

OSPF

This chapter describes how to configure the Cisco ASA to route data, perform authentication, and redistribute routing information using the Open Shortest Path First (OSPF) routing protocol.

- About OSPF, on page 1
- Guidelines for OSPF, on page 4
- Configure OSPFv2, on page 6
- Configure OSPFv2 Router ID, on page 10
- Configure OSPF Fast Hello Packets, on page 11
- Customize OSPFv2, on page 11
- Configure OSPFv3, on page 26
- Configure Graceful Restart, on page 46
- Example for OSPFv2, on page 51
- Examples for OSPFv3, on page 52
- Monitoring OSPF, on page 53
- History for OSPF, on page 56

About OSPF

OSPF is an interior gateway routing protocol that uses link states rather than distance vectors for path selection. OSPF propagates link-state advertisements rather than routing table updates. Because only LSAs are exchanged instead of the entire routing tables, OSPF networks converge more quickly than RIP networks.

OSPF uses a link-state algorithm to build and calculate the shortest path to all known destinations. Each router in an OSPF area contains an identical link-state database, which is a list of each of the router usable interfaces and reachable neighbors.

The advantages of OSPF over RIP include the following:

- OSPF link-state database updates are sent less frequently than RIP updates, and the link-state database is updated instantly, rather than gradually, as stale information is timed out.
- Routing decisions are based on cost, which is an indication of the overhead required to send packets across a certain interface. The ASA calculates the cost of an interface based on link bandwidth rather than the number of hops to the destination. The cost can be configured to specify preferred paths.

The disadvantage of shortest path first algorithms is that they require a lot of CPU cycles and memory.

The ASA can run two processes of OSPF protocol simultaneously on different sets of interfaces. You might want to run two processes if you have interfaces that use the same IP addresses (NAT allows these interfaces to coexist, but OSPF does not allow overlapping addresses). Or you might want to run one process on the inside and another on the outside, and redistribute a subset of routes between the two processes. Similarly, you might need to segregate private addresses from public addresses.

You can redistribute routes into an OSPF routing process from another OSPF routing process, a RIP routing process, or from static and connected routes configured on OSPF-enabled interfaces.

The ASA supports the following OSPF features:

- Intra-area, inter-area, and external (Type I and Type II) routes.
- Virtual links.
- LSA flooding.
- Authentication to OSPF packets (both password and MD5 authentication).
- Configuring the ASA as a designated router or a designated backup router. The ASA also can be set up as an ABR.
- Stub areas and not-so-stubby areas.
- Area boundary router Type 3 LSA filtering.

OSPF supports MD5 and clear text neighbor authentication. Authentication should be used with all routing protocols when possible because route redistribution between OSPF and other protocols (such as RIP) can potentially be used by attackers to subvert routing information.

If NAT is used, if OSPF is operating on public and private areas, and if address filtering is required, then you need to run two OSPF processes—one process for the public areas and one for the private areas.

A router that has interfaces in multiple areas is called an Area Border Router (ABR). A router that acts as a gateway to redistribute traffic between routers using OSPF and routers using other routing protocols is called an Autonomous System Boundary Router (ASBR).

An ABR uses LSAs to send information about available routes to other OSPF routers. Using ABR Type 3 LSA filtering, you can have separate private and public areas with the ASA acting as an ABR. Type 3 LSAs (inter-area routes) can be filtered from one area to other, which allows you to use NAT and OSPF together without advertising private networks.



Note Only Type 3 LSAs can be filtered. If you configure the ASA as an ASBR in a private network, it will send Type 5 LSAs describing private networks, which will get flooded to the entire AS, including public areas.

If NAT is employed but OSPF is only running in public areas, then routes to public networks can be redistributed inside the private network, either as default or Type 5 AS external LSAs. However, you need to configure static routes for the private networks protected by the ASA. Also, you should not mix public and private networks on the same ASA interface.

You can have two OSPF routing processes, one RIP routing process, and one EIGRP routing process running on the ASA at the same time.

OSPF Support for Fast Hello Packets

The OSPF Support for Fast Hello Packets feature provides a way to configure the sending of hello packets in intervals less than one second. Such a configuration would result in faster convergence in an Open Shortest Path First (OSPF) network.

Prerequisites for OSPF Support for Fast Hello Packets

OSPF must be configured in the network already or configured at the same time as the OSPF Support for Fast Hello Packets feature.

About OSPF Support for Fast Hello Packets

The key concepts related to OSPF support for fast hello packets and the benefits of OSPF Fast Hello Packets are described below:

OSPF Hello Interval and Dead Interval

OSPF hello packets are packets that an OSPF process sends to its OSPF neighbors to maintain connectivity with those neighbors. The hello packets are sent at a configurable interval (in seconds). The defaults are 10 seconds for an Ethernet link and 30 seconds for a non broadcast link. Hello packets include a list of all neighbors for which a hello packet has been received within the dead interval. The dead interval is also a configurable interval (in seconds), and defaults to four times the value of the hello interval. The value of all hello intervals must be the same within a network. Likewise, the value of all dead intervals must be the same within a network.

These two intervals work together to maintain connectivity by indicating that the link is operational. If a router does not receive a hello packet from a neighbor within the dead interval, it will declare that neighbor to be down.

OSPF Fast Hello Packets

OSPF fast hello packets refer to hello packets being sent at intervals of less than 1 second. To understand fast hello packets, you should already understand the relationship between OSPF hello packets and the dead interval. See OSPF Hello Interval and Dead Interval, on page 3.

OSPF fast hello packets are achieved by using the ospf dead-interval command. The dead interval is set to 1 second, and the hello-multiplier value is set to the number of hello packets you want sent during that 1 second, thus providing subsecond or "fast" hello packets.

When fast hello packets are configured on the interface, the hello interval advertised in the hello packets that are sent out this interface is set to 0. The hello interval in the hello packets received over this interface is ignored.

The dead interval must be consistent on a segment, whether it is set to 1 second (for fast hello packets) or set to any other value. The hello multiplier need not be the same for the entire segment as long as at least one hello packet is sent within the dead interval.

Benefits of OSPF Fast Hello Packets

The benefit of the OSPF Fast Hello Packets feature is that your OSPF network will experience faster convergence time than it would without fast hello packets. This feature allows you to detect lost neighbors within 1 second. It is especially useful in LAN segments, where neighbor loss might not be detected by the Open System Interconnection (OSI) physical layer and data-link layer.

Implementation Differences Between OSPFv2 and OSPFv3

OSPFv3 is not backward compatible with OSPFv2. To use OSPF to route both IPv4 and IPv6 traffic, you must run both OSPFv2 and OSPFv3 at the same time. They coexist with each other, but do not interact with each other.

The additional features that OSPFv3 provides include the following:

- Protocol processing per link.
- Removal of addressing semantics.
- Addition of flooding scope.
- Support for multiple instances per link.
- Use of the IPv6 link-local address for neighbor discovery and other features.
- LSAs expressed as prefix and prefix length.
- Addition of two LSA types.
- Handling of unknown LSA types.
- Authentication support using the IPsec ESP standard for OSPFv3 routing protocol traffic, as specified by RFC-4552.

Guidelines for OSPF

Context Mode Guidelines

OSPFv2 supports single and multiple context mode.

- OSPFv2 instances cannot form adjacencies with each other across shared interfaces because, by default, inter-context exchange of multicast traffic is not supported across shared interfaces. However, you can use the static neighbor configuration under OSPFv2 process configuration under OSPFv2 process to bring up OSPFv2 neighbourship on a shared interface.
- Inter-context OSPFv2 on separate interfaces is supported.

OSPFv3 supports single mode only.

Key Chain Authentication Guidelines

OSPFv2 supports key chain authentication on both single and multiple mode, both in physical and virtual modes. However, in multiple mode, you can configure the key chain only in context mode.

- The rotating keys are applicable only for OSPFv2 protocol. OSPF area authentication with key chain is not supported.
- The existing MD5 authentication without time range in OSPFv2 is still supported along with new rotating keys.
- Though the platform supports SHA1 and MD5 cryptographic algorithms, only MD5 cryptographic algorithm is used for authentication.

Firewall Mode Guidelines

OSPF supports routed firewall mode only. OSPF does not support transparent firewall mode.

Failover Guidelines

OSPFv2 and OSPFv3 support Stateful Failover.

IPv6 Guidelines

- OSPFv2 does not support IPv6.
- OSPFv3 supports IPv6.
- OSPFv3 uses IPv6 for authentication.
- The ASA installs OSPFv3 routes into the IPv6 RIB, provided it is the best route.
- OSPFv3 packets can be filtered out using IPv6 ACLs in the **capture** command.

OSPFv3 Hello Packets and GRE

Typically, OSPF traffic does not pass through GRE tunnel. When OSPFv3 on IPv6 is encapsulated inside GRE, the IPv6 header validation for security check such as Multicast Destination fails. The packet is dropped due to the implicit security check validation, as this packet has destination IPv6 multicast.

You may define a pre-filter rule to bypass GRE traffic. However, with pre-filter rule, inner packets would not be interrogated by the inspection engine.

Clustering Guidelines

- OSPFv3 encryption is not supported. An error message appears if you try to configure OSPFv3 encryption in a clustering environment.
- In Spanned interface mode, dynamic routing is not supported on management-only interfaces.
- In Individual interface mode, make sure that you establish the control and data units as either OSPFv2 or OSPFv3 neighbors.
- In Individual interface mode, OSPFv2 adjacencies can only be established between two contexts on a shared interface on the control unit. Configuring static neighbors is supported only on point-to point-links; therefore, only one neighbor statement is allowed on an interface.
- When a control role change occurs in the cluster, the following behavior occurs:
  - In spanned interface mode, the router process is active only on the control unit and is in a suspended state on the data units. Each cluster unit has the same router ID because the configuration has been synchronized from the control unit. As a result, a neighboring router does not notice any change in the router ID of the cluster during a role change.
  - In individual interface mode, the router process is active on all the individual cluster units. Each cluster unit chooses its own distinct router ID from the configured cluster pool. A control role change in the cluster does not change the routing topology in any way.

Multiprotocol Label Switching (MPLS) and OSPF Guidelines

When a MPLS-configured router sends Link State (LS) update packets containing opaque Type-10 link-state advertisements (LSAs) that include an MPLS header, authentication fails and the appliance silently drops the update packets, rather than acknowledging them. Eventually the peer router will terminate the neighbor relationship because it has not received any acknowledgments.

Disable the opaque capability on the ASA to ensure that the neighbor relationship remains stable:

```
router ospf process_ID_number
no nsf ietf helper
no capability opaque
```

Route Redistribution Guidelines

Redistribution of route maps with IPv4 or IPv6 prefix list on OSPFv2 or OSPFv3 is not supported. Use connected routes on OSPF for redistribution.

Additional Guidelines

- OSPFv2 and OSPFv3 support multiple instances on an interface.
- OSPFv3 supports encryption through ESP headers in a non-clustered environment.
- OSPFv3 supports Non-Payload Encryption.
- OSPFv2 supports Cisco NSF Graceful Restart and IETF NSF Graceful Restart mechanisms as defined in RFCs 4811, 4812 & 3623 respectively.
- OSPFv3 supports Graceful Restart mechanism as defined in RFC 5187.
- There is a limit to the number of intra area (type 1) routes that can be distributed. For these routes, a single type-1 LSA contains all prefixes. Because the system has a limit of 35 KB for packet size, 3000 routes result in a packet that exceeds the limit. Consider 2900 type 1 routes to be the maximum number supported.
- To avoid adjacency flaps due to route updates being dropped if the route update is larger than the minimum MTU on the link, ensure that you configure the same MTU on the interfaces on both sides of the link.

Configure OSPFv2

This section describes how to enable an OSPFv2 process on the ASA.

After you enable OSPFv2, you need to define a route map. For more information, see Define a Route Map. Then you generate a default route. For more information, see Configure a Static Route.

After you have defined a route map for the OSPFv2 process, you can customize it for your particular needs, To learn how to customize the OSPFv2 process on the ASA, see Customize OSPFv2, on page 11.

To enable OSPFv2, you need to create an OSPFv2 routing process, specify the range of IP addresses associated with the routing process, then assign area IDs associated with that range of IP addresses.

You can enable up to two OSPFv2 process instances. Each OSPFv2 process has its own associated areas and networks.

To enable OSPFv2, perform the following steps:

Procedure

Step 1 Create an OSPF routing process:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 2
```

The *process_id* argument is an internally used identifier for this routing process and can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

If there is only one OSPF process enabled on the ASA, then that process is selected by default. You cannot change the OSPF process ID when editing an existing area.

Step 2 Define the IP addresses on which OSPF runs and the area ID for that interface:

network *ip_address mask* **area** *area_id*

Example:

```
ciscoasa(config)# router ospf 2
ciscoasa(config-rtr)# network 10.0.0.0 255.0.0.0 area 0
```

When adding a new area, enter the area ID. You can specify the area ID as either a decimal number or an IP address. Valid decimal values range from 0 to 4294967295. You cannot change the area ID when editing an existing area.

Configure a Key Chain for Authentication

To enhance data security and protection of devices, you can enable rotating keys for authenticating IGP peers. The rotating keys prevent any malicious user from guessing the keys used for routing protocol authentication and thereby protecting the network from advertising incorrect routes and redirecting traffic. Changing the keys frequently reduces the risk of them eventually being guessed. When configuring authentication for routing protocols that provide key chains, configure the keys in a key chain to have overlapping lifetimes. This helps to prevent loss of key-secured communication due to absence of an active key. If the key lifetime expires and no active keys are found, OSPF uses the last valid key to maintain the adjacency with the peers.

This section describes how to create a key chain for OSPF peer authentication. After configuring a key chain object, you can use it in defining the OSPFv2 authentication for an interface and for a virtual link. Use the same authentication type (MD5 or Key Chain) and key ID for the peers to establish a successful adjacency. To learn how to define authentication for an interface, see Configure OSPFv2 Interface Parameters, on page 15.

To configure a key chain, perform the following steps:

Procedure

Step 1 Configure a key chain with a name:

key chain*key-chain-name*

Example:

```
ciscoasa(config)# key chain CHAIN1
ciscoasa(config-keychain)#
```

You can now proceed to define the associated parameters for the key chain.

Step 2 Configure the identifier for the key chain:

key*key-id*

The key id value can be between 0 and 255. Use the value 0 only when you want to signal an invalid key.

Example:

```
ciscoasa(config-keychain)# key 1
ciscoasa(config-keychain-key)#
```

Step 3 Configure the key or password for the key chain:

key-string [**0** | **8**] *key-string-text*

- Use **0** to indicate an unencrypted password follows as shown in the example.
- Use **8** to indicate an encrypted password to follow.
- The password can be of a maximum length of 80 characters.
- The passwords cannot be a single digit nor those starting with a digit followed by a white space. For example, "0 pass" or "1" are invalid.

Example:

```
ciscoasa(config-keychain-key)# key-string 0 CHAIN1KEY1STRING
ciscoasa(config-keychain-key)#
```

Step 4 Configure the cryptographic algorithm for the key chain:

cryptographic-algorithm*md5*

You need to provide the cryptographic authentication algorithm. Though the platform supports SHA1 and MD5, only MD5 is supported for key chain management.

Example:

```
ciscoasa(config-keychain-key)# cryptographic-algorithm md5
ciscoasa(config-keychain-key)#
```

Step 5 (Optional) Configure the lifetime settings for the key chain:

accept-lifetime [**local** | *start-time*] [**duration** *duration value* | **infinite** | *end-time*]

send-lifetime [**ocal** | *start-time*] [**duration** *duration value* | **infinite** | *end-time*]

You can specify the time interval for the device to accept/send the key during key exchange with another device. The end time can be the duration or the absolute time when the accept/send lifetime ends or infinite.

Following are the validation rules for the start and end values:

- Start lifetime cannot be null when the end lifetime is specified.
- The start lifetime for accept or send lifetime must be earlier than the end lifetime.

Example:

```
ciscoasa(config-keychain-key)# accept-lifetime 11:22:33 1 SEP 2018 infinite
ciscoasa(config-keychain-key)#
```

You can use the **show key chain** command to view the start-up key chain configuration on the device; **show run key chain** command to view the key chain configuration that is currently running on the device.

```
ciscoasa# show key chain
Key-chain CHAIN2:
    key 1 -- text "KEY1CHAIN2"
        accept lifetime (always valid) - (always valid) [valid now]
        send lifetime (always valid) - (always valid) [valid now]
  * key 2 -- text "(unset)"
        accept lifetime (11:00:12 UTC Sep 1 2018) - (11:12:12 UTC Sep 1 2018)
        send lifetime (always valid) - (always valid) [valid now]
Key-chain CHAIN1:
    key 1 -- text "CHAIN1KEY1STRING"
        accept lifetime (11:22:33 UTC Sep 1 2018) - (-1 seconds)
        send lifetime (always valid) - (always valid) [valid now]
ciscoasa#
```

```
ciscoasa# sh run key chain
key chain CHAIN2
 key 1
  key-string KEY1CHAIN2
  cryptographic-algorithm md5
 key 2
  accept-lifetime 11:00:12 Sep 1 2018 11:12:12 Sep 1 2018
  cryptographic-algorithm md5
key chain CHAIN1
 key 1
  key-string CHAIN1KEY1STRING
  accept-lifetime 11:22:33 Sep 1 2018 duration -1
  cryptographic-algorithm md5
ciscoasa# sh run key chain CHAIN1
key chain CHAIN1
 key 1
  key-string CHAIN1KEY1STRING
  accept-lifetime 11:22:33 Sep 1 2018 duration -1
  cryptographic-algorithm md5
ciscoasa#
```

What to do next

You can now apply the configured key chain to define the OSPFv2 authentication for an interface.

• Configure OSPFv2 Interface Parameters, on page 15

Configure OSPFv2 Router ID

The OSPF Router-ID is used to identify a specific device within an OSPF database. No two routers in an OSPF system can have the same router-id.

If a router-id is not configured manually in the OSPF routing process the router will automatically configure a router-id determined from the highest IP address of an active interface. When configuring a router-id, the neighbors will not be updated automatically until that router has failed or the OSPF process has been cleared and the neighbor relationship has been re-established.

Manually Configure OSPF Router-ID

This section describes how to manually configure router-id in OSPFv2 process on the ASA.

Procedure

Step 1 To use a fixed router ID, use the **router-id** command.

router-id *ip-address*

Example:

```
ciscoasa(config-router)# router-id 193.168.3.3
```

Step 2 To revert to the previous OSPF router ID behavior, use the **no router-id** command.

no router-id *ip-address*

Example:

```
ciscoasa(config-router)# no router-id 193.168.3.3
```

Router ID Behaviour while Migrating

While migrating OSPF configuration from one ASA , say ASA 1 to another ASA, say ASA 2, the following router id selection behaviour is observed:

1. ASA 2 does not use any IP address for OSPF router-id when all interfaces are in shutdown mode. The possibilities for configuring router-id when all interfaces are in "admin down" state or shutdown mode are:
   - If ASA 2 does not have any router-id configured before, you would see this message:

     ```
     %OSPF: Router process 1 is not running, please configure a router-id
     ```

     After the first interface is brought up, ASA 2 will take IP address of this interface as router id.

- If ASA 2 had router-id configured before and all interfaces were in "admin down" state when "no router-id" command was issued, ASA 2 will use old router id. ASA 2 uses the old router id, even if IP addresses on the interface that is brought up is changed, until "clear ospf process" command is issued.

2. ASA 2 uses new router id, when ASA 2 had router-id configured before and at least one of interfaces were not in "admin down" state or shutdown mode when "no router-id" command was issued. ASA 2 will use new router id from the IP address of the interfaces even when interfaces are in "down/down" state.

Configure OSPF Fast Hello Packets

This section describes how to configure OSPF Fast Hello Packets.

Procedure

Step 1 Configure an interface:

interface *port-channel number*

Example:

```
ciscoasa(config)# interface port-channel 10
```

The *number* argument indicates the port-channel interface number.

Step 2 Set the interval during which at least one hello packet must be received, or else the neighbor is considered down:

ospf dead-interval minimal hello-multiplier *no.of times*

Example:

```
ciscoasa(config-if)# ospf dead-interval minimal hell0-multiplier 5
ciscoasa
```

The no. of times argument indicates the number of hello packets to be sent every second. Valid values are between 3 and 20.

In this example, OSPF Support for Fast Hello Packets is enabled by specifying the minimal keyword and the hello-multiplier keyword and value. Because the multiplier is set to 5, five hello packets will be sent every second.

Customize OSPFv2

This section explains how to customize the OSPFv2 processes.

Redistribute Routes Into OSPFv2

The ASA can control the redistribution of routes between OSPFv2 routing processes.



Note If you want to redistribute a route by defining which of the routes from the specified routing protocol are allowed to be redistributed into the target routing process, you must first generate a default route. See Configure a Static Route, and then define a route map according to Define a Route Map.

To redistribute static, connected, RIP, or OSPFv2 routes into an OSPFv2 process, perform the following steps:

Procedure

Step 1 Create an OSPF routing process:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 2
```

The *process_id* argument is an internally used identifier for this routing process and can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Redistribute connected routes into the OSPF routing process:

redistribute connected [[**metric** *metric-value*] [**metric-type** {**type-1** | **type-2**}] [**tag** *tag_value*] [**subnets**] [**route-map** *map_name*]

Example:

```
ciscoasa(config)# redistribute connected 5 type-1 route-map-practice
```

Step 3 Redistribute static routes into the OSPF routing process:

redistribute static [**metric** *metric-value*] [**metric-type** {**type-1** | **type-2**}] [**tag** *tag_value*] [**subnets**] [**route-map** *map_name*

Example:

```
ciscoasa(config)# redistribute static 5 type-1 route-map-practice
```

Step 4 Redistribute routes from an OSPF routing process into another OSPF routing process:

redistribute ospf pid [**match {internal** | **external** [**1** | **2]** | **nssa-external** [**1** | **2**]}] [**metric** *metric-value*] [**metric-type** {**type-1** | **type-2**}] [**tag** *tag_value*] [**subnets**] [**route-map** *map_name*]

Example:

```
ciscoasa(config)# route-map 1-to-2 permit
ciscoasa(config-route-map)# match metric 1
```

```
ciscoasa(config-route-map)# set metric 5
ciscoasa(config-route-map)# set metric-type type-1
ciscoasa(config-route-map)# router ospf 2
ciscoasa(config-rtr)# redistribute ospf 1 route-map 1-to-2
```

You can either use the **match** options in this command to match and set route properties, or you can use a route map. The **subnets** option does not have equivalents in the **route-map** command. If you use both a route map and **match** options in the **redistribute** command, then they must match.

The example shows route redistribution from OSPF process 1 into OSPF process 2 by matching routes with a metric equal to 1. The ASA redistributes these routes as external LSAs with a metric of 5 and a metric type of Type 1.

Step 5 Redistribute routes from a RIP routing process into the OSPF routing process:

redistribute rip [**metric** *metric-value*] [**metric-type** {**type-1** | **type-2**}] [**tag** *tag_value*] [**subnets**] [**route-map** *map_name*]

Example:

```
ciscoasa(config)# redistribute rip 5
ciscoasa(config-route-map)# match metric 1
ciscoasa(config-route-map)# set metric 5
ciscoasa(config-route-map)# set metric-type type-1
ciscoasa(config-rtr)# redistribute ospf 1 route-map 1-to-2
```

Step 6 Redistribute routes from an EIGRP routing process into the OSPF routing process:

redistribute eigrp as-num [**metric** *metric-value*] [**metric-type** {**type-1** | **type-2**}] [**tag** *tag_value*] [**subnets**] [**route-map** *map_name*]

Example:

```
ciscoasa(config)# redistribute eigrp 2
ciscoasa(config-route-map)# match metric 1
ciscoasa(config-route-map)# set metric 5
ciscoasa(config-route-map)# set metric-type type-1
ciscoasa(config-rtr)# redistribute ospf 1 route-map 1-to-2
```

Configure Route Summarization When Redistributing Routes Into OSPFv2

When routes from other protocols are redistributed into OSPF, each route is advertised individually in an external LSA. However, you can configure the ASA to advertise a single route for all the redistributed routes that are included for a specified network address and mask. This configuration decreases the size of the OSPF link-state database.

Routes that match the specified IP address mask pair can be suppressed. The tag value can be used as a match value for controlling redistribution through route maps.

Add a Route Summary Address

To configure the software advertisement on one summary route for all redistributed routes included for a network address and mask, perform the following steps:

Procedure

Step 1 Create an OSPF routing process:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 1
```

The *process_id* argument is an internally used identifier for this routing process and can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Set the summary address:

summary-address *ip_address mask* [**not-advertise**] [tag *tag*]

Example:

```
ciscoasa(config)# router ospf 1
ciscoasa(config-rtr)# summary-address 10.1.0.0 255.255.0.0
```

In this example, the summary address 10.1.0.0 includes addresses 10.1.1.0, 10.1.2.0, 10.1.3.0, and so on. Only the 10.1.0.0 address is advertised in an external link-state advertisement.

Configure Route Summarization Between OSPFv2 Areas

Route summarization is the consolidation of advertised addresses. This feature causes a single summary route to be advertised to other areas by an area boundary router. In OSPF, an area boundary router advertises networks in one area into another area. If the network numbers in an area are assigned in a way so that they are contiguous, you can configure the area boundary router to advertise a summary route that includes all the individual networks within the area that fall into the specified range.

To define an address range for route summarization, perform the following steps:

Procedure

Step 1 Create an OSPF routing process and enters router configuration mode for this OSPF process:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 1
```

The *process_id* argument is an internally used identifier for this routing process. It can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Set the address range:

area *area-id* range *ip-address mask* [**advertise** | **not-advertise**]

Example:

```
ciscoasa(config-rtr)# area 17 range 12.1.0.0 255.255.0.0
```

In this example, the address range is set between OSPF areas.

Configure OSPFv2 Interface Parameters

You can change some interface-specific OSPFv2 parameters, if necessary. You are not required to change any of these parameters, but the following interface parameters must be consistent across all routers in an attached network: **ospf hello-interval**, **ospf dead-interval**, and **ospf authentication-key**. If you configure any of these parameters, be sure that the configurations for all routers on your network have compatible values.

To configure OSPFv2 interface parameters, perform the following steps:

Procedure

Step 1 Create an OSPF routing process:

router ospf*process-id*

Example:

```
ciscoasa(config)# router ospf 2
```

The *process-id* argument is an internally used identifier for this routing process and can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Define the IP addresses on which OSPF runs and the area ID for that interface:

network*ip-address mask***area***area-id*

Example:

```
ciscoasa(config)# router ospf 2
ciscoasa(config-rtr)# network 10.0.0.0 255.0.0.0 area 0
```

Step 3 Enter interface configuration mode:

interface*interface-name*

Example:

```
ciscoasa(config)# interface my_interface
```

Step 4 Specify the authentication type for an interface:

ospf authentication [**key-chain** *key-chain-name* | **message-digest** | **null**]

Provide the key chain name configured. For information on configuring key chain, see Configure a Key Chain for Authentication, on page 7

Example:

```
ciscoasa(config-interface)# ospf authentication message-digest
```

Step 5 Assign a password to be used by neighboring OSPF routers on a network segment that is using the OSPF simple password authentication:

ospf authentication-key*key*

Example:

```
ciscoasa(config-interface)# ospf authentication-key cisco
```

The *key* argument can be any continuous string of characters up to 8 bytes in length.

The password created by this command is used as a key that is inserted directly into the OSPF header when the ASA software originates routing protocol packets. A separate password can be assigned to each network on a per-interface basis. All neighboring routers on the same network must have the same password to be able to exchange OSPF information.

Step 6 Explicitly specify the cost of sending a packet on an OSPF interface:

ospf cost*cost*

Example:

```
ciscoasa(config-interface)# ospf cost 20
```

The *cost* is an integer from 1 to 65535.

In this example, the cost is set to 20.

Step 7 Set the number of seconds that a device must wait before it declares a neighbor OSPF router down because it has not received a hello packet:

ospf dead-interval*seconds*

Example:

```
ciscoasa(config-interface)# ospf dead-interval 40
```

The value must be the same for all nodes on the network.

Step 8 Specify the length of time between the hello packets that the ASA sends on an OSPF interface:

ospf hello-interval*seconds*

Example:

```
ciscoasa(config-interface)# ospf hello-interval 10
```

The value must be the same for all nodes on the network.

Step 9 Enable OSPF MD5 authentication:

ospf message-digest-key*key-id***md5***key*

Example:

```
ciscoasa(config-interface)# ospf message-digest-key 1 md5 cisco
```

The following argument values can be set:

key-id—An identifier in the range from 1 to 255.

key—An alphanumeric password of up to 16 bytes.

Usually, one key per interface is used to generate authentication information when sending packets and to authenticate incoming packets. The same key identifier on the neighbor router must have the same key value.

We recommend that you not keep more than one key per interface. Every time you add a new key, you should remove the old key to prevent the local system from continuing to communicate with a hostile system that knows the old key. Removing the old key also reduces overhead during rollover.

Step 10 Set the priority to help determine the OSPF designated router for a network:

ospf priority *number-value*

Example:

```
ciscoasa(config-interface)# ospf priority 20
```

The *number_value* argument ranges from 0 to 255.

In multiple context mode, for shared interfaces, specify 0 to ensure the device does not become the designated router. OSPFv2 instances cannot form adjacencies with each other across shared interfaces.

Step 11 Specify the number of seconds between LSA retransmissions for adjacencies belonging to an OSPF interface:

ospf retransmit-interval*number-value*

Example:

```
ciscoasa(config-interface)# ospf retransmit-interval seconds
```

The value for *seconds* must be greater than the expected round-trip delay between any two routers on the attached network. The range is from 1 to 8192 seconds. The default value is 5 seconds.

Step 12 Set the estimated number of seconds required to send a link-state update packet on an OSPF interface:

ospf transmit-delay*seconds*

Example:

```
ciscoasa(config-interface)# ospf transmit-delay 5
```

The *seconds* value ranges from 1 to 8192 seconds.The default value is 1 second.

Step 13 Set the number of hello packets sent during 1 second:

ospf dead-interval minimal hello-interval multiplier*integer*

Example:

```
ciscoasa(config-if)# ospf dead-interval minimal hello-multiplier 6
```

Valid values are integers between 3 and 20.

Step 14 Specify the interface as a point-to-point, non-broadcast network:

ospf network point-to-point non-broadcast

Example:

```
ciscoasa(config-interface)# ospf network point-to-point non-broadcast
```

When you designate an interface as point-to-point and non-broadcast, you must manually define the OSPF neighbor; dynamic neighbor discovery is not possible. See Define Static OSPFv2 Neighbors, on page 21 for more information. Additionally, you can only define one OSPF neighbor on that interface.

Configure OSPFv2 Area Parameters

You can configure several OSPF area parameters. These area parameters (shown in the following task list) include setting authentication, defining stub areas, and assigning specific costs to the default summary route. Authentication provides password-based protection against unauthorized access to an area.

Stub areas are areas into which information on external routes is not sent. Instead, there is a default external route generated by the ABR into the stub area for destinations outside the autonomous system. To take advantage of the OSPF stub area support, default routing must be used in the stub area. To further reduce the number of LSAs sent into a stub area, you can use the **no-summary** keyword of the **area stub** command on the ABR to prevent it from sending a summary link advertisement (LSA Type 3) into the stub area.

Procedure

Step 1 Create an OSPF routing process:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 2
```

The *process_id* argument is an internally used identifier for this routing process and can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Enable authentication for an OSPF area:

area *area-id* authentication

Example:

```
ciscoasa(config-rtr)# area 0 authentication
```

Step 3 Enable MD5 authentication for an OSPF area:

area *area-id* authentication message-digest

Example:

```
ciscoasa(config-rtr)# area 0 authentication message-digest
```

Configure OSPFv2 Filter Rules

Use the following procedure to filter routes or networks received or transmitted in OSPF updates.

Procedure

Step 1 Enable an OSPF routing process and enter router configuration mode:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 2
```

Step 2 Filter routes or networks received in incoming or advertised in outgoing OSPF updates:

distribute-list *acl-number* **in** [**interface** *ifname*]

distribute-list *acl-number* **out** [*protocol process-number* | **connected** | **static**]

The argument *acl-number* specifies IP access list number. The access list defines which networks are to be received and which are to be suppressed in routing updates.

To apply the filter to incoming updates, specify **in**. You can optionally specify an interface to limit the filter to updates received on that interface.

To apply the filter to outbound updates, specify **out**. You can optionally specify a protocol (**bgp**, **eigrp**, **ospf**, or **rip**) with a process number (except for RIP) to apply to the distribution list. You can also filter on whether the peers and networks were learned through **connected** or **static** routes.

Example:

```
ciscoasa(config-rtr)# distribute-list ExampleAcl in interface inside
```

Configure an OSPFv2 NSSA

The OSPFv2 implementation of an NSSA is similar to an OSPFv2 stub area. NSSA does not flood Type 5 external LSAs from the core into the area, but it can import autonomous system external routes in a limited way within the area.

NSSA imports 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.

You can simplify administration if you are an ISP or a network administrator that must connect a central site using OSPFv2 to a remote site that is using a different routing protocol with NSSA.

Before the implementation of NSSA, the connection between the corporate site border router and the remote router could not be run as an OSPFv2 stub area because routes for the remote site could not be redistributed into the stub area, and two routing protocols needed to be maintained. A simple protocol such as RIP was usually run and handled the redistribution. With NSSA, you can extend OSPFv2 to cover the remote connection by defining the area between the corporate router and the remote router as an NSSA.

Before you use this feature, consider these guidelines:

- You can set a Type 7 default route that can be used to reach external destinations. When configured, the router generates a Type 7 default into the NSSA or the NSSA area boundary router.
- Every router within the same area must agree that the area is NSSA; otherwise, the routers cannot communicate with each other.

Procedure

Step 1 Create an OSPF routing processs:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 2
```

The *process_id* argument is an internally used identifier for this routing process. It can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Define an NSSA area:

area *area-id* nssa [**no-redistribution**] [**default-information-originate**]

Example:

```
ciscoasa(config-rtr)# area 0 nssa
```

Step 3 Set the summary address and helps reduce the size of the routing table:

summary-address *ip_address mask* [**not-advertise**] [tag *tag*]

Example:

```
ciscoasa(config-rtr)# summary-address 10.1.0.0 255.255.0.0
```

Using this command for OSPF causes an OSPF ASBR to advertise one external route as an aggregate for all redistributed routes that are covered by the address.

In this example, the summary address 10.1.0.0 includes addresses 10.1.1.0, 10.1.2.0, 10.1.3.0, and so on. Only the 10.1.0.0 address is advertised in an external link-state advertisement.

Note OSPF does not support summary-address 0.0.0.0 0.0.0.0.

Configure an IP Address Pool for Clustering (OSPFv2 and OSPFv3)

You can assign a range of IPv4 addresses for the router ID cluster pool if you are using Individual Interface clustering.

To assign a range of IPv4 addresses for the router ID cluster pool in Individual Interface clustering for OSPFv2 and OSPFv3, enter the following command:

Procedure

Specify the router ID cluster pool for Individual Interface clustering:

router-id cluster-pool hostname | A.B.C.D ip_pool

Example:

```
hostname(config)# ip local pool rpool 1.1.1.1-1.1.1.4
hostname(config)# router ospf 1
hostname(config-rtr)# router-id cluster-pool rpool
hostname(config-rtr)# network 17.5.0.0 255.255.0.0 area 1
hostname(config-rtr)# log-adj-changes
```

The **cluster-pool** keyword enables configuration of an IP address pool when Individual Interface clustering is configured. The **hostname** | **A.B.C.D.** keyword specifies the OSPF router ID for this OSPF process. The *ip_pool* argument specifies the name of the IP address pool.

Note If you are using clustering, then you do not need to specify an IP address pool for the router ID. If you do not configure an IP address pool, then the ASA uses the automatically generated router ID.

Define Static OSPFv2 Neighbors

You need to define static OSPFv2 neighbors to advertise OSPFv2 routes over a point-to-point, non-broadcast network. This feature lets you broadcast OSPFv2 advertisements across an existing VPN connection without having to encapsulate the advertisements in a GRE tunnel.

Before you begin, you must create a static route to the OSPFv2 neighbor. See Configure a Static Route for more information about creating static routes.

Procedure

Step 1 Create an OSPFv2 routing process:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 2
```

The *process_id* argument is an internally used identifier for this routing process and can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Define the OSPFv2 neighborhood:

neighbor *addr* [**interface** if_name]

Example:

```
ciscoasa(config-rtr)# neighbor 255.255.0.0 [interface my_interface]
```

The *addr* argument is the IP address of the OSPFv2 neighbor. The *if_name* argument is the interface used to communicate with the neighbor. If the OSPF v2neighbor is not on the same network as any of the directly connected interfaces, you must specify the interface.

Configure Route Calculation Timers

You can configure the delay time between when OSPFv2 receives a topology change and when it starts an SPF calculation. You also can configure the hold time between two consecutive SPF calculations.

Procedure

Step 1 Create an OSPFv2 routing process:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 2
```

The *process_id* argument is an internally used identifier for this routing process and can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Configure the route calculation times:

timers throttle spf *spf-start spf-hold spf-maximum*

Example:

```
ciscoasa(config-router)# timers throttle spf 500 500 600
```

The *spf-start* argument is the delay time (in milliseconds) between when OSPF receives a topology change and when it starts an SPF calculation. It can be an integer from 0 to 600000.

The *spf-hold* argument is the minimum time (in milliseconds) between two consecutive SPF calculations. It can be an integer from 0 to 600000.

The spf-maximum argument is the maximum time (in milliseconds) between two consecutive SPF calculations. It can be integer from 0 to 600000.

Log Neighbors Going Up or Down

By default, a syslog message is generated when an OSPFv2 neighbor goes up or down.

Configure the **log-adj-changes** command if you want to know about OSPFv2 neighbors going up or down without turning on the **debug ospf adjacency** command. The **log-adj-changes** command provides a higher level view of the peer relationship with less output. Configure the **log-adj-changes detail** command if you want to see messages for each state change.

Procedure

Step 1 Create an OSPFv2 routing process:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 2
```

The *process_id* argument is an internally used identifier for this routing process and can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Configure logging for neighbors going up or down:

log-adj-changes [**detail**]

Configure a Key Chain for Authentication

To enhance data security and protection of devices, you can enable rotating keys for authenticating IGP peers. The rotating keys prevent any malicious user from guessing the keys used for routing protocol authentication and thereby protecting the network from advertising incorrect routes and redirecting traffic. Changing the keys frequently reduces the risk of them eventually being guessed. When configuring authentication for routing protocols that provide key chains, configure the keys in a key chain to have overlapping lifetimes. This helps to prevent loss of key-secured communication due to absence of an active key. If the key lifetime expires and no active keys are found, OSPF uses the last valid key to maintain the adjacency with the peers.

This section describes how to create a key chain for OSPF peer authentication. After configuring a key chain object, you can use it in defining the OSPFv2 authentication for an interface and for a virtual link. Use the same authentication type (MD5 or Key Chain) and key ID for the peers to establish a successful adjacency. To learn how to define authentication for an interface, see Configure OSPFv2 Interface Parameters, on page 15.

To configure a key chain, perform the following steps:

Procedure

Step 1 Configure a key chain with a name:

key chain*key-chain-name*

Example:

```
ciscoasa(config)# key chain CHAIN1
ciscoasa(config-keychain)#
```

You can now proceed to define the associated parameters for the key chain.

Step 2 Configure the identifier for the key chain:

key*key-id*

The key id value can be between 0 and 255. Use the value 0 only when you want to signal an invalid key.

Example:

```
ciscoasa(config-keychain)# key 1
ciscoasa(config-keychain-key)#
```

Step 3 Configure the key or password for the key chain:

key-string [**0** | **8**] *key-string-text*

- Use **0** to indicate an unencrypted password follows as shown in the example.
- Use **8** to indicate an encrypted password to follow.
- The password can be of a maximum length of 80 characters.
- The passwords cannot be a single digit nor those starting with a digit followed by a white space. For example, "0 pass" or "1" are invalid.

Example:

```
ciscoasa(config-keychain-key)# key-string 0 CHAIN1KEY1STRING
ciscoasa(config-keychain-key)#
```

Step 4 Configure the cryptographic algorithm for the key chain:

cryptographic-algorithm*md5*

You need to provide the cryptographic authentication algorithm. Though the platform supports SHA1 and MD5, only MD5 is supported for key chain management.

Example:

```
ciscoasa(config-keychain-key)# cryptographic-algorithm md5
ciscoasa(config-keychain-key)#
```

Step 5 (Optional) Configure the lifetime settings for the key chain:

accept-lifetime [**local** | *start-time*] [**duration** *duration value* | **infinite** | *end-time*]

send-lifetime [**ocal** | *start-time*] [**duration** *duration value* | **infinite** | *end-time*]

You can specify the time interval for the device to accept/send the key during key exchange with another device. The end time can be the duration or the absolute time when the accept/send lifetime ends or infinite.

Following are the validation rules for the start and end values:

- Start lifetime cannot be null when the end lifetime is specified.
- The start lifetime for accept or send lifetime must be earlier than the end lifetime.

Example:

```
ciscoasa(config-keychain-key)# accept-lifetime 11:22:33 1 SEP 2018 infinite
ciscoasa(config-keychain-key)#
```

You can use the **show key chain** command to view the start-up key chain configuration on the device; **show run key chain** command to view the key chain configuration that is currently running on the device.

```
ciscoasa# show key chain
Key-chain CHAIN2:
    key 1 -- text "KEY1CHAIN2"
        accept lifetime (always valid) - (always valid) [valid now]
        send lifetime (always valid) - (always valid) [valid now]
  * key 2 -- text "(unset)"
        accept lifetime (11:00:12 UTC Sep 1 2018) - (11:12:12 UTC Sep 1 2018)
        send lifetime (always valid) - (always valid) [valid now]
Key-chain CHAIN1:
    key 1 -- text "CHAIN1KEY1STRING"
        accept lifetime (11:22:33 UTC Sep 1 2018) - (-1 seconds)
        send lifetime (always valid) - (always valid) [valid now]
ciscoasa#
```

```
ciscoasa# sh run key chain
key chain CHAIN2
 key 1
  key-string KEY1CHAIN2
  cryptographic-algorithm md5
 key 2
  accept-lifetime 11:00:12 Sep 1 2018 11:12:12 Sep 1 2018
  cryptographic-algorithm md5
key chain CHAIN1
 key 1
  key-string CHAIN1KEY1STRING
  accept-lifetime 11:22:33 Sep 1 2018 duration -1
  cryptographic-algorithm md5
ciscoasa# sh run key chain CHAIN1
key chain CHAIN1
 key 1
```

```
  key-string CHAIN1KEY1STRING
  accept-lifetime 11:22:33 Sep 1 2018 duration -1
  cryptographic-algorithm md5
ciscoasa#
```

What to do next

You can now apply the configured key chain to define the OSPFv2 authentication for an interface.

- Configure OSPFv2 Interface Parameters, on page 15

Configure OSPFv3

This section describes the tasks involved in configuring an OSPFv3 routing process.

Enable OSPFv3

To enable OSPFv3, you need to create an OSPFv3 routing process, create an area for OSPFv3, enable an interface for OSPFv3, then redistribute the route into the targeted OSPFv3 routing processes.

Procedure

Step 1 Create an OSPFv3 routing process:

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config)# ipv6 router ospf 10
```

The *process-id* argument is an internally used tag for this routing process and can be any positive integer. This tag does not have to match the tag on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Enable an interface:

interface *interface_name*

Example:

```
ciscoasa(config)# interface Gigabitethernet0/0
```

Step 3 Create the OSPFv3 routing process with the specified process ID and an area for OSPFv3 with the specified area ID:

ipv6 ospf *process-id* **area** *area_id*

Example:

```
ciscoasa(config)# ipv6 ospf 200 area 100
```

Configure OSPFv3 Interface Parameters

You can change certain interface-specific OSPFv3 parameters, if necessary. You are not required to change any of these parameters, but the following interface parameters must be consistent across all routers in an attached network: the hello interval and the dead interval. If you configure any of these parameters, be sure that the configurations for all routers on your network have compatible values.

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config-if)# ipv6 router ospf 10
```

The *process-id* argument is an internally used tag for this routing process and can be any positive integer. This tag does not have to match the tag on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Create an OSPFv3 area:.

ipv6 ospf **area** [*area-num*] [**instance**]

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
```

The *area-num* argument is the area for which authentication is to be enabled and can be either a decimal value or an IP address. The **instance** keyword specifies the area instance ID that is to be assigned to an interface. An interface can have only one OSPFv3 area. You can use the same area on multiple interfaces, and each interface can use a different area instance ID.

Step 3 Specify the cost of sending a packet on an interface:

ipv6 ospf cost *interface-cost*

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
```

The *interface-cost* argument specifies an unsigned integer value expressed as the link-state metric, which can range in value from 1 to 65535. The default cost is based on the bandwidth.

Step 4 Filter outgoing LSAs to an OSPFv3 interface:

ipv6 ospf database-filter all out

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf database-filter all out
```

All outgoing LSAs are flooded to the interface by default.

Step 5 Set the time period in seconds for which hello packets must not be seen before neighbors indicate that the router is down:

ipv6 ospf dead-interval *seconds*

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf dead-interval 60
```

The value must be the same for all nodes on the network and can range from 1 to 65535. The default is four times the interval set by the **ipv6 ospf hello-interval** command.

Step 6 Specify the encryption type for an interface:

ipv6 ospf encryption {**ipsec spi** *spi* **esp** *encryption-algorithm* [[*key-encryption-type*] *key*] *authentication-algorithm* [[*key-encryption-type*] *key* | **null**}

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf encryption ipsec spi 1001 esp null sha1 123456789A123456789B123456789C123456789D
```

The **ipsec** keyword specifies the IP security protocol. The **spi** *spi* keyword-argument pair specifies the security policy index, which must be in the range of 256 to 42949667295 and entered as a decimal.

The **esp** keyword specifies the encapsulating security payload. The *encryption-algorithm* argument specifies the encryption algorithm to be used with ESP. Valid values include the following:

- aes-cdc—Enables AES-CDC encryption.
- 3des—Enables 3DES encryption.
- des—Enables DES encryption.
- null—Specifies ESP with no encryption.

The *key-encryption-type* argument can be one of the following two values:

- 0—The key is not encrypted.
- 7—The key is encrypted.

The *key* argument specifies the number used in the calculation of the message digest. The number is 32 hexadecimal digits (16 bytes) long. The size of the key depends on the encryption algorithm used. Some algorithms, such as AES-CDC, allow you to choose the size of the key. The *authentication-algorithm* argument specifies the encryption authentication algorithm to be used, which can be one of the following:

- md5—Enables message digest 5 (MD5).
- sha1—Enables SHA-1.

The **null** keyword overrides area encryption.

If OSPFv3 encryption is enabled on an interface and a neighbor is on different area (for example, area 0), and you want the ASA to form adjacencies with that area, you must change the area on the ASA. After you have changed the area on the ASA to 0, there is a delay of two minutes before the OSPFv3 adjacency comes up.

Step 7 Specify the flood reduction of LSAs to the interface:

ipv6 ospf flood-reduction

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf flood reduction
```

Step 8 Specify the interval in seconds between hello packets sent on the interface:

ipv6 ospf hello-interval *seconds*

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf hello-interval 15
```

The value must be the same for all nodes on a specific network and can range from 1 to 65535. The default interval is 10 seconds for Ethernet interfaces and 30 seconds for non-broadcast interfaces.

Step 9 Disable the OSPF MTU mismatch detection when DBD packets are received:

ipv6 ospf mtu-ignore

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf mtu-ignore
```

OSPF MTU mismatch detection is enabled by default.

Step 10 Set the OSPF network type to a type other than the default, which depends on the network type:

ipv6 ospf network {**broadcast** | **point-to-point non-broadcast**}

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf network point-to-point non-broadcast
```

The **point-to-point non-broadcast** keyword sets the network type to point-to-point non-broadcast. The **broadcast** keyword sets the network type to broadcast.

Step 11 Set the router priority, which helps determine the designated router for a network:

ipv6 ospf priority *number-value*

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf priority 4
```

Valid values range from 0 to 255.

Step 12 Configure OSPFv3 router interconnections to non-broadcast networks:

ipv6 ospf neighbor *ipv6-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*] [**database-filter all out**]

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf neighbor FE80::A8BB:CCFF:FE00:C01
```

Step 13 Specify the time in seconds between LSA retransmissions for adjacencies that belong to the interface:

ipv6 ospf retransmit-interval *seconds*

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf retransmit-interval 8
```

The time must be greater than the expected round-trip delay between any two routers on the attached network. Valid values range from 1 to 65535 seconds. The default is 5 seconds.

Step 14 Set the estimated time in seconds to send a link-state update packet on the interface:

ipv6 ospf transmit-delay *seconds*

Example:

```
ciscoasa(config-if)# interface GigabitEthernet3/2.200
 vlan 200
 nameif outside
 security-level 100
 ip address 10.20.200.30 255.255.255.0 standby 10.20.200.31
 ipv6 address 3001::1/64 standby 3001::8
 ipv6 address 6001::1/64 standby 6001::8
 ipv6 enable
 ospf priority 255
 ipv6 ospf cost 100
 ipv6 ospf 100 area 10 instance 200
ipv6 ospf retransmit-delay 3
```

Valid values range from 1 to 65535 seconds. The default is 1 second.

Configure OSPFv3 Router Parameters

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config)# ipv6 router ospf 10
```

The *process-id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Configure OSPFv3 area parameters:

area

Example:

```
ciscoasa(config-rtr)# area 10
```

Supported parameters include the area ID as a decimal value from 0 to 4294967295 and the area ID in the IP address format of **A.B.C.D**.

Step 3 Set a command to its default value:

default

Example:

```
ciscoasa(config-rtr)# default originate
```

The **originate** parameter distributes the default route.

Step 4 Control distribution of default information:

default-information

Step 5 Define the OSPFv3 route administrative distance based on the route type:

distance

Example:

```
ciscoasa(config-rtr)# distance 200
```

Supported parameters include the administrative distance with values from 1 to 254 and **ospf** for the OSPFv3 distance.

Step 6 Suppress the sending of syslog messages with the **lsa** parameter when the router receives a link-state advertisement (LSA) for Type 6 Multicast OSPF (MOSPF) packets:

ignore

Example:

```
ciscoasa(config-rtr)# ignore lsa
```

Step 7 Configure the router to send a syslog message when an OSPFv3 neighbor goes up or down:

log-adjacency-changes

Example:

```
ciscoasa(config-rtr)# log-adjacency-changes detail
```

With the **detail** parameter, all state changes are logged.

Step 8 Suppress the sending and receiving of routing updates on an interface:

passive-interface [*interface_name*]

Example:

```
ciscoasa(config-rtr)# passive-interface inside
```

The *interface_name* argument specifies the name of the interface on which the OSPFv3 process is running.

Step 9 Configure the redistribution of routes from one routing domain into another:

redistribute {**connected** | **ospf** | **static**}

Where:

- **connected**—Specifies connected routes.
- **ospf**—Specifies OSPFv3 routes.
- **static**—Specifies static routes.

Example:

```
ciscoasa(config-rtr)# redistribute ospf
```

Step 10 Create a fixed router ID for a specified process:

router-id {*A.B.C.D* | **cluster-pool** | **static**}

Where:

A.B.C.D—Specifies the OSPF router ID in IP address format.

cluster-pool—Configures an IP address pool when Individual Interface clustering is configured. For more information about IP address pools used in clustering, see Configure an IP Address Pool for Clustering (OSPFv2 and OSPFv3), on page 21.

Example:

```
ciscoasa(config-rtr)# router-id 10.1.1.1
```

Step 11 Configure IPv6 address summaries with valid values from 0 to 128:

summary-prefix *X:X:X:X::X/*

Example:

```
ciscoasa(config-if)# ipv6 router ospf 1
ciscoasa(config-router)# router-id 192.168.3.3
ciscoasa(config-router)# summary-prefix FECO::/24
```

```
ciscoasa(config-router)# redistribute static
```

The *X:X:X:X::X/* parameter specifies the IPv6 prefix.

Step 12 Adjust routing timers:

timers

The routing timer parameters are the following:

- **lsa**—Specifies OSPFv3 LSA timers.
- **nsf**—Specifies OSPFv3 NSF wait timers.
- **pacing**—Specifies OSPFv3 pacing timers.
- **throttle**—Specifies OSPFv3 throttle timers.

Example:

```
ciscoasa(config)# ipv6 router ospf 10
ciscoasa(config-rtr)# timers throttle spf 6000 12000 14000
```

Configure OSPFv3 Area Parameters

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config)# ipv6 router ospf 1
```

The *process-id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535.

This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Set the summary default cost of an NSSA area or a stub area:

area *area-id* **default-cost** *cost*

Example:

```
ciscoasa(config-rtr)# area 1 default-cost nssa
```

Step 3 Summarize routes that match the address and mask for border routers only:

area *area-id* **range** *ipv6-prefix*/ *prefix-length* [**advertise** | **not advertise**] [**cost** *cost*]

Example:

```
ciscoasa(config-rtr)# area 1 range FE01:1::1/64
```

- The *area-id* argument identifies the area for which routes are to be summarized. The value can be specified as a decimal or an IPv6 prefix.
- The *ipv6-prefix* argument specifies the IPv6 prefix. The *prefix-length* argument specifies the prefix length.
- The **advertise** keyword sets the address range status to advertised and generates a Type 3 summary LSA.
- The **not-advertise** keyword sets the address range status to DoNotAdvertise.
- The Type 3 summary LSA is suppressed, and the component networks remain hidden from other networks.
- The **cost** *cost* keyword-argument pair specifies the metric or cost for the summary route, which is used during OSPF SPF calculations to determine the shortest paths to the destination.
- Valid values range from 0 to 16777215.

Step 4 Specify an NSSA area:

area *area-id* **nssa**

Example:

```
ciscoasa(config-rtr)# area 1 nssa
```

Step 5 Specify a stub area:

area *area-id* **stub**

Example:

```
ciscoasa(config-rtr)# area 1 stub
```

Step 6 Define a virtual link and its parameters:

area *area-id* **virtual-link** *router-id* [**hello-interval** *seconds*] [**retransmit-interval** *seconds*] [**transmit-delay** *seconds*] [**dead-interval** *seconds*] [**ttl-security hops** *hop-count*]

Example:

```
ciscoasa(config-rtr)# area 1 virtual-link 192.168.255.1 hello-interval 5
```

- The *area-id* argument identifies the area for which routes are to be summarized. The **virtual link** keyword specifies the creation of a virtual link neighbor.
- The *router-id* argument specifies the router ID that is associated with the virtual link neighbor.
- Enter the **show ospf** or **show ipv6 ospf** command to display the router ID. There is no default value.
- The **hello-interval** keyword specifies the time in seconds between the hello packets that are sent on an interface. The hello interval is an unsigned integer that is to be advertised in the hello packets. The value must be the same for all routers and access servers that are attached to a common network. Valid values range from 1 to 8192. The default is 10.

- The **retransmit-interval** *seconds* keyword-argument pair specifies the time in seconds between LSA retransmissions for adjacencies that belong to the interface. The retransmit interval is the expected round-trip delay between any two routers on the attached network. The value must be greater than the expected round-trip delay, and can range from 1 to 8192. The default is 5.
- The **transmit-delay** *seconds* keyword-argument pair specifies the estimated time in seconds that is required to send a link-state update packet on the interface. The integer value must be greater than zero. LSAs in the update packet have their own ages incremented by this amount before transmission. The range of values can be from 1 to 8192. The default is 1.
- The **dead-interval** *seconds* keyword-argument pair specifies the time in seconds that hello packets are not seen before a neighbor indicates that the router is down. The dead interval is an unsigned integer. The default is four times the hello interval, or 40 seconds. The value must be the same for all routers and access servers that are attached to a common network. Valid values range from 1 to 8192.
- The **ttl-security hops** keyword configures the time-to-live (TTL) security on a virtual link. The *hop-count* argument value can range from 1 to 254.

Configure OSPFv3 Passive Interfaces

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process_id*

Example:

```
ciscoasa(config-if)# ipv6 router ospf 1
```

The *process_id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Suppress the sending and receiving of routing updates on an interface:

passive-interface [*interface_name*]

Example:

```
ciscoasa(config-rtr)# passive-interface inside
```

The *interface_name* argument specifies the name of the interface on which the OSPFv3 process is running. If the *no interface_name* argument is specified, all of the interfaces in the OSPFv3 process *process_id* are made passive.

Configure OSPFv3 Administrative Distance

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process_id*

Example:

```
ciscoasa(config-if)# ipv6 router ospf 1
```

The *process_id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Set the administrative distance for OSPFv3 routes:

distance [**ospf** {**external** | **inter-area** | **intra-area**}] *distance*

Example:

```
ciscoasa(config-rtr)# distance ospf external 200
```

The **ospf** keyword specifies OSPFv3 routes. The **external** keyword specifies the external Type 5 and Type 7 routes for OSPFv3. The **inter-area** keyword specifies the inter-area routes for OSPVv3. The **intra-area** keyword specifies the intra-area routes for OSPFv3. The *distance* argument specifies the administrative distance, which is an integer from 10 to 254.

Configure OSPFv3 Timers

You can set LSA arrival, LSA pacing, and throttling timers for OSPFv3.

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config-if)# ipv6 router ospf 1
```

The *process-id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Set the minimum interval at which the ASA accepts the same LSA from OSPF neighbors:

timers lsa arrival *milliseconds*

Example:

```
ciscoasa(config-rtr)# timers lsa arrival 2000
```

The *milliseconds* argument specifies the minimum delay in milliseconds that must pass between acceptance of the same LSA arriving from neighbors. The range is from 0 to 6,000,000 milliseconds. The default is 1000 milliseconds.

Step 3 Configure LSA flood packet pacing:

timers pacing flood *milliseconds*

Example:

```
ciscoasa(config-rtr)# timers lsa flood 20
```

The *milliseconds* argument specifies the time in milliseconds at which LSAs in the flooding queue are paced in between updates. The configurable range is from 5 to 100 milliseconds. The default value is 33 milliseconds.

Step 4 Change the interval at which OSPFv3 LSAs are collected into a group and refreshed, checksummed, or aged:

timers pacing lsa-group *seconds*

Example:

```
ciscoasa(config-rtr)# timers pacing lsa-group 300
```

The *seconds* argument specifies the number of seconds in the interval at which LSAs are grouped, refreshed, check summed, or aged. The range is from 10 to 1800 seconds. The default value is 240 seconds.

Step 5 Configure LSA retransmission packet pacing:

timers pacing retransmission *milliseconds*

Example:

```
ciscoasa(config-rtr)# timers pacing retransmission 100
```

The *milliseconds* argument specifies the time in milliseconds at which LSAs in the retransmission queue are paced. The configurable range is from 5 to 200 milliseconds. The default value is 66 milliseconds.

Step 6 Configure OSPFv3 LSA throttling:

timers throttle lsa *milliseconds1 milliseconds2 milliseconds3*

Example:

```
ciscoasa(config-rtr)# timers throttle lsa 500 6000 8000
```

- The *milliseconds1* argument specifies the delay in milliseconds to generate the first occurrence of the LSA. The *milliseconds2* argument specifies the maximum delay in milliseconds to originate the same LSA. The *milliseconds3* argument specifies the minimum delay in milliseconds to originate the same LSA.

- For LSA throttling, if the minimum or maximum time is less than the first occurrence value, then OSPFv3 automatically corrects to the first occurrence value. Similarly, if the maximum delay specified is less than the minimum delay, then OSPFv3 automatically corrects to the minimum delay value.
- For *milliseconds1*, the default value is 0 milliseconds.
- For *milliseconds2* and *milliseconds3*, the default value is 5000 milliseconds.

Step 7 Configure OSPFv3 SPF throttling:

timers throttle spf *milliseconds1 milliseconds2 milliseconds3*

Example:

```
ciscoasa(config-rtr)# timers throttle spf 5000 12000 16000
```

- The *milliseconds1* argument specifies the delay in milliseconds to receive a change to the SPF calculation. The *milliseconds2* argument specifies the delay in milliseconds between the first and second SPF calculations. The *milliseconds3* argument specifies the maximum wait time in milliseconds for SPF calculations.
- For SPF throttling, if *milliseconds2* or *milliseconds3* is less than *milliseconds1*, then OSPFv3 automatically corrects to the *milliseconds1* value. Similarly, if *milliseconds3* is less than *milliseconds2*, then OSPFv3 automatically corrects to the *milliseconds2* value.
- For *milliseconds1*, the default value of SPF throttling is 5000 milliseconds.
- For *milliseconds2* and *milliseconds3*, the default value of SPF throttling is 10000 milliseconds.

Define Static OSPFv3 Neighbors

You need to define static OSPFv3 neighbors to advertise OSPF routes over a point-to-point, non-broadcast network. This feature lets you broadcast OSPFv3 advertisements across an existing VPN connection without having to encapsulate the advertisements in a GRE tunnel.

Before you begin, you must create a static route to the OSPFv3 neighbor. See Configure a Static Route for more information about creating static routes.

Procedure

Step 1 Enable an OSPFv3 routing process and enters IPv6 router configuration mode.

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config)# ipv6 router ospf 1
```

The *process-id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Configure OSPFv3 router interconnections to non-broadcast networks.

ipv6 ospf neighbor *ipv6-address* [**priority** *number*] [**poll-interval** *seconds*] [**cost** *number*] [**database-filter all out**]

Example:

```
ciscoasa(config-if)# interface ethernet0/0 ipv6 ospf neighbor FE80::A8BB:CCFF:FE00:C01
```

Reset OSPFv3 Default Parameters

To return an OSPFv3 parameter to its default value, perform the following steps:

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config-if)# ipv6 router ospf 1
```

The *process_id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Return an optional parameter to its default value:

default [**area** | **auto-cost** | **default-information** | **default-metric** | **discard-route** | **discard-route | distance** | **distribute-list** | **ignore** | **log-adjacency-changes** | **maximum-paths** | **passive-interface** | **redistribute** | **router-id** | **summary-prefix** | **timers**]

Example:

```
ciscoasa(config-rtr)# default metric 5
```

- The **area** keyword specifies the OSPFv3 area parameters. The **auto-cost** keyword specifies the OSPFv3 interface cost according to bandwidth.
- The **default-information** keyword distributes default information. The **default-metric** keyword specifies the metric for a redistributed route
- The **discard-route** keyword enables or disables the discard-route installation. The **distance** keyword specifies the administrative distance.
- The **distribute-list** keyword filters networks in routing updates.
- The **ignore** keyword ignores a specific event. The **log-adjacency-changes** keyword logs changes in the adjacency state.

- The **maximum-paths** keyword forwards packets over multiple paths.
- The **passive-interface** keyword suppresses routing updates on an interface.
- The **redistribute** keyword redistributes IPv6 prefixes from another routing protocol.
- The **router-id** keyword specifies the router ID for the specified routing process.
- The **summary-prefix** keyword specifies the IPv6 summary prefix.
- The **timers** keyword specifies the OSPFv3 timers.

Send Syslog Messages

Configure the router to send a syslog message when an OSPFv3 neighbor goes up or down.

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config-if)# ipv6 router ospf 1
```

The *process-id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Configure the router to send a syslog message when an OSPFv3 neighbor goes up or down:

log-adjacency-changes [**detail**]

Example:

```
ciscoasa(config-rtr)# log-adjacency-changes detail
```

The **detail** keyword sends a syslog message for each state, not only when an OSPFv3 neighbor goes up or down.

Suppress Syslog Messages

To suppress the sending of syslog messages when the route receives unsupported LSA Type 6 multicast OSPF (MOSPF) packets, perform the following steps:

Procedure

Step 1 Enable an OSPFv2 routing process:

router ospf *process_id*

Example:

```
ciscoasa(config-if)# router ospf 1
```

The *process_id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Suppress the sending of syslog messages when the router receives unsupported LSA Type 6 MOSPF packets:

ignore lsa mospf

Example:

```
ciscoasa(config-rtr)# ignore lsa mospf
```

Calculate Summary Route Costs

Procedure

Restore the methods that are used to calculate summary route costs according to RFC 1583:

compatible rfc1583

Example:

```
ciscoasa (config-rtr)# compatible rfc1583
```

Generate a Default External Route into an OSPFv3 Routing Domain

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config-if)# ipv6 router ospf 1
```

The *process-id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Generate a default external route into an OSPFv3 routing domain:

default-information originate [**always**] **metric** *metric-value* [**metric-type** *type-value*] [**route-map** *map-name*]

Example:

```
ciscoasa(config-rtr)# default-information originate always metric 3 metric-type 2
```

- The **always** keyword advertises the default route whether or not the default route exists.
- The **metric** *metric-value* keyword-argument pair specifies the metric used for generating the default route.
- If you do not specify a value using the **default-metric** command, the default value is 10. Valid metric values range from 0 to 16777214.
- The **metric-type** *type-value* keyword-argument pair specifies the external link type that is associated with the default route that is advertised into the OSPFv3 routing domain. Valid values can be one of the following:
  - 1—Type 1 external route
  - 2—Type 2 external route

  The default is the type 2 external route.
- The **route-map** *map-name* keyword-argument pair specifies the routing process that generates the default route if the route map is satisfied.

Configure an IPv6 Summary Prefix

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config-if)# ipv6 router ospf 1
```

The *process_id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Configure an IPv6 summary prefix:

summary-prefix *prefix* [**not-advertise | tag** *tag-value*]

Example:

```
ciscoasa(config-if)# ipv6 router ospf 1
ciscoasa(config-rtr)# router-id 192.168.3.3
ciscoasa(config-rtr)# summary-prefix FECO::/24
ciscoasa(config-rtr)# redistribute static
```

The *prefix* argument is the IPv6 route prefix for the destination. The **not-advertise** keyword suppresses routes that match the specified prefix and mask pair. This keyword applies to OSPFv3 only. The **tag** *tag-value* keyword-argument pair specifies the tag value that can be used as a match value for controlling redistribution through route maps. This keyword applies to OSPFv3 only.

Redistribute IPv6 Routes

Procedure

Step 1 Enable an OSPFv3 routing process:

ipv6 router ospf *process-id*

Example:

```
ciscoasa(config-if)# ipv6 router ospf 1
```

The *process-id* argument is an internally used identifier for this routing process, is locally assigned, and can be any positive integer from 1 to 65535. This ID does not have to match the ID on any other device; it is for internal administrative use only. You can use a maximum of two processes.

Step 2 Redistribute IPv6 routes from one OSPFv3 process into another:

redistribute *source-protocol* [*process-id*] [**include-connected** {[**level-1** | **level-2**}] [*as-number*] [**metric** [*metric-value* | **transparent**}] **[metric-type** *type-value*] [**match** {**external** [**1**|**2**] | **internal** | **nssa-external** [**1**|**2**]}] [**tag** *tag-value*] [**route-map** *map-tag*]

Example:

```
ciscoasa(config-rtr)# redistribute connected 5 type-1
```

- The *source-protocol* argument specifies the source protocol from which routes are being redistributed, which can be static, connected, or OSPFv3.
- The *process-id* argument is the number that is assigned administratively when the OSPFv3 routing process is enabled.
- The **include-connected** keyword allows the target protocol to redistribute routes learned by the source protocol and connected prefixes on those interfaces over which the source protocol is running.

- The **level-1** keyword specifies that for Intermediate System-to-Intermediate System (IS-IS), Level 1 routes are redistributed into other IP routing protocols independently.
- The **level-1-2** keyword specifies that for IS-IS, both Level 1 and Level 2 routes are redistributed into other IP routing protocols.
- The **level-2** keyword specifies that for IS-IS, Level 2 routes are redistributed into other IP routing protocols independently.
- For the **metric** *metric-value* keyword-argument pair, when redistributing routes from one OSPFv3 process into another OSPFv3 process on the same router, the metric is carried through from one process to the other if no metric value is specified. When redistributing other processes into an OSPFv3 process, the default metric is 20 when no metric value is specified.
- The **metric transparent** keyword causes RIP to use the routing table metric for redistributed routes as the RIP metric.
- The **metric-type** *type-value* keyword-argument pair specifies the external link type that is associated with the default route that is advertised into the OSPFv3 routing domain. Valid values can be one of the following: 1 for a Type 1 external route or 2 for a Type 2 external route. If no value is specified for the **metric-type** keyword, the ASA adopts a Type 2 external route. For IS-IS, the link type can be one of the following: internal for an IS-IS metric that is less than 63 or external for an IS-IS metric that is greater than 64 and less than 128. The default is internal.
- The **match** keyword redistributes routes into other routing domains and is used with one of the following options: **external** [**1**|**2**] for routes that are external to the autonomous system, but are imported into OSPFv3 as Type 1 or Type 2 external routes; **internal** for routes that are internal to a specific autonomous system; **nssa-external** [**1**|**2**] for routes that are external to the autonomous system, but are imported into OSPFv3 in an NSSA for IPv6 as Type 1 or Type 2 external routes.
- The **tag** *tag-value* keyword-argument pair specifies the 32-bit decimal value that is attached to each external route, which may be used to communicate information between ASBRs. If none is specified, then the remote autonomous system number is used for routes from BGP and EGP. For other protocols, zero is used. Valid values range from 0 to 4294967295.
- The **route-map** keyword specifies the route map to check for filtering the importing of routes from the source routing protocol to the current routing protocol. If this keyword is not specified, all routes are redistributed. If this keyword is specified, but no route map tags are listed, no routes are imported. The *map-tag* argument identifies a configured route map.

Configure Graceful Restart

The ASA may experience some known failure situations, that should not affect packet forwarding across the switching platform. The Non-Stop Forwarding (NSF) capability allows data forwarding to continue along known routes, while the routing protocol information is being restored. This capability is useful when there is a component failure (i.e., active unit crash with standby unit taking over in failover (HA) mode, control unit crash with data unit elected as new control unit in cluster mode), or when there is a scheduled hitless software upgrade.

Graceful restart is supported on both OSPFv2 and OSPFv3. You can configure graceful restart on OSPFv2 by using either using NSF Cisco (RFC 4811 and RFC 4812) or NSF IETF (RFC 3623). You can configure graceful restart on OSPFv3 using graceful-restart (RFC 5187).

Configuring the NSF graceful-restart feature involves two steps; configuring capabilities and configuring a device as NSF-capable or NSF-aware. A NSF-capable device can indicate its own restart activities to neighbors and a NSF-aware device can help a restarting neighbor.

A device can be configured as NSF-capable or NSF-aware, depending on some conditions:

- A device can be configured as NSF-aware irrespective of the mode in which it is.
- A device has to be in either Failover or Spanned Etherchannel (L2) cluster mode to be configured as NSF-capable.
- For a device to be either NSF-aware or NSF-capable, it should be configured with the capability of handling opaque Link State Advertisements (LSAs)/ Link Local Signaling (LLS) block as required.



Note When fast hellos are configured for OSPFv2, graceful restart does not occur when the active unit reloads and the standby unit becomes active. This is because the time taken for the role change is more than the configured dead interval.

Configure Capabilities

The Cisco NSF Graceful Restart mechanism depends on the LLS capability as it sends an LLS block with the RS-bit set in the Hello packet, to indicate the restart activity. The IETF NSF mechanism depends on the opaque LSA capability as it sends opaque-LSAs of type-9 to indicate the restart activity. To configure capabilities enter the following commands:

Procedure

Step 1 Create an OSPF routing process and enters router configuration mode for the OSPF process that you want to redistribute:

router ospf *process_id*

Example:

```
ciscoasa(config)# router ospf 2
```

The process_id argument is an internally used identifier for this routing process and can be any positive integer. This ID does not have to match the ID on any other device; it is for internal use only. You can use a maximum of two processes.

Step 2 Enable the use of LLS data block or opaque LSAs to enable NSF:

capability {lls|opaque}

The lls keyword is used to enable LLS capability for Cisco NSF Graceful Restart mechanism.

The opaque keyword is used to enable opaque LSA capability for IETF NSF Graceful Restart mechanism.

Configuring Graceful Restart for OSPFv2

There are two graceful restart mechanisms for OSPFv2, Cisco NSF and IETF NSF. Only one of these graceful restart mechanisms can be configured at a time for an ospf instance. An NSF-aware device can be configured as both Cisco NSF helper and IETF NSF helper but a NSF-capable device can be configured in either Cisco NSF or IETF NSF mode at a time for an ospf instance.

Configure Cisco NSF Graceful Restart for OSPFv2

Configure Cisco NSF Graceful Restart for OSPFv2, for a NSF-capable or NSF-aware device.

Procedure

Step 1 Enable Cisco NSF on a NSF-capable device:

nsf cisco [enforce global]

Example:

```
ciscoasa(config-router)# nsf cisco
```

The enforce global keyword cancels NSF restart when non-NSF-aware neighbor devices are detected.

Step 2 Enable Cisco NSF helper mode on NSF-aware device:

capability {lls|opaque}

Example:

```
ciscoasa(config-router)# capability lls
```

This command is enabled by default. Using the no form of the command disables it.

Configure IETF NSF Graceful Restart for OSPFv2

Configure IETF NSF Graceful Restart for OSPFv2, for a NSF-capable or NSF-aware device.

Procedure

Step 1 Enable IETF NSF on a NSF-capable device:

nsf ietf [restart-interval *seconds***]**

Example:

```
ciscoasa(config-router)# nsf ietf restart-interval 80
```

You can specify the length of the graceful restart interval, in seconds. Valid values are from 1 to 1800 seconds. The default value is 120 seconds.

Graceful restart might be terminated when restart interval is configured with a value less than the time taken for the adjacency to come up. For example, a restart interval below 30 seconds, is not supported.

Step 2 Enable IETF NSF helper mode on NSF-aware device:

nsf ietf helper [strict-lsa-checking]

Example:

```
ciscoasa(config-router)# nsf ietf helper
```

The strict-LSA-checking keyword indicates that the helper router will terminate the process of the restarting router if it detects that there is a change to a LSA that would be flooded to the restarting router, or if there is a changed LSA on the retransmission list of the restarting router when the graceful restart process is initiated.

This command is enabled by default. Using the no form of the command disables it.

Configuring Graceful Restart for OSPFv3

Configuring the NSF graceful-restart feature for OSPFv3 involves two steps; configuring a device to be NSF-capable and then configuring a device to be NSF-aware.

Procedure

Step 1 Enable IPv6 processing on an interface that has not been configured with an explicit IPv6 address:

interface physical_interface ipv6 enable

Example:

```
ciscoasa(config)# interface ethernet 0/0
ciscoasa(config-if)# ipv6 enable
```

The physical_interface argument identifies the interface that participates in OSPFv3 NSF.

Step 2 Enable graceful-restart for OSPFv3 on a NSF-capable device:

graceful-restart [restart interval seconds]

Example:

```
ciscoasa(config-router)# graceful-restart restart interval 80
```

The restart interval seconds specifies the length of the graceful restart interval, in seconds. Valid values are from 1 to 1800 seconds. The default value is 120 seconds.

Graceful restart might be terminated when restart interval is configured with a value less than the time taken for the adjacency to come up.For example, a restart interval below 30 seconds, is not supported.

Step 3 Enable graceful-restart for OSPFv3 on a NSF-aware device:

graceful-restart helper [strict-lsa-checking]

Example:

```
ciscoasa(config-router)# graceful-restart helper strict-lsa-checking
```

The strict-LSA-checking keyword indicates that the helper router will terminate the process of the restarting router if it detects that there is a change to a LSA that would be flooded to the restarting router, or if there is a changed LSA on the retransmission list of the restarting router when the graceful restart process is initiated.

The graceful-restart helper mode is enabled by default.

Remove the OSPFv2 Configuration

Remove the OSPFv2 configuration.

Procedure

Remove the entire OSPFv2 configuration that you have enabled.

clear configure router ospf *pid*

Example:

```
ciscoasa(config)# clear configure router ospf 1000
```

After the configuration is cleared, you must reconfigure OSPF using the **router ospf** command.

Remove the OSPFv3 Configuration

Remove the OSPFv3 configuration.

Procedure

Remove the entire OSPFv3 configuration that you have enabled:

clear configure ipv6 router ospf *process-id*

Example:

```
ciscoasa(config)# clear configure ipv6 router ospf 1000
```

After the configuration is cleared, you must reconfigure OSPFv3 using the **ipv6 router ospf** command.

Example for OSPFv2

The following example shows how to enable and configure OSPFv2 with various optional processes:

1. To enable OSPFv2, enter the following commands:

```
ciscoasa(config)# router ospf 2
ciscoasa(config-rtr)# network 10.0.0.0 255.0.0.0 area 0
```

2. (Optional) To redistribute routes from one OSPFv2 process to another OSPFv2 process, enter the following commands:

```
ciscoasa(config)# route-map 1-to-2 permit
ciscoasa(config-route-map)# match metric 1
ciscoasa(config-route-map)# set metric 5
ciscoasa(config-route-map)# set metric-type type-1
ciscoasa(config-route-map)# router ospf 2
ciscoasa(config-rtr)# redistribute ospf 1 route-map 1-to-2
```

3. (Optional) To configure OSPFv2 interface parameters, enter the following commands:

```
ciscoasa(config)# router ospf 2
ciscoasa(config-rtr)# network 10.0.0.0 255.0.0.0 area 0
ciscoasa(config-rtr)# interface inside
ciscoasa(config-interface)# ospf cost 20
ciscoasa(config-interface)# ospf retransmit-interval 15
ciscoasa(config-interface)# ospf transmit-delay 10
ciscoasa(config-interface)# ospf priority 20
ciscoasa(config-interface)# ospf hello-interval 10
ciscoasa(config-interface)# ospf dead-interval 40
ciscoasa(config-interface)# ospf authentication-key cisco
ciscoasa(config-interface)# ospf message-digest-key 1 md5 cisco
ciscoasa(config-interface)# ospf authentication message-digest
```

4. (Optional) To configure OSPFv2 area parameters, enter the following commands:

```
ciscoasa(config)# router ospf 2
ciscoasa(config-rtr)# area 0 authentication
ciscoasa(config-rtr)# area 0 authentication message-digest
ciscoasa(config-rtr)# area 17 stub
ciscoasa(config-rtr)# area 17 default-cost 20
```

5. (Optional) To configure the route calculation timers and show the log neighbor up and down messages, enter the following commands:

```
ciscoasa(config-rtr)# timers spf 10 120
ciscoasa(config-rtr)# log-adj-changes [detail]
```

6. (Optional) To show current OSPFv2 configuration settings, enter the **show ospf** command.

   The following is sample output from the **show ospf** command:

```
ciscoasa(config)# show ospf

Routing Process "ospf 2" with ID 10.1.89.2 and Domain ID 0.0.0.2
Supports only single TOS(TOS0) routes
Supports opaque LSA
SPF schedule delay 5 secs, Hold time between two SPFs 10 secs
Minimum LSA interval 5 secs. Minimum LSA arrival 1 secs
Number of external LSA 5. Checksum Sum 0x 26da6
Number of opaque AS LSA 0. Checksum Sum 0x     0
Number of DCbitless external and opaque AS LSA 0
Number of DoNotAge external and opaque AS LSA 0
Number of areas in this router is 1. 1 normal 0 stub 0 nssa
External flood list length 0
    Area BACKBONE(0)
        Number of interfaces in this area is 1
        Area has no authentication
        SPF algorithm executed 2 times
        Area ranges are
        Number of LSA 5. Checksum Sum 0x 209a3
        Number of opaque link LSA 0. Checksum Sum 0x     0
        Number of DCbitless LSA 0
        Number of indication LSA 0
        Number of DoNotAge LSA 0
        Flood list length 0
```

7. To clear the OSPFv2 configuration, enter the following command:

```
ciscoasa(config)# clear configure router ospf pid
```

Examples for OSPFv3

The following example shows how to enable and configure OSPFv3 at the interface level:

```
ciscoasa (config)# interface GigabitEthernet3/1
ciscoasa (config-if)# ipv6 enable
ciscoasa (config-if)# ipv6 ospf 1 area 1
```

The following is sample output from the **show running-config ipv6** command:

```
ciscoasa (config)# show running-config ipv6
ipv6 router ospf 1
 log-adjacency-changes
```

The following is sample output from the **show running-config interface** command:

```
ciscoasa (config-if)# show running-config interface GigabitEthernet3/1
interface GigabitEthernet3/1
 nameif fda
 security-level 100
 ip address 1.1.11.1 255.255.255.0 standby 1.1.11.2
 ipv6 address 9098::10/64 standby 9098::11
```

```
ipv6 enable
ipv6 ospf 1 area 1
```

The following examples show how to configure OSPFv3-specific interfaces:

```
ciscoasa (config)# interface GigabitEthernet3/1
ciscoasa (config-if)# nameif fda
ciscoasa (config-if)# security-level 100
ciscoasa (config-if)# ip address 10.1.11.1 255.255.255.0 standby 10.1.11.2
ciscoasa (config-if)# ipv6 address 9098::10/64 standby 9098::11
ciscoasa (config-if)# ipv6 enable
ciscoasa (config-if)# ipv6 ospf cost 900
ciscoasa (config-if)# ipv6 ospf hello-interval 20
ciscoasa (config-if)# ipv6 ospf network broadcast
ciscoasa (config-if)# ipv6 ospf database-filter all out
ciscoasa (config-if)# ipv6 ospf flood-reduction
ciscoasa (config-if)# ipv6 ospf mtu-ignore
ciscoasa (config-if)# ipv6 ospf 1 area 1 instance 100
ciscoasa (config-if)# ipv6 ospf encryption ipsec spi 890 esp null md5
12345678901234567890123456789012

ciscoasa (config)# ipv6 router ospf 1
ciscoasa (config)# area 1 nssa
ciscoasa (config)# distance ospf intra-area 190 inter-area 100 external 100
ciscoasa (config)# timers lsa arrival 900
ciscoasa (config)# timers pacing flood 100
ciscoasa (config)# timers throttle lsa 900 900 900
ciscoasa (config)# passive-interface fda
ciscoasa (config)# log-adjacency-changes
ciscoasa (config)# redistribute connected metric 100 metric-type 1 tag 700
```

For an example of how to configure an OSPFv3 virtual link, see the following URL:

http://www.cisco.com/en/US/tech/tk365/technologies_configuration_example09186a0080b8fd06.shtml

Monitoring OSPF

You can display specific statistics such as the contents of IP routing tables, caches, and databases. You can also use the information provided to determine resource utilization and solve network problems. You can also display information about node reachability and discover the routing path that your device packets are taking through the network.

To monitor or display various OSPFv2 routing statistics, enter one of the following commands:

Command	Purpose
show ospf [*process-id* [*area-id*]]	Displays general information about OSPFv2 routing processes.
show ospf border-routers	Displays the internal OSPFv2 routing table entries to the ABR and ASBR.
show ospf [*process-id* [*area-id*]] database	Displays lists of information related to the OSPFv2 database for a specific router.

Command	Purpose
show ospf flood-list *if-name*	Displays a list of LSAs waiting to be flooded over an interface (to observe OSPF v2packet pacing). OSPFv2 update packets are automatically paced so they are not sent less than 33 milliseconds apart. Without pacing, some update packets could get lost in situations where the link is slow, a neighbor could not receive the updates quickly enough, or the router could run out of buffer space. For example, without pacing, packets might be dropped if either of the following topologies exist: • A fast router is connected to a slower router over a point-to-point link. • During flooding, several neighbors send updates to a single router at the same time. Pacing is also used between resends to increase efficiency and minimize lost retransmissions. You also can display the LSAs waiting to be sent out of an interface. Pacing enables OSPFv2 update and retransmission packets to be sent more efficiently. There are no configuration tasks for this feature; it occurs automatically.
show ospf interface [*if_name*]	Displays OSPFv2-related interface information.
show ospf neighbor [*interface-name*] [*neighbor-id*] [**detail**]	Displays OSPFv2 neighbor information on a per-interface basis.
show ospf request-list *neighbor if_name*	Displays a list of all LSAs requested by a router.
show ospf retransmission-list *neighbor if_name*	Displays a list of all LSAs waiting to be resent.
show ospf [*process-id*] summary-address	Displays a list of all summary address redistribution information configured under an OSPFv2 process.
show ospf [*process-id*] **traffic**	Displays a list of different types of packets being sent or received by a specific OSPFv2 instance.
show ospf [*process-id*] virtual-links	Displays OSPFv2-related virtual links information.
show route cluster	Displays additional OSPFv2 route synchronization information in clustering.

To monitor or display various OSPFv3 routing statistics, enter one of the following commands:

Command	Purpose
show ipv6 ospf [*process-id* [*area-id*]]	Displays general information about OSPFv3 routing processes.

Command	Purpose
show ipv6 ospf [*process-id*] **border-routers**	Displays the internal OSPFv3 routing table entries to the ABR and ASBR.
show ipv6 ospf [*process-id* [*area-id*]] database [**external** | **inter-area prefix** | **inter-area-router** | **network** | **nssa-external** | **router** | **area** | **as** | **ref-lsa** | [*destination-router-id*] [**prefix** *ipv6-prefix*] [*link-state-id*]] [**link** [**interface** *interface-name*] [**adv-router** *router-id*] | **self-originate**] [**internal**] [**database-summary**]	Displays lists of information related to the OSPFv3 database for a specific router.
show ipv6 ospf [*process-id* [*area-id*]] **events**	Displays OSPFv3 event information.
show ipv6 ospf [*process-id*] [*area-id*] flood-list *interface-type interface-number*	Displays a list of LSAs waiting to be flooded over an interface (to observe OSPFv3 packet pacing). OSPFv3 update packets are automatically paced so they are not sent less than 33 milliseconds apart. Without pacing, some update packets could get lost in situations where the link is slow, a neighbor could not receive the updates quickly enough, or the router could run out of buffer space. For example, without pacing, packets might be dropped if either of the following topologies exist: • A fast router is connected to a slower router over a point-to-point link. • During flooding, several neighbors send updates to a single router at the same time. Pacing is also used between retransmissions to increase efficiency and minimize lost retransmissions. You also can display the LSAs waiting to be sent out of an interface. Pacing enables OSPFv3 update and retransmission packets to be sent more efficiently. There are no configuration tasks for this feature; it occurs automatically.
show ipv6 ospf [*process-id*] [*area-id*] **interface** [*type number*] [**brief**]	Displays OSPFv3-related interface information.
show ipv6 ospf neighbor [*process-id*] [*area-id*] [*interface-type interface-number*] [*neighbor-id*] [**detail**]	Displays OSPFv3 neighbor information on a per-interface basis.
show ipv6 ospf [*process-id*] [*area-id*] **request-list** [*neighbor*] [*interface*] [*interface-neighbor*]	Displays a list of all LSAs requested by a router.
show ipv6 ospf [*process-id*] [*area-id*] retransmission-list [*neighbor*] [*interface*] [*interface-neighbor*]	Displays a list of all LSAs waiting to be resent.
show ipv6 ospf statistic [*process-id*] [**detail**]	Displays various OSPFv3 statistics.
show ipv6 ospf [*process-id*] summary-prefix	Displays a list of all summary address redistribution information configured under an OSPFv3 process.

Command	Purpose
show ipv6 ospf [*process-id*] **timers** [**lsa-group** | **rate-limit**]	Displays OSPFv3 timers information.
show ipv6 ospf [*process-id*] **traffic** [*interface_name*]	Displays OSPFv3 traffic-related statistics.
show ipv6 ospf virtual-links	Displays OSPFv3-related virtual links information.
show ipv6 route cluster [**failover**] [**cluster**] [**interface**] [**ospf**] [**summary**]	Displays the IPv6 routing table sequence number, IPv6 reconvergence timer status, and IPv6 routing entries sequence number in a cluster.

History for OSPF

Table 1: Feature History for OSPF

Feature Name	Platform Releases	Feature Information
OSPF Support	7.0(1)	Support was added for route data, authentication, and redistribution and monitoring of routing information using the Open Shortest Path First (OSPF) routing protocol. We introduced the following command: **route ospf**
Dynamic Routing in Multiple Context Mode	9.0(1)	OSPFv2 routing is supported in multiple context mode.
Clustering	9.0(1)	For OSPFv2 and OSPFv3, bulk synchronization, route synchronization, and Spanned EtherChannel load balancing are supported in the clustering environment. We introduced or modified the following commands: **show route cluster**, **show ipv6 route cluster**, **debug route cluster, router-id cluster-pool**.

Feature Name	Platform Releases	Feature Information
OSPFv3 Support for IPv6	9.0(1)	OSPFv3 routing is supported for IPv6. We introduced or modified the following commands: **ipv6 ospf**, **ipv6 ospf area**, **ipv6 ospf cost, ipv6 ospf database-filter all out, ipv6 ospf dead-interval, ipv6 ospf encryption, ipv6 ospf hello-interval, ipv6 ospf mtu-ignore, ipv6 ospf neighbor, ipv6 ospf network, ipv6 ospf flood-reduction, ipv6 ospf priority, ipv6 ospf retransmit-interval, ipv6 ospf transmit-delay, ipv6 router ospf, ipv6 router ospf area, ipv6 router ospf default, ipv6 router ospf default-information, ipv6 router ospf distance, ipv6 router ospf exit, ipv6 router ospf ignore, ipv6 router ospf log-adjacency-changes, ipv6 router ospf no, ipv6 router ospf passive-interface, ipv6 router ospf redistribute, ipv6 router ospf router-id, ipv6 router ospf summary-prefix, ipv6 router ospf timers, area encryption, area range, area stub, area nssa, area virtual-link, default, default-information originate, distance, ignore lsa mospf, log-adjacency-changes, redistribute, router-id, summary-prefix, timers lsa arrival, timers pacing flood, timers pacing lsa-group, timers pacing retransmission, timers throttle, show ipv6 ospf, show ipv6 ospf border-routers, show ipv6 ospf database, show ipv6 ospf events, show ipv6 ospf flood-list, show ipv6 ospf graceful-restart, show ipv6 ospf interface, show ipv6 ospf neighbor, show ipv6 ospf request-list, show ipv6 ospf retransmission-list, show ipv6 ospf statistic, show ipv6 ospf summary-prefix, show ipv6 ospf timers, show ipv6 ospf traffic, show ipv6 ospf virtual-links, show ospf, show running-config ipv6 router, clear ipv6 ospf, clear configure ipv6 router, debug ospfv3, ipv6 ospf neighbor**.

Feature Name	Platform Releases	Feature Information
OSPF support for Fast Hellos	9.2(1)	OSPF supports the Fast Hello Packets feature, resulting in a configuration that results in faster convergence in an OSPF network. We modified the following command: **ospf dead-interval**
Timers	9.2(1)	New OSPF timers were added; old ones were deprecated. We introduced the following commands: timers lsa arrival, timers pacing, timers throttle We removed the following commands: Timers spf, timers lsa-grouping-pacing
Route filtering using access-list	9.2(1)	Route filtering using ACL is now supported. We introduced the following command: distribute-list
OSPF Monitoring enhancements	9.2(1)	Additional OSPF monitoring information was added. We modified the following commands: show ospf events, show ospf rib, show ospf statistics, show ospf border-routers [detail], show ospf interface brief
OSPF redistribute BGP	9.2(1)	OSPF redistribution feature was added. We added the following command: redistribute bgp
OSPF Support for Non-Stop Forwarding (NSF)	9.3(1)	OSPFv2 and OSPFv3 support for NSF was added. We added the following commands: capability, nsf cisco, nsf cisco helper, nsf ietf, nsf ietf helper, nsf ietf helper strict-lsa-checking, graceful-restart, graceful-restart helper, graceful-restart helper strict-lsa-checking

Feature Name	Platform Releases	Feature Information
		NSF wait timer was added. We added a new command for setting the timer for the NSF restart interval. This command was introduced to ensure the wait interval is not longer that the router dead interval. We introduced the following command: **timers nsf wait <seconds>**