (Type 2) and reoriginates the RI LSA. OSPF also includes the Router Capabilities TLV (Type 1) in all its RI LSAs. The PCE Discovery TLV contains the PCE address sub-TLV (Type 1) and the Path Scope Sub-TLV (Type 2).

The PCE Address Sub-TLV specifies the IP address that must be used to reach the PCE. It should be a loop-back address that is always reachable, this TLV is mandatory, and must be present within the PCE Discovery TLV. The Path Scope Sub-TLV indicates the PCE path computation scopes, which refers to the PCE ability to compute or participate in the computation of intra-area, inter-area, inter-AS or inter-layer TE LSPs.

PCE extensions to OSPFv2 include support for the Router Information Link State Advertisement (RI LSA). OSPFv2 is extended to receive all area scopes (LSA Types 9, 10, and 11). However, OSPFv2 originates only area scope Type 10.

For detailed information for the Path Computation Element feature see the *Implementing MPLS Traffic Engineering* module of the *MPLS Configuration guide* and the following IETF drafts:

- draft-ietf-ospf-cap-09
- draft-ietf-pce-disco-proto-ospf-00

Management Information Base (MIB) for OSPFv3

Cisco IOS XR supports full MIBs and traps for OSPFv3, as defined in RFC 5643. The RFC 5643 defines objects of the Management Information Base (MIB) for use with the Open Shortest Path First (OSPF) Routing Protocol for IPv6 (OSPF version 3).

The OSPFv3 MIB implementation is based on the IETF draft *Management Information Base for OSPFv3 (draft-ietf-ospf-ospfv3-mib-8)*. Users need to update the NMS application to pick up the new MIB when upgraded to RFC 5643.

Multiple OSPFv3 Instances

SNMPv3 supports "contexts" that can be used to implement MIB views on multiple OSPFv3 instances, in the same system.

OSPFv3 Timers Update

The Open Shortest Path First version 3 (OSPFv3) timers link-state advertisements (LSAs) and shortest path first (SPF) throttle default values are updated to:

- **timers throttle lsa all**—*start-interval*: 50 milliseconds and *hold-interval*: 200 milliseconds
- **timers throttle spf**—*spf-start*: 50 milliseconds, *spf-hold*: 200 milliseconds, *spf-max-wait*: 5000 milliseconds

