# ArubaOS-Switch IPv6 Configuration Guide for KA/KB.16.04

a Hewlett Packard
Enterprise company

Part Number: 5200-3117
Published: July 2017
Edition: 1

## Notices

The information contained herein is subject to change without notice. The only warranties for Hewlett Packard Enterprise products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Hewlett Packard Enterprise shall not be liable for technical or editorial errors or omissions contained herein.

Confidential computer software. Valid license from Hewlett Packard Enterprise required for possession, use, or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Links to third-party websites take you outside the Hewlett Packard Enterprise website. Hewlett Packard Enterprise has no control over and is not responsible for information outside the Hewlett Packard Enterprise website.

## Acknowledgments

Intel®, Itanium®, Pentium®, Intel Inside®, and the Intel Inside logo are trademarks of Intel Corporation in the United States and other countries.

Microsoft® and Windows® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Adobe® and Acrobat® are trademarks of Adobe Systems Incorporated.

Java® and Oracle® are registered trademarks of Oracle and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

# Chapter 1 About this guide

This guide provides information on the IPv6 protocol information that are supported on the switch.

## Applicable products

This guide applies to these products:

Aruba 3800 Switch Series (J9573A, J9574A, J9575A, J9576A, J9584A)

Aruba 3810 Switch Series (JL071A, JL072A, JL073A, JL074A, JL075A, JL076A)

Aruba 5400R zl2 Switch Series (J9821A, J9822A, J9850A, J9851A, JL001A, JL002A, JL003A, JL095A)

## Switch prompts used in this guide

Examples in this guide are representative and may not match your particular switch/environment. The following table explains the types of command prompts that may be used in examples, along with information on what each prompt indicates.

| Prompt | Explanation |
|---|---|
| `switch#` | `#` indicates manager context (authority). |
| `switch>` | `>` indicates operator context (authority). |
| `switch(config)#` | `(config)` indicates the config context. |
| `switch(vlan-`*x*`)#` | `(vlan-`*x*`)` indicates the vlan context of config, where *x* represents the VLAN ID. For example: `switch(vlan-128)#`. |
| `switch(eth-`*x*`)#` | `(eth-`*x*`)` indicates the interface context of config, where `x` represents the interface. For example: `switch(eth-48)#`. |
| `switch-Stack#` | `Stack` indicates stacking is enabled. |
| `switch-Stack(config)#` | `Stack(config)` indicates the config context while stacking is enabled. |
| `switch-Stack(stacking)#` | `Stack(stacking)` indicates the stacking context of config while stacking is enabled. |
| `switch-Stack(vlan-`*x*`)#` | `Stack(vlan-`*x*`)` indicates the vlan context of config while stacking is enabled, where *x* represents the VLAN ID. For example: `switch-Stack(vlan-128)#`. |
| `switch-Stack(eth-`*x*`/`*y*`)#` | `Stack(eth-`*x*`/`*y*`)` indicates the interface context of config, in the form `(eth-`*<member-in-stack>*`/`*<interface>*`)`. For example: `switch(eth-1/48)#` |

This chapter provides the following:

- General steps for IPv6 configuration.
- IPv6 command syntax descriptions, including `show` commands.

Most IPv6 configuration commands are applied per-VLAN. The exceptions are ICMP, ND, and the (optional) authorized-managers features, which are configured at the global configuration level. (ICMP and ND for IPv6 are enabled with default values when IPv6 is first enabled and can either be left in their default settings or reconfigured, as needed.)

NOTE

Beginning with software release K.13.01, the switch can operate in dual-stack mode, where IPv4 and IPv6 run concurrently on a given VLAN.

# Configuring IPv6

This section provides an overview of the general configuration steps for enabling IPv6 on a given VLAN and can be enabled by any one of several commands. The following steps provide a suggested progression for getting started.

**Procedure**

1. If IPv6 DHCP service is available, enable IPv6 DHCP on the VLAN. If IPv6 is not already enabled on the VLAN, enabling DHCPv6 also enables IPv6 and automatically configures a link-local address using the EUI-64 format.

   NOTE

   The ICMP and Neighbor Discovery (ND) parameters are set to default values at the global configuration level, are satisfactory for many applications, and generally do not need adjustment when you are first configuring IPv6 on the switch.

   In the default configuration, IPv6 is disabled on all VLANs.

2. If IPv6 DHCP service is not enabled on the VLAN, do either of the following:
   - Enable IPv6 on the VLAN. This automatically configures a link-local address with an EUI-64 interface identifier.
   - Statically configure a unicast IPv6 address on the VLAN. This enables IPv6 on the VLAN and, if you configure anything other than a link-local address, the link-local address is automatically configured with an EUI-64 interface identifier.
3. If an IPv6 router is connected on the VLAN, then enable IPv6 address autoconfiguration to configure automatically global unicast addresses with prefixes included in advertisements received from the router. The interface identifier used in addresses configured by this method will be the same as the interface identifier in the current link-local address.
4. If needed, statically configure IPv6 unicast addressing on the VLAN interface as needed. This can include statically replacing the automatically generated link-local address.

# Commands for configuring IPv6 address

In the default configuration on a VLAN, any of the following commands enables IPv6 and creates a link-local address. Thus, while any of these methods is configured on a VLAN, IPv6 remains enabled and a link-local address is present.

**`ipv6 enable`**

(See **ipv6 enable** on page 15.)

**`ipv6 address autoconfig`**

(See **Enable auto configuration of a global unicast address and a default router identity on a VLAN** on page 16.)

**`ipv6 address dhcp full [rapid-commit]`**

(See **ipv6 address dhcp full** on page 17.)

**`ipv6 address fe80:0:0:0: <interface-identifier> link-local`**

(See **ipv6 address link-local** on page 17.)

**`ipv6 address <prefix:interface-identifier>`**

(See **Commands to statically configure a global unicast address** on page 18.)

# ipv6 enable

### Syntax

`ipv6 enable`

`no ipv6 enable`

### Description

This command enables IPv6 on the VLAN and automatically configures the VLAN's link-local unicast address with a 64-bit EUI-64 interface identifier generated from the VLAN MAC address if IPv6 has not already been enabled on a VLAN by another IPv6 command option.

If no other IPv6-enabling command is configured on the VLAN, the `no` form of the command disables IPv6 on the VLAN. See **Commands to disable IPv6 on a VLAN** on page 27.

### Restrictions

Only one link-local IPv6 address is enabled on the VLAN interface. Subsequent static or DHCP configuration of another link-local address overwrites the existing link-local address.

### Usage

With IPv6 enabled, the VLAN uses received RAs to designate the default IPv6 router.

After verification of uniqueness by DAD, a link-local IPv6 address assigned automatically is set to the `preferred` status, with a "permanent" lifetime.

Default: Disabled

# Commands to view currently configured IPv6 unicast addresses

To view the current IPv6 enable setting and any statically configured IPv6 addresses per-VLAN, use `show run`.

To view all currently configured IPv6 unicast addresses, use the following commands:

**`show ipv6`**

(Lists IPv6 addresses for all VLANs configured on the switch.)

**`show ipv6 vlan <vid>`**

(Lists IPv6 addresses configured on the VLAN.)

# Enable auto configuration of a global unicast address and a default router identity on a VLAN

**More Information**

**Router access and default router selection** on page 30

## ipv6 address autoconfig

**Syntax**

`ipv6 address autoconfig`

`no ipv6 address autoconfig`

**Description**

Implements unicast address autoconfiguration as follows:

- If IPv6 is not already enabled on the VLAN, enables IPv6 and generates a link-local (EUI-64) address.
- Generates router solicitations (RSs) on the VLAN.
- If an RA is received on the VLAN, the switch uses the route prefix in the RA to configure a global unicast address. The interface identifier for this address is the same as the interface identifier used in the current link-local address at the time the RA is received. This can be either a statically configured or the (automatic) EUI-64 interface identifier, depending on how the link-local address was configured.If an RA is not received on the VLAN after autoconfig is enabled, a link-local address is present, but no global unicast addresses are autoconfigured.

NOTE

If a link-local address is already configured on the VLAN, a later, autoconfigured global unicast address, use the same interface identifier as the link-local address.

Default: Disabled.

The `no` form of the command produces different results, depending on how IPv6 is configured on the VLAN: If IPv6 was enabled only by the `autoconfig` command, deleting this command disables IPv6 on the VLAN.

## Commands to view current IPv6 autoconfiguration settings

To view the current IPv6 autoconfiguration settings per-VLAN or per-tunnel, use `show run`.

To view all currently configured IPv6 unicast addresses, use the following commands:

**`show ipv6`**

Lists IPv6 addresses for all VLANs configured on the switch.

**`show ipv6 vlan`** ***`vid`***

Lists IPv6 addresses configured on the VLAN.

**`show ipv6 tunnel`** ***`tunnel`***

Lists IPv6 addresses configured on the tunnel.

## Default IPv6 gateway

Instead of using static or DHCPv6 configuration, a default IPv6 gateway for an interface (VLAN) is determined from the default router list of reachable or probably reachable routers the switch detects from periodic multicast RAs received on the interface. For a given interface, there can be multiple default gateways, with different nodes on the link using different gateways. If the switch does not detect any IPv6 routers that are reachable from a given interface, it assumes (for that interface) that it can reach only the other devices connected to the interface.

NOTE

In IPv6 for the switches covered in this guide, the default route cannot be statically configured. Also, DHCPv6 does not include default route configuration.)

# ipv6 address dhcp full

Enabling the DHCPv6 option on a VLAN enables the switch to obtain a global unicast address. It also enables the switch to obtain additional information such as an NTP server address or a DNS server address that can be used by the switch.

**Syntax**

```
ipv6 address dhcp full [ rapid-commit ]
```

```
no ipv6 address dhcp full [ rapid-commit ]
```

**Description**

Configures DHCPv6 on a VLAN, which initiates transmission of DHCPv6 requests for service. If IPv6 is not already enabled on the VLAN by the `ipv6 enable` command, this option enables IPv6 and causes the switch to autoconfigure a link-local unicast address with an EUI-64 interface identifier.

If no other IPv6-enabling command is configured on the VLAN, the `no` form of the command removes the DHCPv6 option from the configuration and disables IPv6 on the VLAN. See **Commands to disable IPv6 on a VLAN** on page 27.

**Options**

`[ rapid-commit ]`

Expedites DHCP configuration by using a two-message exchange with the server (solicit-reply) instead of the default four-message exchange (solicit-advertise-request-reply).

**Usage**

A DHCPv6-assigned address can be configured on a VLAN when the following is true:

- The assigned address is not on the same subnet as a previously configured autoconfig address.
- The maximum IPv6 address limit on the VLAN or the switch has not been reached.

If the switch is an IPv6 host, `ipv6 address dhcp full` must be configured on the DHCPv6 client to obtain relevant information from the DHCPv6 server. M-bit and O-bit settings in RAs from a router are not used by the switch in host mode. If the switch is operating as an IPv6 router, it includes M-bit and O-bit values in the RAs it transmits.

Default: Disabled

NOTE

A DHCPv6 server does not assign link-local addresses, and enabling DHCPv6 on a VLAN does not affect a pre-existing link-local address.

# Configuring a static IPv6 address on a VLAN

This option enables configuring of unique, static unicast IPv6 addresses for global, and link-local applications, including:

- link-local unicast (including EUI and non-EUI interface identifiers)
- global unicast (and unique local unicast)

## ipv6 address link-local

**Syntax**

```
ipv6 address fe80::<interface-id> link-local
no ipv6 address fe80::<interface-id> link-local
```

**Description**

- If IPv6 is not already enabled on the VLAN, this command enables IPv6 and configures a static link-local address.
- If IPv6 is already enabled on the VLAN, this command overwrites the current, link-local address with the specified static address. (One link-local address is enabled per VLAN interface.)

For link-local addressing, the `no` form of the static IPv6 address command produces different results, depending on how IPv6 is configured on the VLAN:

- If IPv6 was enabled only by a statically configured link-local address, deleting the link-local address disables IPv6 on the VLAN.
- If other IPv6-enabling commands have been configured on the VLAN, deleting the statically configured link-local address causes the switch to replace it with the default (EUI-64) link-local address for the VLAN, and IPv6 remains enabled.

**Options**

***`<interface-id>`***

The low-order 64 bits, in 16-bit blocks, comprise this value in a link-local address:

`xxxx xxxx : xxxx xxxx : xxxx xxxx : xxxx xxxx`

Where a static link-local address is already configured, a new, autoconfigured global unicast addresses assignment uses the same interface identifier as the link-local address.

**Usage**

An existing link-local address is replaced, and is not deprecated, when a static replacement is configured. The prefix for a statically configured link-local address is always 64 bits, with all blocks after fe80 set to zero. That is: fe80:0:0:0. After verification of uniqueness by DAD, a statically configured link-local address status is set to `preferred`, with a `permanent` lifetime.

## Commands to statically configure a global unicast address

**Syntax**

```
ipv6 address [<network-prefix> <interface-id> | <prefix-length> ]
ipv6 address [<network-prefix> ::/ <prefix-length> eui-64 ]
```

**Options**

***`<network-prefix>`***

This includes the global routing prefix and the subnet ID for the address.

***`<interface-id>`***

Enters a user-defined interface identity.

***`<prefix-length>`***

Specifies the number of bits in the network prefix. If you are using the `eui-64` option, this value must be 64.

**`eui-64`**

Specifies using the Extended Unique Identifier (EUI) format to create an interface identifier based on the VLAN MAC address.

**Usage**

If IPv6 is not already enabled on a VLAN, either of these command options do the following:

- enable IPv6 on the VLAN
- configure a link-local address using the EUI-64 format
- statically configure a global unicast address

If IPv6 is already enabled on the VLAN, the above commands statically configure a global unicast address, but have no effect on the current link-local address.

After verification of uniqueness by DAD, the lifetime of a statically configured IPv6 address assigned to a VLAN is set to `permanent` and is configured as a preferred address.

## Operating notes

- With IPv6 enabled, the switch determines the default IPv6 router for the VLAN from the RAs it receives.
- If DHCPv6 is configured on a VLAN, then configuring a static global unicast address on the VLAN removes DHCPv6 from the VLAN configuration and deletes the DHCPv6-assigned global unicast address.
- For a statically configured global unicast address to be routable, a gateway router must be transmitting RAs on the VLAN.
- If an autoconfigured global unicast address exists for the same subnet as a new, statically configured global unicast address, the statically configured address is denied. In the reverse case, you can add an autoconfig command to the VLAN configuration, but it will not be implemented unless the static address is removed from the configuration.

## Duplicate address detection (DAD) for statically configured addresses

Statically configured IPv6 addresses are designated as permanent. If DAD determines that a statically configured address duplicates a previously configured and reachable address on another device belonging to the VLAN, the more recent, duplicate address is designated as `duplicate`.

# IPv6 loopback interfaces

This section describes how to configure and use user-defined loopback interfaces on the switch.

By default, each switch has eight internal IPv6 loopback interfaces (`lo-0` as through `lo-7`) with IPv6 address :: 1/128 configured by default on `lo-0`. This address (::1/128) is used only for internal traffic transmitted within the switch and is not used in packet headers in egress traffic sent to network devices.

Each loopback interface can have multiple IPv6 addresses, all of which must be unique.

User-defined IPv6 loopback addresses provide these benefits when a routing protocol is enabled:

- A loopback interface is a virtual interface that is always up and reachable as long as at least one of the IPv6 interfaces on the switch is operational. As a result, a loopback interface is useful for debugging tasks because its address can always be pinged if any other switch interface is up.
- You can use a loopback interface to establish a Telnet session, ping the switch, and access the switch through SNMP and HTTP (Web Agent).

## interface loopback

**Syntax**

`interface loopback` *`<0-7>`* `ipv6 address` *`<ipv6-addr>`*

**Description**

The following command enables nondefault IPv6 address configuration on loopback interfaces.

**Restrictions**

You cannot remove the (reserved) default loopback interface address `::1/128` from `lo-0`.

**Usage**

You can configure up to 32 IPv6 addresses (and up to 32 IPv4 addresses) on a loopback interface. To configure an IPv6 address for the loopback interface, enter the `ipv6 address <ip-address>` command at the loopback interface configuration level, as shown in the following example. When you configure an IPv6 address for a loopback interface, you do not specify a prefix. The default prefix/128 applies automatically.

**Example input**

Configuring an IPv6 address on a loopback interface

```
Switch(config)# interface loopback 1
Switch(lo-1)# IPv6 address 2001:db8::1
```

NOTE

- You can configure a loopback interface only from the CLI; you cannot configure a loopback interface from the Web Agent or Menu interface.
- IPv6 loopback interfaces share IPv6 address space with VLAN configurations. The maximum number of IPv6 addresses supported on a switch is 2048, which includes all IPv6 addresses configured for both VLANs and loopback interfaces (except for the default loopback IPv6 address, ::1 /128).
- Each IPv6 address that you configure on a loopback interface must be unique in the autonomous system (AS). This means that the address cannot be used by a VLAN interface or another loopback interface.
- You can configure up to 32 IPv6 and 32 IPv4 addresses on a loopback interface (`lo0` to `lo7`).

## show ipv6

**Syntax**

`show ipv6`

**Description**

This command displays the list of loopback interfaces configured with nondefault IPv6 addresses. (Loopback interface 0, if configured only with the default ::1/128 IPv6 address, does not open in this listing.)

NOTE

A loopback interface does not open in the `show ipv6` command output unless it is configured with a nondefault IPv6 address.

# Commands to view the current IPv6 addressing configuration

Use these commands to view the status of the IPv6 configuration on the switch.

**`show ipv6`**

Lists the current global IPv6 settings and per-VLAN IPv6 addressing on the switch.

**`IPv6 Routing`**

For software releases K.13.01 through K.14.01, this setting is always `Disabled`. This is a global setting and is not configured per-VLAN.

**`Default Gateway`**

Lists the IPv4 default gateway, if any, configured on the switch. This is a globally configured router gateway address and is not configured per-VLAN.

**`ND DAD`**

Indicates whether DAD is enabled (the default) or disabled. Using `ipv6 nd dad-attempts 0` disables ND.

**DAD Attempts**

Indicates the number of neighbor solicitations the switch transmits per-address for duplicate (IPv6) address detection. Implemented when a new address is configured or when an interface with configured addresses (such as after a reboot). Default: 3; Range: *0 – 255* ms. 0 disables duplicate address detection.

**VLAN Name**

Lists the name of a VLAN statically configured on the switch.

**IPv6 Status**

For the indicated VLAN, shows whether IPv6 is disabled (the default) or enabled.

**Address Origin**

**Autoconfig**

The address was configured using SLAAC. In this case, the interface identifier for global unicast addresses copied from the current link-local unicast address.

**DHCP**

The address was assigned by a DHCPv6 server. Addresses having a DHCP origin are listed with a 128-bit prefix length.

**Manual**

The address was statically configured on the VLAN.

**IPv6 Address/Prefix Length**

Lists each IPv6 address and prefix length configured on the indicated VLAN.

**Address Status**

**Tentative**

DAD has not yet confirmed the address as unique, and it is not usable for sending and receiving traffic.

**Preferred**

The address has been confirmed as unique by DAD and usable for sending and receiving traffic. The Expiry time shown for this address by the `show ipv6 vlan` *vid* command output is the preferred lifetime assigned to the address.

**Deprecated**

The preferred lifetime for the address has been exceeded, but there is time remaining in the valid lifetime.

**Duplicate**

Indicates a statically configured IPv6 address that is a duplicate of another IPv6 address that exists on another device belonging to the same VLAN interface. A duplicate address is not used.

### Example output

This example shows the output on a switch having IPv6 enabled on one VLAN.

```
Switch# show ipv6
Internet (IPv6) Service

  IPv6 Routing    : Enabled
  ND DAD          : Enabled
  DAD Attempts    : 3


                    VLAN Interfaces
  Interface Name  : DEFAULT_VLAN
  IPv6 Status     : Disabled
  Layer 3 Status  : Enabled
```

```
 Interface Name  : VLAN22
 IPv6 Status     : Enabled
 Layer 3 Status  : Enabled

 Address    |                                   Address
 Origin     | IPv6 Address/Prefix Length        Status
 ---------- + --------------------------------- ---------
 autoconfig | fe80::218:71ff:feb9:8500/64       tentative


 Tunnel Interfaces

Interface Name   : TUNNEL3
IPv6 Status      : Enabled
Layer 3 Status   : NA

 Address    |                                   Address
 Origin     | IPv6 Address/Prefix Length        Status
 ---------- + --------------------------------- ---------
 autoconfig | fe80::218:71ff:feb9:8500/64       tentative
```

## show ipv6 nd

### Syntax

`show ipv6 nd`

### Description

Displays the current IPv6 ND settings on the configured VLAN interfaces.

### Example output

This example shows the output on a switch having IPv6 enabled on VLANs 1 and 22.

```
Switch# show ipv6 nd
IPV6 Neighbor Discovery Configuration

  Interface DAD       RCH Time   NS Interval
  Name      Attempts  (msecs)    (msecs)
  --------- --------- ---------- -----------
  1         3         30000      1000
  22        3         30000      1000
```

## show ipv6 vlan

### Syntax

`show ipv6 vlan [vlan vid|tunnel` *`tunnel-id`*`]`

### Description

Displays IP and IPv6 global configuration settings, the IPv6 status for the specified VLAN, the IPv6 addresses (with prefix lengths) configured on the specified VLAN, and the expiration data (Expiry) for each address.

### Example output

This example shows IPv6 status for the specified VLAN.

```
Switch# show ipv6 vlan 10

Internet (IPv6) Service

  IPv6 Routing     : Disabled
  Default Gateway : fe80::213:c4ff:fedd:14b0%vlan10
```

```
ND DAD            : Enabled
DAD Attempts      : 3

Vlan Name         : VLAN10
IPv6 Status       : Enabled

IPv6 Address/Prefixlength          Expiry
---------------------------------- -------------------------
2001:db8:a03:e102::1:101/64        Fri May 19 11:51:15 2009

fe80::1:101/64                     permanent
```

## show run

### Syntax

`show run`

### Description

In addition to the other elements of the current configuration, this command lists the statically configured, global unicast IPv6 addressing and the current IPv6 configuration per-VLAN. The listing might include one or more of the following, depending on what other IPv6 options are configured on the VLAN. Any SLAAC commands in the configuration are also listed in the output, but the actual addresses resulting from these commands are not included in the output.

### Options

`ipv6 address fe80::`*`interface-id`*` link-local`

`ipv6 address `*`prefix`*`:`*`interface-id`*`/`*`prefix-length`*

`ipv6 address autoconfig`

`ipv6 address dhcp full [ rapid-commit ]`

`ipv6 `*`global-unicast-address`*`/`*`prefix`*

### Example output

`show run` output listing the current IPv6 addressing commands

```
Switch# show run

Running configuration:
.
.
.
vlan 10
   name "VLAN10"
   untagged 1-12
   ipv6 address fe80::1:101 link-local 1
   ipv6 address dhcp full rapid-commit 2
```

[1] Statically configured IPv6 addresses open in the `show run` output.

[2] Commands for automatic IPv6 address configuration appear in the show run output, but the addresses resulting from these commands do not appear in the output.

# View IPv6 gateway, route, and router neighbors

Use these commands to view the current routing table content of the switch and connectivity to routers per VLAN. This includes information received in RAs from IPv6 routers on VLANs enabled with IPv6 on the switch.

## show ipv6 route

### Syntax

`show ipv6 route` *`ipv6-addr connected`*

### Description

Displays the routes in the IPv6 routing table of the switch.

### Options

**`ipv6-addr`**

Optional. Limits the output to show the gateway to the specified IPv6 address.

**`connected`**

Optional. Limits the output to show only the gateways to IPv6 addresses connected to VLAN interfaces configured on the switch, including the loopback (::1/128) address.

**`Dest`**

The destination address for a detected route.

**`Gateway`**

The IPv6 address or VLAN interface used to reach the destination. (Includes the loopback address.)

**`Type`**

Indicates route type (static, connected, or OSPF).

**`Distance`**

The administrative route distance, used to determine the best path to the destination.

**`Metric`**

Indicates the route cost for the selected destination.

### Example output

This example displays the routes in the IPv6 routing table of the switch.

**"Unknown" Address**

```
Dest : ::/0                                       Type : static
Gateway : fe80::213:c4ff:fedd:14b0%vlan10               Dist. : 40 Metric : 0
```

**Loopback Address**

```
Dest : ::1/128                                    Type : connected
Gateway : lo0                                           Dist. : 0 Metric : 1
```

**Global Unicast Address configured on the Switch**

```
Dest : 2001:db8:a03:e102::/64                     Type : connected
Gateway : VLAN10                                        Dist. : 0 Metric : 1
```

**Link-Local Address configured on the Switch**

```
Dest : fe80::%vlan10                              Type : connected
Gateway : VLAN10                                        Dist. : 0 Metric : 1
```

**Link-Local Address Assigned to the Loopback Address**

```
Dest : fe80::1%lo0                                Type : connected
Gateway : lo0                                           Dist. : 0 Metric : 1
```

## show ipv6 routers

### Syntax

`show ipv6 routers [ vlan vid ]`

### Description

Lists the switch's IPv6 router table entries for all VLANs configured on the switch or for a single VLAN. This output provides information about the IPv6 routers from which RAs have been received on the switch.

### Options

**vlan *vid***

Optional. Specifies only the information on IPv6 routers on the indicated VLAN.

**Router Address**

The IPv6 address of the router interface.

**Preference**

The relative priority of prefix assignments received from the router when prefix assignments are also received on the same switch VLAN interface from other IPv6 routers.

**Interface**

The VLAN interface on which the router exists.

**MTU**

The maximum transmission unit (in bytes) allowed for frames on the path to the indicated router.

**Hop Limit**

The maximum number of router hops allowed.

**Prefix Advertised**

Lists the prefix and prefix size (number of leftmost bits in an address) originating with the indicated router.

**Valid Lifetime**

The total time the address is available, including the preferred lifetime and the additional time (if any) allowed for the address to exist in the deprecated state.

**Preferred Lifetime**

The length of time during which the address can be used freely as both a source and a destination address for traffic exchanges with other devices.

**On/Off Link**

Indicates whether the entry source is on the same VLAN as is indicated in the `Interface` field.

### Example output

This example shows `show ipv6 routers` command output

```
Switch(config)# show ipv6 routers

 IPv6 Router Table Entries

  Router Address : fe80::213:c4ff:fedd:14b0
  Preference     : Medium
  Interface      : VLAN10
  MTU            : 1500
  Hop Limit      : 64

                             Valid        Preferred    On/Off
```

```
Prefix Advertised            Lifetime(s)  Lifetime(s)  Link
---------------------------  ------------ ------------ -------
2001:db8:a03:e102::/64       864000       604800       Onlink
```

# Address lifetimes

Every configured IPv6 unicast address has a lifetime setting that determines how long the address can be used before it must be refreshed or replaced. Some addresses are set as "permanent" and do not expire. Others have both a "preferred" and a "valid" lifetime that specifies the duration of their use and availability.

## Preferred lifetime

This is the length of time during which the address can be used freely as both a source and a destination address for traffic exchanges with other devices. This time span is equal to or less than the valid lifetime also assigned to the address. If this time expires without the address being refreshed, the address becomes deprecated and should be replaced with a new, preferred address. In the deprecated state, an address can continue to be used as a destination for existing communication exchanges but is not used for new exchanges or as a source for traffic sent from the interface. A new, preferred address and its deprecated counterpart both appear in the `show ipv6 vlan` *vid* output as long as the deprecated address is within its valid lifetime.

## Valid lifetime

The valid lifetime, which is the total time the address is available, is equal to or greater than the preferred lifetime. The valid lifetime enables communication to continue for transactions that began before the address became deprecated. However, in this time frame, the address should no longer be used for new communications. If this time expires without the deprecated address being refreshed, the address becomes invalid and might be assigned to another interface.

**Table 1:** *IPv6 unicast addresses lifetimes*

| Address source | Lifetime criteria |
|---|---|
| Link-local | Permanent |
| Statically configured unicast | Permanent |
| Autoconfigured global | Finite preferred and valid lifetimes |
| DHCPv6-configured | Finite preferred and valid lifetimes |

A new, preferred address used as a replacement for a deprecated address can be acquired from a manual, DHCPv6, or autoconfiguration source.

# DHCPv6 client

The DHCPv6 client allows a host to request global unicast IPv6 address assignments from a DHCPv6 server. If there are multiple DHCPv6 servers, the client can select a server based on the preference value sent in DHCPv6 messages.

The DHCPv6 client can request that the server send only the configuration information. In this case, a router on the same interface (VLAN) as the host provides the global IPv6 address to the host through router advertisements.

NOTE

If the switch is rebooted with a default configuration, only the default DHCPv4 client is enabled on the default VLAN. The DHCPv6 client has to be explicitly enabled on a VLAN using the command `ipv6 address dhcp` or `ipv6 address autoconfig`.

# Duplicate address detection (DAD) for statically configured addresses

Statically configured IPv6 addresses are designated as permanent. If DAD determines that a statically configured address duplicates a previously configured and reachable address on another device belonging to the VLAN, the more recent, duplicate address is designated as `duplicate`.

# IPv6 loopback interfaces

By default, each switch has eight internal IPv6 loopback interfaces (`lo-0` as through `lo-7`) with IPv6 address ::1/128 configured by default on `lo-0`. This address (::1/128) is used only for internal traffic transmitted within the switch and is not used in packet headers in egress traffic sent to network devices.

Each loopback interface can have multiple IPv6 addresses, all of which must be unique. Routing protocols such as OSPFv3, advertise the configured loopback addresses throughout a network or autonomous system.

User-defined IPv6 loopback addresses provide these benefits when a routing protocol is enabled:

- A loopback interface is a virtual interface that is always up and reachable as long as at least one of the IPv6 interfaces on the switch is operational. As a result, a loopback interface is useful for debugging tasks because its address can always be pinged if any other switch interface is up.
- You can use a loopback interface to establish a Telnet session, ping the switch, and access the switch through SNMP.

# Commands to disable IPv6 on a VLAN

While one IPv6-enabling command is configured on a VLAN, IPv6 remains enabled on that VLAN. In this case, removing the only IPv6-enabling command from the configuration disables IPv6 operation on the VLAN. That is, to disable IPv6 on a VLAN, the following commands must be removed from the VLAN's configuration:

```
ipv6 enable

ipv6 address dhcp full [rapid-commit]

ipv6 address autoconfig

ipv6 address fe80::<interface-id> link-local

ipv6 address <prefix>:<interface-id>
```

If any of the above remain enabled, IPv6 remains enabled on the VLAN and, at a minimum, a link-local unicast address is present.

# Neighbor Discovery (ND)

Neighbor Discovery (ND) is the IPv6 equivalent of the IPv4 ARP for layer 2 address resolution. ND uses IPv6 ICMP messages to provide for discovery of IPv6 devices such as other switches, routers, management stations, and servers on the same interface. ND runs automatically in the default configuration and provides services in addition to those provided in IPv4 by ARP. For example:

- Determine the link-layer address of neighbors on the same VLAN interface.
- Verify that a neighbor is reachable. Track neighbor (local) routers.
- Track neighbor (local) routers.

Neighbor Discovery enables functions such as the following:

- router and neighbor solicitation and discovery
- detecting address changes for devices on a VLAN
- identifying a replacement for a router or router path that has become unavailable
- duplicate address detection (DAD)
- RA processing
- neighbor reachability
- autoconfiguration of unicast addresses
- resolution of destination addresses
- changes to link-layer addresses

An instance of ND is triggered on a device when a new (tentative) or changed IPv6 address is detected. (This includes stateless, stateful, and static address configuration.) ND operates in a per-VLAN scope, that is, within the VLAN on which the device running the ND instance is a member. ND actually occurs when there is communication between devices on a VLAN. That is, a device needing to determine the link-layer address of another device on the VLAN initiates a (multicast) neighbor solicitation message (containing a solicited-node multicast address that corresponds to the IPv6 address of the destination device) on the VLAN. When the destination device receives the neighbor solicitation, it responds with a neighbor advertisement message identifying its link-layer address. When the initiating device receives this advertisement, the two devices are ready to exchange traffic on the VLAN interface. Also, when an IPv6 interface becomes operational, it transmits a router solicitation on the interface and listens for an RA.

NOTE

Neighbor and router solicitations must originate on the same VLAN as the receiving device. To support this operation, IPv6 is designed to discard any incoming neighbor or router solicitation that does not have a value of 255 in the IP Hop Limit field. For a complete list of requirements, see RFC 246.

When a pair of IPv6 devices in a VLAN exchange communication, they enter each other's IPv6 and corresponding MAC addresses in their respective neighbor caches. These entries are maintained for a time after communication ceases and then dropped.

For related information, see RFC 2461: "Neighbor Discovery for IP Version 6 (IPv6)."

## Duplicate address detection (DAD)

DAD verifies that a configured unicast IPv6 address is unique before it is assigned to a VLAN interface on the switch. DAD is enabled in the default IPv6 configuration and can be reconfigured, disabled, or re-enabled at the global config or per-interface command level. DAD can be useful in helping to troubleshoot erroneous replies to DAD requests, or where the neighbor cache contains a large number of invalid entries caused by an unauthorized station sending false replies to the switch's ND queries. If DAD verifies that a unicast IPv6 address is a duplicate, the address is not used. If the link-local address of the VLAN interface is found to be a duplicate of an address for another device on the interface, the interface stops processing IPv6 traffic.

### DAD operation

On a given VLAN interface, when a new unicast address is configured, the switch runs DAD for this address by sending a neighbor solicitation to the All-Nodes multicast address (ff02::1). This operation discovers other devices on the VLAN and verifies whether the proposed unicast address assignment is unique on the VLAN. (During this time, the address being checked for uniqueness is held in a tentative state and cannot be used to receive traffic other than neighbor solicitations and neighbor advertisements.) A device that receives the neighbor solicitation responds with a neighbor advertisement that includes its link-local address. If the newly configured address is from a static or DHCPv6 source and is found to be a duplicate, it is labeled as duplicate in the "Address Status" field of the `show ipv6` command and is not used. If an autoconfigured address is found to be a duplicate, it is dropped and a similar message appears in the Event Log:

```
W <date> <time> 00019 ip: <ip address> <IPv6-address> removed from vlan id <vid>
```

DAD does not perform periodic checks of existing addresses. However, when a VLAN comes up with IPv6 unicast addresses configured (as can occur during a reboot), the switch runs DAD for each address on the interface by sending neighbor solicitations to the All-Nodes multicast address, as described above.

If an address is configured while DAD is disabled, the address is assumed to be unique and is assigned to the interface. If you want to verify the uniqueness of an address configured while DAD was disabled, re-enable DAD and then either delete and reconfigure the address, or reboot the switch.

## Configuring DAD

The following commands are used to configure DAD:

### ipv6 nd dad-attempts

**Syntax**

`ipv6 nd dad-attempts` *`0 - 255`*

`no ipv6 nd dad-attempts` *`0 - 255`*

**Description**

This command is executed at the global or per-interface config level, and configures the number of neighbor solicitations to send when performing DAD for a unicast address configured on a VLAN or tunnel interface. A per-interface configuration overrides a globally set configuration.

**Specifiers**

Default: 3 (enabled); 0 (disabled); Range: *0 - 255* (0 = disabled)

**Options**

***0 - 255***

The number of consecutive neighbor solicitation messages sent for DAD inquiries on an interface. Setting this value to 0 disables DAD on the interface, which bypasses checks for uniqueness on newly configured addresses. If a reboot is performed while DAD is disabled, the duplicate address check is not performed on any IPv6 addresses configured on the switch.

**`no`**

The `no` form of the command restores the default setting.

NOTE

Software version K.14.xx supports a `dad-attempts` range of 0 to 600. However, software version K.15.xx or greater supports a range of 0 to 255. If `dad-attempts` is set higher than 255, updating from K.14.xx to K.15.xx or greater and rebooting truncates this value to 255. Similarly, if the switch is running version K.15.xx or greater: (1) downloading a configuration file created using version K.14.xx with `dad-attempts` set higher than 255 and then (2) rebooting the switch using this configuration file truncates the setting to 255.

### ipv6 nd ns-interval

**Syntax**

`ipv6 nd ns-interval`

**Description**

Used on VLAN interfaces to reconfigure the ND time in milliseconds between DAD neighbor solicitations sent for an unresolved destination, or between duplicate address detection neighbor solicitation requests. Increasing this setting is indicated where neighbor solicitation retries or failures are occurring, or in a "slow" (WAN) network. This value can be configured in a router advertisement to help ensure that all hosts on a VLAN are using the same retransmit interval for ND.

Default: 1000 ms; Range: 1000 ms to 3600000 ms

### ipv6 nd reachable-time

**Syntax**

`ipv6 nd reachable-time` *`milliseconds`*

**Description**

Used on VLAN interfaces to configure the length of time in milliseconds a neighbor is considered reachable after the Neighbor Unreachability Detection algorithm has confirmed it to be reachable. When the switch operates in host mode, this setting can be overridden by a reachable time received in a router advertisement.

This value can be configured in an RA a router advertisement to help ensure that all hosts on a VLAN are using the same reachable time in their neighbor cache.

**Options**

**`show ipv6 nd`**

Use this command to view current setting.

## Operating notes for ND

- A verified link-local unicast address must exist on a VLAN interface before the switch can run DAD on other addresses associated with the interface.
- If a previously configured unicast address is changed, a neighbor advertisement (an all-nodes multicast message--ff02::1) is sent to notify other devices on the VLAN and to perform DAD.
- IPv6 addresses on a VLAN interface are assigned to multicast address groups identified with well-known prefixes.
- DAD is performed on all stateful, stateless, and statically configured unicast addresses.
- Neighbor solicitations for DAD do not cause the neighbor cache of neighboring switches to be updated.
- If a previously configured unicast address is changed, a neighbor advertisement is sent on the VLAN to notify other devices and for duplicate address detection.
- If DAD is disabled when an address is configured, the address is assumed to be unique and is assigned to the interface.

# Router access and default router selection

Traffic can be routed between destinations on different VLANs configured on the switch or to a destination on an off-switch VLAN. This is done by placing the switch on the same VLAN interface or subnet as an IPv6-capable router configured to route traffic to other IPv6 interfaces or to tunnel IPv6 traffic across an IPv4 network.

**More Information**

**IPv6 routing overview** on page 144

**Enable auto configuration of a global unicast address and a default router identity on a VLAN** on page 16

## Router advertisements

An IPv6 router periodically transmits RAs on the VLANs to which it belongs to notify other devices of its presence. The switch uses these advertisements for purposes such as:

- Learning the MAC and link-local addresses of IPv6 routers on the VLAN. (For devices other than routers, the switch must use ND to learn these addresses.)
- Building a list of default (reachable) routers, along with router lifetime and prefix lifetime data.
- Learning the prefixes and the valid and preferred lifetimes to use for stateless (autoconfigured) global unicast addresses. (This is required for autoconfiguration of global unicast IPv6 addresses.)
- Learning the hop limit for traffic leaving the VLAN interface.
- Learning the MTU to apply to frames intended to be routed.

## IPv6 router advertisement options for DNS configuration

Two new options in IPv6 Router Advertisements allow IPv6 routers to advertise a list of recursive DNS Server (RDNSS) addresses and a DNS Search List (DNSSL) to IPv6 hosts. RA-based DNS configuration enables the full configuration of basic networking information for hosts without requiring DHCPv6. An IPv6 host can acquire the DNS configuration (that is, the DNS recursive server addresses and DNS Search List) for the links to which the host is connected. The host learns this DNS configuration from the same RA message that provides configuration information for the link.

DNS options are included by default in every emitted RA unless the inclusion is suppressed via CLI commands or the SNMP MIB. The suppress option can be configured either globally or for an IP interface.

NOTE

This is supported in RFC 6106

### nd suppress-ra-dns

**Syntax**

```
no nd suppress-ra-dns
```

**Description**

This command suppresses the inclusion of RDNSS and SNSSL in outgoing Router Advertisements across all interfaces on a switch.

**Example input**

Suppressing the inclusion of RDNSS and SNSSL in outgoing RAs for an IP interface

```
Switch(config)# vlan 2
Switch(vlan-2)# ipv6 nd ra suppress-dns
```

## IP interface configuration

This example shows the command that suppresses the inclusion of RDNSS and SNSSL in outgoing Router Advertisements for an IP interface. The command is executed in VLAN context

**Suppressing the inclusion of RDNSS and SNSSL in outgoing RAs for an IP interface**

```
Switch(config)# vlan 2
Switch(vlan-2)# ipv6 nd ra suppress-dns
```

## Display configuration

The following example is the output showing which interfaces has RA-DNS suppressed.

**Displaying the configuration**

```
Switch(config)# show ipv6 nd ra

  IPv6 Router Advertisement Configuration
  Global RA Suppress:     : No
  Global RA-DNS Suppress  : Yes
  Global Hop Limit        : 64
  IPv6 Unicast Routing    : Disabled

          Suppress Suppress Interval Lifetime Mngd Other RCH Time Interval Hop
Interface RA       RA-DNS   Min/Max   (sec)   Flag Flag  (ms)      (ms)    Lim
--------- -------- -------- -------- -------- ---- ----- -------- -------- ---
```

```
vlan-1    No        No        200/600  1800      No    No     0         0         64
vlan-2    No        Yes       200/600  1800      No    No     0         0         64
vlan-3    No        No        200/600  1800      No    No     0         0         64
```

**Output from running-config command with router advertisement DNS suppressed**

```
Switch(config)# show running-config

Running configuration:

; J8698A Configuration Editor; Created on release #K.15.10.XXXX
; Ver #03:01.1f.ef:f2

Hostname " Switch"
module 1 type J9550A
ipv6 nd suppress-ra-dns
vlan 1
   name "DEFAULT_VLAN"
   untagged A1-A24
   ip address dhcp-bootp
   exit
vlan 2
   name "vlan2"
   no ip address
   ipv6 nd ra suppress-dns
   exit
spanning-tree
```

## Router solicitations

When an IPv6 interface becomes operational on the switch, a router solicitation is automatically sent to trigger an RA from any IPv6 routers reachable on the VLAN. (Router solicitations are sent to the All-Routers multicast address; ff02::2. If an RA is not received within one second of sending the initial router solicitation, the switch sends up to three additional solicitations at intervals of four seconds. If an RA is received, the sending router is added to the switch's default router list and the switch stops sending router solicitations. If an RA is not received, IPv6 traffic on that VLAN cannot be routed, and the only usable unicast IPv6 address on the VLAN is the link-local address.

NOTE

If the switch does not receive an RA after sending the router solicitations, as described above, no further router solicitations are sent on that VLAN unless a new IPv6 setting is configured, IPv6 on the VLAN is disabled and then re-enabled, or the VLAN itself is disconnected and then reconnected.

## Default IPv6 router

If IPv6 is enabled on a VLAN where there is at least one accessible IPv6 router, the switch selects a default IPv6 router.

- If the switch receives RAs from a single IPv6 router on the same VLAN or subnet, the switch configures a global unicast address and selects the advertising router as the default IPv6 router.
- If multiple IPv6 routers on a VLAN send RAs advertising the same network, the switch configures one global unicast address and selects one router as the default router, based on the router's relative reachability, using factors such as router priority and route cost.
- If multiple IPv6 routers on a VLAN send RAs advertising different subnets, the switch configures a corresponding global unicast address for each RA and selects one of the routers as the default IPv6 router, based on route cost. When multiple RAs are received on a VLAN, the switch uses the router priority and route cost information included in the RAs to identify the default router for the VLAN.

## Router redirection

With multiple routers on a VLAN, if the default (first-hop) router for an IPv6-enabled VLAN on the switch determines that there is a better first-hop router for reaching a given remote destination, the default router can redirect the switch to use that other router as the default router. For more information on routing IPv6 traffic, see the documentation provided for the IPv6 router.

This chapter focuses on the IPv6 application of management features that support both IPv6 and IPv4 operation. For additional information on these features, see the current *ArubaOS-Switch Management and Configuration Guide* for your switch.

NOTE

All commands previously in the Summary of commands table are indexed under the entry *Command syntax*.

# Viewing the neighbor cache

Neighbor discovery occurs when there is communication between the switch and another, reachable IPv6 device on the same VLAN. A neighbor destination is reachable from a given source address if a confirmation (neighbor solicitation) has been received at the source verifying that traffic has been received at the destination.

The switch maintains an IPv6 neighbor cache that is populated as a result of communication with other devices on the same VLAN.

You can view the contents of the neighbor cache using the commands described in this section.

## show ipv6 neighbors

**Syntax**

```
show ipv6 neighbors [vlan vid]
```

**Description**

Displays IPv6 neighbor information currently held in the neighbor cache. After a period without communication with a given neighbor, the switch drops that neighbor's data from the cache. The command lists neighbors for all VLAN interfaces on the switch or for only the specified VLAN. The following fields are included for each entry in the cache:

**Options**

**IPv6 Address**

Lists the 128-bit addresses for the local host and any neighbors (on the same VLAN) with whom there has been recent communication.

**MAC address**

The MAC address corresponding to each of the listed IPv6 addresses.

**VLAN *vid***

Optional. Causes the switch to list only the IPv6 neighbors on a specific VLAN configured on the switch.

**Type**

Appears only when VLAN is not specified and indicates whether the corresponding address is local (configured on the switch) or dynamic (configured on a neighbor device.

**Age**

Appears only when the VLAN is specified and indicates the length of time the entry has remained unused.

**Port**

Identifies the switch port on which the entry was learned. If this field is empty for a given address, the address is configured on the switch itself.

**State**

A neighbor destination is reachable from a given source address if confirmation has been received at the source verifying that traffic has been received at the destination. This field shows the reachability status of each listed address:

**INCMP**

(Incomplete): Neighbor address resolution is in progress, but has not yet been determined.

**REACH**

(Reachable): The neighbor is known to have been reachable recently.

**STALE**

A timeout has occurred for reachability of the neighbor, and an unsolicited discovery packet has been received from the neighbor address. If the path to the neighbor is then used successfully, this state is restored to REACH.

**DELAY**

Indicates waiting for a response to traffic sent recently to the neighbor address. The time for determining the neighbor's reachability has been extended.

**PROBE**

The neighbor might not be reachable. Periodic, unicast neighbor solicitations are being sent to verify reachability.

**Example output**

Neighbor cache without specifying a VLAN

```
Switch(config)# show ipv6 neighbor
 IPv6 ND Cache Entries
 IPv6 Address                MAC Address   State Type    Port
 --------------------------- ------------- ----- ------- ----
 2001:db8:260:212::101       0013c4-dd14b0 STALE dynamic 1
 2001:db8:260:214::1:15      001279-88a100 REACH local
 fe80::1:1                   001279-88a100 REACH local
 fe80::10:27                 001560-7aadc0 REACH dynamic 3
 fe80::213:c4ff:fedd:14b0    0013c4-dd14b0 REACH dynamic 1
```

## clear ipv6 neighbors

**Syntax**

```
clear ipv6 neighbors
```

**Description**

You can clear the contents of the neighbor cache using the commands described in this section. Executed at the global config level, this command removes all nonlocal IPv6 neighbor addresses and corresponding MAC addresses from the neighbor cache, except neighbor entries specified as next-hops for active routes. The Layer-2 address information for any next-hop route is cleared until the route is refreshed in the neighbor cache.

**Example output**

Clearing the contents of the neighbor cache:

```
Switch(config)# clear ipv6 neighbors
Switch(config)# show ipv6 neighbors
```

```
Switch# show ipv6 neighbors
 IPv6 ND Cache Entries
 IPv6 Address                MAC Address   State Type
 --------------------------- ------------- ----- -------
 fe80::213:c4ff:fedd:14b0    000000-000000 INCMP dynamic
```

# IPv6 Telnet operations

This section describes Telnet operation for IPv6 on the switch. For IPv4 Telnet operation, see the *ArubaOS-Switch Management and Configuration Guide* for your switch.

## Using outbound Telnet to another device

Outbound Telnet establishes a Telnet session from the switch CLI to another IPv6 device and includes these options:

### telnet link-local-addr

**Syntax**

telnet *link-local-addr* %vlan *vid*[ oobm ]

**Description**

Telnet for link-local addresses on the same VLAN requires the link-local address and interface scope.

**Options**

***link-local-addr***

Specifies the link-local IPv6 address of the destination device.

**%vlan *vid***

Suffix specifying the interface on which the destination device is located. No spaces are allowed in the suffix.

### telnet global-unicast-addr

**Syntax**

telnet *global-unicast-addr [ oobm ]*

**Description**

Telnet for global unicast addresses requires a global unicast address for the destination. Also, the switch must be receiving RAs from an IPv6 gateway router.

**Options**

***global-unicast-addr***

Specifies the global IPv6 address of the destination device.

## show telnet

**Syntax**

show telnet

**Description**

This command shows the active incoming and outgoing Telnet sessions on the switch (for both IPv4 and IPv6). Command output includes the following:

**Session**

The session number. The switch allows one outbound session and up to five inbound sessions.

**Privilege**

Manager or Operator.

**From**

Console (for outbound sessions) or the source IP address of the inbound session.

**To**

The destination of the outbound session, if in use.

**Example output**

This example shows the active incoming and outgoing Telnet sessions on the switch.

```
Switch# show telnet

 Telnet Activity

 ---------------------------------------------------
 Session  : 1
 Privilege: Manager
 From     : Console
 To       : 10.0.10.140
 ---------------------------------------------------
 Session  : 2
 Privilege: Manager
 From     : 2620:0:260:212::2:219
 To       :
 ---------------------------------------------------
 Session  : ** 3
 Privilege: Manager
 From     : fe80::2:101
 To       :
```

## telnet-server listen

**Syntax**

`telnet-server listen [oobm|data|both]`

`no telnet-server listen [oobm|data|both]`

**Description**

This command is used at the global config level to enable (the default) or disable all (IPv4 and IPv6) inbound Telnet access to the switch.

**Options**

**`no`**

The `no` form of the command disables inbound Telnet.

**`listen`**

The `listen` parameter is available only on switches that have a separate OOBM port. Values for this parameter are:

**oobm**

Inbound Telnet access is enabled only on the OOBM port.

**data**

Inbound Telnet access is enabled only on the data ports.

**both**

Inbound Telnet access is enabled on both the OOBM port and on the data ports. This is the default value.

## show console

### Syntax

`show console`

### Description

This command shows the current configuration of IPv4 and IPv6 inbound Telnet permissions, as well as other information. For both protocols, the default setting allows inbound sessions.

### Example output

This example shows the current configuration of IPv4 and IPv6 inbound Telnet permissions.

```
Switch(config)# show console

 Console/Serial Link

  Inbound Telnet Enabled [Yes] : Yes
  Web Agent Enabled [Yes] : Yes
  Terminal Type [VT100] : VT100
  Screen Refresh Interval (sec) [3] : 3
  Displayed Events [All] : All

  Baud Rate [Speed Sense] : speed-sense
  Flow Control [XON/XOFF] : XON/XOFF
  Session Inactivity Time (min) [0] : 0
```

# sntp server priority

### Syntax

`sntp server priority` *`1 - 3 link-local-addr %vlan 1 - 3 [ oobm ][ 1 -7 ]`*

`no sntp server priority` *`1 - 3 link-local-addr %vlan 1 - 3 [ oobm ][ 1 -7 ]`*

### Description

Configures an IPv6 address for an SNTP server.

### Parameters

**`server priority`** ***`1 - 3`***

Specifies the priority of the server addressing being configured. When the SNTP mode is set to unicast and more than one server is configured, this value determines the order in which the configured servers will be accessed for a time value. The switch polls multiple servers in order until a response is received or until all servers on the list have been tried without success. Up to three server addresses (IPv6 and/or IPv4) can be configured.

***link-local-addr***

Specifies the link-local IPv6 address of the destination device.

**%vlan *vid***

Suffix specifying the interface on which the destination device is located. No spaces are allowed in the suffix.

***global-unicast-addr***

Specifies the global IPv6 address of the destination device.

**oobm**

For switches that have a separate OOBM port, `oobm` specifies that SNTP traffic goes through that port. (By default, SNTP traffic goes through the data ports.)

**[ *1* - *7* ]**

This optional setting specifies the SNTP server version expected for the specified server. Default: 3

**Example**

To configure link-local and global unicast SNTP server addresses of:

- fe80::215:60ff:fe7a:adc0 (on VLAN 10, configured on the switch)
- 2001:db8::215:60ff:fe79:8980

as the priority "1" and "2" SNTP servers, respectively, using version 7, you would enter these commands at the global config level, as shown below:

```
Switch(config)# sntp server priority 1 fe80::215:60ff:fe7a:adc0%vlan10 7
Switch(config)# sntp server priority 2 2001:db8::215:60ff:fe79:8980 7
```

NOTE: In the preceding example, using a link-local address requires that you specify the local scope for the address; VLAN 10 in this case. This is always indicated by `%vlan` followed immediately (without spaces) by the VLAN identifier.

## show sntp

**Syntax**

`show sntp`

**Description**

Displays the current SNTP configuration, including the following:

**Options**

**Time sync mode**

Indicates whether `timesync` is disabled or set to either `SNTP` or `Timep`. Default: `timep`

**SNTP mode**

Indicates whether SNTP uses the broadcast or unicast method of contacting a time server. The broadcast option does not require you to configure a time server address. The unicast option does require configuration of a time server address.

**Poll interval**

Indicates the interval between consecutive time requests to an SNTP server.

**Priority**

Indicates the configured priority for the corresponding SNTP server address.

**SNTP server address**

Lists the currently configured SNTP server addresses.

**Protocol version**

Lists the SNTP server protocol version to expect from the server at the corresponding address.

**Example output**

The `show sntp` output for the proceeding `sntp server` command example would appear as follows:

```
Switch(config)# show sntp

 SNTP Configuration

  Time Sync Mode: Sntp
  SNTP Mode : Broadcast
  Poll Interval (sec) [720] : 719

  Priority SNTP Server Address                   Protocol Version
  -------- ----------------------------------- ----------------
  1         2001:db8::215:60ff:fe79:8980         7
  2         10.255.5.24                          3
```

NOTE

The `show management` command can also be used to display SNTP server information.

## show timep

**Syntax**

`show timep`

**Description**

Displays the current timep configuration, including the following:

**Time sync mode**

Indicates whether timesync is disabled or set to either SNTP or Timep. Default: Disabled

**Timep mode**

Indicates whether Timep is configured to use a DHCP server to acquire a Timep server address or to use a statically configured Timep server address.

**Server address**

Lists the currently configured Timep server address.

**Poll interval (min) [ 720 ]**

Indicates the interval between consecutive time requests to the configured Timep server.

**Example output**

The `show timep` output for the preceding `ip timep manual` command example would appear as follows:

```
Switch(config)# show timep

 Timep Configuration

  Time Sync Mode: Timep
  TimeP Mode [Disabled] : Manual
```

```
Server Address : fe80::215:60ff:fe7a:adc0%vlan10
Poll Interval (min) [720] : 720
```

## ip timep

### Syntax

`ip timep dhcp interval 1 - 9999`

`ip timep manual {ipv6-addr|ipv4-addr}[ interval 1 - 9999 ][ oobm ]`

### Description

Used at the global config level to configure a Timep server address.

NOTE: The switch allows one Timep server configuration.

### Options

**timep dhcp**

Configures the switch to obtain the address of a Timep server from an IPv4 or IPv6 DHCP server.

**timep manual**

Specifies static configuration of a Timep server address.

***ipv6-addr***

Specifies the IPv6 address of an SNTP server.

**[ interval *1 - 9999* ]**

This optional setting specifies the interval in minutes between Timep requests. Default: 720

**[ oobm ]**

For switches that have a separate OOBM port, `oobm` specifies that Timep traffic goes through that port. (By default, Timep traffic goes through the data ports.)

### Example

To configure a link-local Timep server address of:

`fe80::215:60ff:fe7a:adc0`

where the address is on VLAN 10, configured on the switch, enter this command at the global config level:

```
Switch(config)# ip timep manual fe80::215:60ff:fe7a:adc0%vlan10
```

NOTE: In the preceding example, using a link-local address requires that you specify the local scope for the address; VLAN 10 in this case. This is always indicated by `%vlan` followed immediately (without spaces) by the VLAN identifier. For a global unicast address, you would enter the address without the `%vlan` suffix.

# TFTP files transfer over IPv6

You can use TFTP `copy` commands over IPv6 to upload, or download files to and from a physically connected device or a remote TFTP server, including:

- Switch software
- Software images
- Switch configurations

- ACL command files
- Diagnostic data (crash data, crash log, and event log)

## Upload/download files to the switch using TFTP

To upload and/or download files to the switch using TFTP in an IPv6 network, you must:

**Procedure**

1. Enable TFTP for IPv6 on the switch.
2. Enter a TFTP `copy` command with the IPv6 address of a TFTP server in the command syntax.
3. Optional: To enable auto-TFTP operation, enter the `auto-tftp` command.

## tftp client|server

Client and server TFTP for IPv6 is enabled by default on the switch. However, if it is disabled, you can re-enable it by specifying TFTP client or server functionality with the `tftp client|server` command.

**Syntax**

`tftp [ client ]| server [ listen oobm | data | both ]`

`no tftp [ client ]| server [ listen oobm | data | both ]`

**Description**

Enables TFTP for IPv4 and IPv6 client or server functionality so that the switch can:

- Use TFTP client functionality to access IPv4- or IPv6-based TFTP servers in the network to receive downloaded files.
- Use TFTP server functionality on the switch to be accessed by other IPv4 or IPv6 hosts requesting to upload files.
- For switches that have a separate OOBM port, the `listen` parameter in a `server` configuration allows you to specify whether transfers take place through the OOBM interface, the data interface, or both. For more information on OOBM, see the *ArubaOS-Switch Management and Configuration Guide*.

**Options**

The `no` form of the command disables the client or server functionality.

Default: TFTP client and server functionality enabled

NOTE

To disable all TFTP client or server operation on the switch except for the auto-TFTP feature, enter the `no tftp [ client | server ]` command. To re-enable TFTP client or server operation, re-enter the `tftp [ client | server ]` command. (Entering `no tftp` without specifying client or server affects only the client functionality. To disable or re-enable the TFTP server functionality, you must specify `server` in the command.)

When TFTP is disabled, instances of TFTP in the CLI `copy` command and the Menu interface "Download OS" screen become unavailable. The `no tftp [` *client|server* ] command does not affect auto-TFTP operation.

## Commands to copy files over IPv6 using TFTP

This section specifies commands for downloading or uploading a data file from or to a TFTP server.

### copy tftp target

**Syntax**

`copy tftp target ipv6-addr filename [ oobm ]`

**Description**

Copies (downloads) a data file from a TFTP server at the specified IPv6 address to a target file on a switch that is enabled with TFTP server functionality.

**Options**

***ipv6-addr***

If this is a link-local address, use this IPv6 address format:

`fe80::`*`device-id`*` %vlan `*`vid`*

For example:

`fe80::123%vlan10`

If this is a global unicast address, use this IPv6 format:

*`ipv6-addr`*

For example:

`2001:db8::123`

***target***

One of the following values:

**`autorun-cert-file`**

Copies an autorun trusted certificate to the switch.

**`autorun-key-file`**

Copies an autorun key file to the switch.

**`command-file`**

Copies a file stored on a remote host and executes the ACL command script on the switch. Depending on the ACL commands stored in the file, one of the following actions is performed in the running-config file on the switch:

- A new ACL is created.
- An existing ACL is replaced.
- `match`, `permit`, or `deny` statements are added to an existing ACL.

For more information on ACLs, see "Creating an ACL Offline" in the "Access Control Lists (ACLs)" chapter in the *Access Security Guide*

**`config`** ***`filename`***

Copies the contents of a file on a remote host to a configuration file on the switch.

**`flash [`*`primary`*`|`*`secondary`*`]`**

Copies a software file stored on a remote host to primary or secondary flash memory on the switch. To run a newly downloaded software image, enter the `reload` or `boot system flash` command.

**`pub-key-file`**

Copies a public-key file to the switch.

**`startup-config`**

Copies a configuration file on a remote host to the startup configuration file on the switch.

**`oobm`**

Switches that have a separate OOBM port specify that the transfer will be through the OOBM interface. (Default is transfer through the data interface.)

## copy source tftp

**Syntax**

`copy source tftp ipv6-addr filename [pc|unix ][ oobm ]`

**Description**

Copies (uploads) a source data file on a switch that is enabled with TFTP server functionality to a file on the TFTP server at the specified IPv6 address, where *source* is one of the following values:

**Options**

**command-output *cli-command***

Copies the output of a CLI command to the specified file on a remote host.

**config *filename***

Copies the specified configuration file to a remote file on a TFTP server.

**crash-data [*slot-id*|*master*]**

Copies the contents of the crash data file to the specified file path on a remote host. The crash data is software-specific and used to determine the cause of a system crash. You can copy crash information from an individual uplink module or from the master crash file on the switch.

**crash-log [*slot-id*|*master*]**

Copies the contents of the crash log to the specified file path on a remote host. The crash log contains processor-specific operational data that is used to determine the cause of a system crash. You can copy the contents of the crash log from an individual uplink module or from the master crash log on the switch.

**event-log**

Copies the contents of the Event Log on the switch to the specified file path on a remote host.

**flash [primary|secondary]**

Copies the software file used as the primary or secondary flash image on the switch to a file on a remote host.

**startup-config**

Copies the startup configuration file in flash memory to a remote file on a TFTP server.

**running-config**

Copies the running configuration file to a remote file on a TFTP server.

***ipv6-addr***

If this is a link-local address, use this IPv6 address format:

`fe80::device-id %vlan vid`

For example:

`fe80::123%vlan10`

If this is a global unicast address, use this IPv6 format:

*ipv6-addr*

For example:

`2001:db8::123`

**oobm**

Switches that have a separate OOBM port specify that the transfer will be through the OOBM interface. (Default is transfer through the data interface.)

## auto tftp

**Syntax**

`auto-tftp` *`ipv6-addr filename`*

**Description**

Configures the switch to download automatically the specified software file from the TFTP server at the specified IPv6 address. The file is downloaded into primary flash memory at switch startup. The switch then automatically reboots from primary flash.

**Options**

The `no` form of the command disables auto-TFTP operation by deleting the `auto-tftp` entry from the startup configuration.

**Subcommands**

The `no auto-tftp` command does not affect the current TFTP-enabled configuration on the switch. However, entering the `ip ssh filetransfer` command automatically disables both auto-tftp and tftp operation.

**Restrictions**

To enable auto-TFTP to copy a software image to primary flash memory, the version number of the downloaded software file (for example, K_14_01.swi) must be different from the version number currently in the primary flash image.

# SNMP management for IPv6

## Supported SNMP features

The same SNMP for IPv4 features is supported over IPv6:

- Access to a switch using SNMP version 1, version 2c, or version 3
- Enhanced security with the configuration of SNMP communities and SNMPv3 user-specific authentication password and privacy (encryption) settings
- SNMP notifications, including:
  - SNMP version 1 or SNMP version 2c traps
  - SNMPv2c informs
  - SNMPv3 notification process, including traps
- Advanced RMON (remote monitoring) management
- Flow sampling using sFlow
- Standard MIBs, such as the Bridge MIB (RFC 1493) and the Ethernet MAU MIB (RFC 1515)

## Supported SNMP configuration commands

This section specifies the SNMP configuration commands supported for IPv6 addresses. For more information on each SNMP configuration procedure, see the current *ArubaOS-Switch Management and Configuration Guide* for your switch.

### snmp-server host

**Syntax**

```
snmp-server host [ipv6-addr|ipv4-addr ][none|all|non-info|critical|debug][ inform
[retries count][timeout interval]]
```

**Description**

Executed at the global config level to configure an SNMP trap receiver to receive SNMPv1 and SNMPv2c traps, SNMPv2c informs, and (optionally) Event Log messages.

## snmpv3 targetaddress

### Syntax

`snmpv3 targetaddress` *`name`* `params` *`params_name`*

### Description

Executed at the global config level to configure an SNMPv3 management station to which notifications (traps and informs) are sent.

### Options

[*ipv4-addr*|*ipv6-addr*]

[addr-mask *ipv4-addr* ]

[filter none|debug|all|not-info|critical]

[ max-msg-size *484-65535* ]

[ max-msg-size *484-65535* ]

[ port-mask *tcp-udp port* ]

[ retries *0 - 255* ]

[ taglist *tag_name* ]

[ timeout *0 - 2147483647* ]

[ udp-port *port-number* ]

The `show snmp-server` command displays the current SNMP policy configuration, including SNMP communities, network security notifications, link-change traps, trap receivers (including the IPv4 or IPv6 address) that can receive SNMPv1 and SNMPv2c traps, and the source IP (interface) address used in IP headers when sending SNMP notifications (traps and informs) or responses to SNMP requests.

### Example output

The `show snmpv3 targetaddress` command displays the configuration (including the IPv4 or IPv6 address) of the SNMPv3 management stations to which notification messages are sent.

```
Switch(config)# show snmpv3 targetaddress

 snmpTargetAddrTable [rfc2573]

  Target Name      IP Address               Parameter
  ---------------- ------------------------ ----------
  1                15.29.17.218             1
  2                15.29.17.219             2
  PP.217           15.29.17.217             marker_p
  PP.218           2620:0:260:211
                  :217:a4ff:feff:1f70 1 marker_p
```

[1] An IPv6 Address is displayed on two lines

# IP preserve for IPv6

IPv6 supports the IP preserve feature, which allows you to copy a configuration file from a TFTP server to multiple switches without overwriting the IPv6 address and subnet mask on VLAN 1 (the default VLAN) in each switch, and the gateway IPv6 address assigned to the switch.

## ip preserve

**Syntax**

`ip preserve`

**Description**

Enter the `ip preserve` statement at the end of the configuration file to be downloaded from a TFTP server. (You do not invoke IP preserve by entering a command from the CLI.)

**Usage**

Entering an `ip preserve` statement as the last line in a configuration file stored on a TFTP server allows you to download and execute the file as the startup-config file on an IPv6 switch. When the switch reboots, the configuration settings in the downloaded file are implemented without changing the IPv6 address and gateway assigned to the switch

**Example output**

To enter IP preserve in a configuration file

```
; J8697A Configuration Editor; Created on release #K.15.xx
hostname "Switch"
time daylight-time-rule None
   *
   *
   *
   *
   *
   *
password manager
password operator
ip preserve
```

## Downloading an IP preserve configuration file to an IPv6-based switch

To copy the file as the new startup-config file on a switch, enter the TFTP `copy` command, as described in **SNMP management for IPv6** on page 45,

When you download an IP Preserve configuration file, the following rules apply:

- If the switch's current IPv6 address for VLAN 1 was statically configured and not dynamically assigned by a DHCP/Bootp server, the switch reboots and retains its current IPv6 address, subnet mask, and gateway address. All other configuration settings in the downloaded configuration file are applied.
- If the switch's current IPv6 address for VLAN 1 was assigned from a DHCP server and not statically configured, IP preserve is suspended. The IPv6 addressing specified in the downloaded configuration file is implemented when the switch copies the file and reboots.
    - If the downloaded file specifies DHCP/Bootp as the source for the IPv6 address of VLAN 1, the switch uses the IPv6 address assigned by the DHCP/Bootp server.
    - If the file specifies a dedicated IPv6 address and subnet mask for VLAN 1 and a gateway IPv6 address, the switch implements these settings in the startup-config file.

## Verifying how IP preserve was implemented in a switch

After the switch reboots, enter the `show run` command. The following example shows all configurations settings which have been copied into the startup-config file except for the IPv6 address of VLAN 1 (2001:db8::214:c2ff:fe4c:e480) and the default IPv6 gateway (2001:db8:0:7::5), which were retained.

If a switch received its IPv6 address from a DHCP server, the "ip address" field under "vlan 1" would display `dhcp-bootp`.

**Configuration file with dedicated IP addressing**

```
Switch(config)# show run

Running configuration:

; J8715A Configuration Editor; Created on release #K.14.01

hostname " Switch"
module 1 type J8702A
module 2 type J8705A
trunk 11-12 Trk1 Trunk
ip default-gateway 2001:db8:0:7::5
snmp-server community "public" Unrestricted
vlan 1
   name "DEFAULT_VLAN"
   untagged 1-10,13-24,1-24,Trk1
   ip address 2001:db8::214:c2ff:fe4c:e480
exit
spanning-tree Trk1 priority 4
password manager
password operator
```

NOTE

Because the switch's IPv6 address and default gateway were statically configured (not assigned by a DHCP server), when the switch boots up with the IP Preserve startup configuration file (see **IP preserve for IPv6** on page 47), its current IPv6 address, subnet mask, and default gateway are not changed. If a switch's current IP address was acquired from a DHCP/Bootp server, the IP Preserve statement is ignored and the IP addresses in the downloaded configuration file are implemented.

**More information**

For more information on how to use the IP preserve feature, see the current *ArubaOS-Switch Basic Operation Guide*.

# View the neighbor cache

ND occurs when there is communication between IPv6 devices on a VLAN. A neighbor destination is reachable from a given source address if a confirmation (neighbor solicitation) has been received at the source verifying that traffic has been received at the destination. The neighbor cache retains data for a given neighbor until the entry times out. You can view and clear the contents of the neighbor cache using the commands described in this section. For more on this topic, see **Neighbor Discovery (ND)** on page 27.

## Clear the neighbor cache

When there is an event such as a topology change or an address change, the neighbor cache might have too many entries to allow efficient use. Also, if an unauthorized client is answering DAD or normal neighbor solicitations with invalid replies, the neighbor cache might contain a large number of invalid entries and communication with some valid hosts might fail, the `show ipv6 neighbors` command output might become too cluttered to efficiently read, or both. In such cases, the fastest way to restore optimum traffic movement on a VLAN might be to statically clear the neighbor table instead of waiting for the unwanted entries to time out.

This chapter describes management security features that are IPv6 counterparts of IPv4 management security features on the switches.

NOTE

All commands previously in the Summary of commands table are indexed under the entry *Command syntax*.

This chapter describes the following IPv6-enabled management security features:

- Authorized IP Managers for IPv6
- Secure Shell for IPv6
- Secure Copy and Secure File Transfer Protocol for IPv6

# ipv6 authorized-managers

To configure one or more IPv6-based management stations to access the switch using the authorized IP managers feature, enter the `ipv6 authorized-managers` command.

**Syntax**

`ipv6 authorized-managers` *`ipv6-addr ipv6-mask`* `[access [operator| manager]]`

`no ipv6 authorized-managers` *`ipv6-addr ipv6-mask`* `[access [operator| manager]]`

**Options**

Configures one or more authorized IPv6 addresses to access the switch, where:

**`ipv6-mask`**

Specifies the mask that is applied to an IPv6 address to determine authorized stations.

Default:

`FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF.`

**`access [operator|manager]`**

Specifies the level of access privilege granted to authorized stations. Applies only to access through Telnet, SSH, and SNMP (version 1, 2, and 3).

Default: `Manager`

**`access-method [all|ssh|telnet|web|snmp|tftp]`**

Configures access levels by access method and IP address. Each management method can have its own set of authorized managers.

Default: `All`

## Single station access configuration

To authorize only one IPv6-based station for access to the switch:

Enter the IPv6 address of the station and set the mask to **FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF**.

NOTE

If you do not enter a value for the `ipv6-mask` parameter when you configure an authorized IPv6 address, the switch automatically uses **FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF** as the default mask.

If you have 10 or fewer management and/or operator stations for which you want to authorize access to the switch, it might be more efficient to configure them by entering each IPv6 address with the default mask in a separate `ipv6 authorized-managers` command.

When used in a mask, "**FFFF**" specifies that each bit in the corresponding 16-bit (hexadecimal) block of an authorized station's IPv6 address must be identical to the same "on" or "off" setting in the IPv6 address entered in the `ipv6 authorized-managers` command. (The binary equivalent of **FFFF** is 1111 1111 1111 1111, where**1** requires the same "on" or "off" setting in an authorized address.)

**Configuring single station access**

As shown below, if you configure a link-local IPv6 address of FE80::202:B3FF:FE1E:8329 with a mask of **FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF**, only a station having an IPv6 address of FE80::202:B3FF:FE1E:8329 has management access to the switch.

**Table 2:** *Mask for configuring a single authorized IPv6 manager station*

| | 1st block | 2nd block | 3rd block | 4th block | 5th block | 6th block | 7th block | 8th block | Manager- or operator-level access |
|---|---|---|---|---|---|---|---|---|---|
| IPv6 mask | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | The “FFFF” in each hexadecimal block of the mask specifies that only the exact value of each bit in the corresponding block of the IPv6 address is allowed. This mask allows management access only to a station having an IPv6 address of FE80::202:B3FF:FE1E: 8329. |
| IPv6 address | FE80 | 0000 | 0000 | 0000 | 202 | B3FF | FE1E | 8329 | |

## Multiple station access configuration

To authorize multiple stations to access the switch without having to re–enter the `ipv6 authorized-managers` command for each station, carefully select the IPv6 address of an authorized IPv6 manager and an associated mask to authorize a range of IPv6 addresses.

## Viewing an authorized IP managers configuration

**Procedure**

1. Use `show ipv6 authorized-managers` command to list the IPv6 stations authorized to access the switch.
2. Analyze the masks displayed in the following table . This shows the IPv6 addresses they authorize.

**Table 3:** *How masks determine authorized IPv6 manager addresses*

| Mask | Authorized IPv6 addresses | Number of authorized addresses |
|---|---|---|
| FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFC | 2001:db8:0:7::4 through 2001:db8:0:7::7 | 4 |
| FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFE | 2001:db8::a:1c:e3:2 and 2001:db8::a:1c:e3:3 | 2 |
| FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF | 2001:db8::214:c2ff:fe4c:e480 | 1 |
| FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FF00 | 2001:db8::0 through 2001:db8::FF | 256 |

**show ipv6 authorized-managers**

```
Switch# show ipv6 authorized-managers

 IPv6 Authorized Managers
---------------------------------------

 Address : 2001:db8:0:7::5
 Mask    : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
 Access  : Manager

 Address : 2001:db8::a:1c:e3:3
 Mask    : ffff:ffff:ffff:ffff:ffff:ffff:ffff:fffe
 Access  : Manager

 Address : 2001:db8::214:c2ff:fe4c:e480
 Mask    : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
 Access  : Manager

 Address : 2001:db8::10
 Mask    : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ff00
 Access  : Operator
```

## Commands to authorize manager access

The following IPv6 commands authorize manager-level access for one link-local station at a time. When you enter a link-local IPv6 address with the `ipv6 authorized-managers` command, you must also enter a VLAN ID in the format: `%vlan`*vlan-id*.

```
Switch(config)# ipv6 authorized-managers fe80::07be:44ff:fec5:c965%vlan2
```

```
Switch(config)# ipv6 authorized-managers fe80::070a:294ff:fea4:733d%vlan2
```

```
Switch(config)# ipv6 authorized-managers fe80::19af:2cff:fe34:b04a%vlan5
```

The following `ipv6 authorized-managers` command authorizes a single, automatically generated (EUI-64) IPv6 address with manager-level access privilege:

```
Switch(config)# ipv6 authorized-managers ::223:04ff:fe03:4501 ::ffff:ffff:ffff:ffff
```

**Default IPv6 mask**

If you do not enter an `ipv6-mask` value when you configure an authorized IPv6 address, the switch automatically uses FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF as the default IPv6 mask. Also, if you do not specify an **access** value to grant either manager- or operator-level access, by default, the switch assigns manager access.

```
Switch# ipv6 authorized-managers 2001:db8::a8:1c:e3:69
Switch# show ipv6 authorized-managers

IPv6 Authorized Managers
--------------------------

Address : 2001:db8::a8:1c:e3:69
Mask    : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
Access  : Manager
```

## Editing an existing authorized IP manager entry

To change the mask or access level for an existing authorized IP manager entry, enter the IPv6 address with the new values. Any parameters not included in the command are reset to their default values.

**Editing an existing authorized IP manager entry**

The following command replaces the existing mask and access level for IPv6 address 2001:DB8::231:17FF:FEC5:C967 with **FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FF00** and **operator**:

```
Switch(config)# ipv6 authorized-managers 2001:db8::231:17ff:fec5:c967
ffff:ffff:ffff:ffff:ffff:ffff:ffff:ff00 access operator
```

The following command replaces the existing mask and access level for IPv6 address 2001:DB8::231:17FF:FEC5:3E61 with **FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF** and **manager** (the default values). It is not necessary to enter either of these parameters:

```
Switch(config)# ipv6 authorized-managers 2001:db8::a05b:17ff:fec5:3f61
```

## Deleting an authorized IP manager entry

Enter only the IPv6 address of the configured authorized IP manager station that you want to delete with the `no` form of the command.

**Deleting an authorized IP manager entry**

```
Switch(config)# no ipv6 authorized-managers 2001:db8::231:17ff:fec5:3e61
```

# SSH for IPv6

Beginning with software release K.14.01, SSH for IPv4 and IPv6 operate simultaneously with the same command set. Both are enabled in the default configuration, and are controlled together by the same command set. SSH for IPv6 provides the same Telnet-like functions through encrypted, authenticated transactions as SSH for IPv4. SSH for IPv6 provides CLI (console) access and secure file transfer functionality. The following types of transactions are supported:

- Client public-key authenticationPublic keys from SSH clients are stored on the switch. Access to the switch is granted only to a client whose private key matches a stored public key.
- Password-only client authenticationThe switch is SSH-enabled but is not configured with the login method that authenticates a client's public-key. Instead, after the switch authenticates itself to a client, users connected to

the client authenticate themselves to the switch by providing a valid password that matches the operator- and/or manager-level password configured and stored locally on the switch or on a RADIUS or TACACS+ server.

- Secure Copy (SCP) and Secure File Transfer Protocol (SFTP) client applicationsYou can use either one SCP session or one SFTP session at a given time to perform secure file transfers to and from the switch.

By default, SSH is automatically enabled for IPv4 and IPv6 connections on a switch. Use the `ip ssh` command options to reconfigure the default SSH settings used in SSH authentication for IPv4 and IPv6 connections:

- TCP port number
- timeout period
- file transfer
- MAC type
- cipher type
- listening port

# ip ssh

### Syntax

`ip ssh`

`no ip ssh`

### Description

Enables SSH for on the switch for both IPv4 and IPv6, and activates the connection with a configured SSH server (RADIUS or TACACS+). The `no` form of the command disables SSH on the switch.

### Options

**`cipher`** ***`cipher-type`***

Specify a cipher type to use for connection.

Valid types are:

- aes128–cbc
- 3des-cbc
- aes192–cbc
- aes256–cbc
- rijndael-cbc@lysator.liu.se
- aes128–ctr
- aes192–ctr
- aes256–ctr

Default: All cipher types are available.

Use the `no` form of the command to disable a cipher type.

**`filetransfer`**

Enables SSH on the switch to connect to an SCP or SFTP client application to transfer files to and from the switch over IPv4 or IPv6.

Default: Disabled

NOTE: Enabling `filetransfer` automatically disables TFTP client and TFTP server functionality.

**mac *MAC-type***

Allows configuration of the set of MACs that can be selected. Valid types are:

- hmac-md5
- hmac-sha1
- hmac-sha1–96
- hmac-md5–96

Default: All MAC types are available.

Use the `no` form of the command to disable a MAC type.

**port [*1 - 65535*|default]**

TCP port number used for SSH sessions in IPv4 and IPv6 connections

Default: 22.

Valid port numbers are from 1 to 65535, except for port numbers 23, 49, 80, 280, 443, 1506, 1513, and 9999, which are reserved for other subsystems.

**public-key [manager|operator]*keystring***

Store a client-generated key for public-key authentication.

**manager**

Allows manager-level access using SSH public-key authentication.

**operator**

Allows operator-level access using SSH public-key authentication.

***keystring***

A legal SSHv2 (RSA or DSA) public key. The text string for the public key must be a single-quoted token. If the keystring contains double quotes, it can be quoted with single quotes ('key-string'). The following restrictions for a keystring apply:-

- A keystring cannot contain both single and double quotes.
- A keystring cannot have extra characters, such as a blank space or a new line. (To improve readability, you can add a backlash at the end of each line.)

For more information on configuring and using SSH public keys to authenticate SSH clients connecting to the switch, see chapter "Configuring Secure Shell" in the latest *Access Security Guide* for your switch.

**timeout *5 - 120***

Time out value allowed to complete an SSH authentication and login on the switch.

Default: 120 seconds.

**listen [oobm|data|both]**

The `listen` parameter is available only on switches that have a separate OOBM port. Values for this parameter are:

**oobm**

Inbound SSH access is enabled only on the OOBM port.

**data**

Inbound SSH access is enabled only on the data ports.

**both**

Inbound SSH access is enabled on both the OOBM port and on the data ports. This is the default value.

**Restrictions**

For both IPv4 and IPv6, the switch supports only SSH version 2. You cannot set up an SSH session with a client device running SSH version 1.

The `listen` parameter is not available on switches that do not have a separate OOBM port.

## show ip ssh

### Syntax

`show ip ssh`

### Description

To verify an SSH configuration and display all SSH sessions running on the switch, enter the `show ip ssh` command. Information on all current SSH sessions (IPv4 and IPv6) is displayed.

### Restrictions

With SSH running, the switch supports one console session and up to five other SSH and Telnet (IPv4 and IPv6) sessions. WebAgent sessions are also supported, but are not displayed in show ip ssh output.

### Example output

SSH configuration display

```
Switch# show ip ssh

  SSH Enabled     : Yes                Secure Copy Enabled : No
  TCP Port Number : 22                 Timeout (sec)       : 120
  Host Key Type   : RSA                Host Key Size       : 2048

  Ciphers : aes128-cbc,3des-cbc,aes192-cbc,aes256-cbc,
            rijndael-cbc@lysator.liu.se,aes128-ctr,aes192-ctr,
            aes256-ctr
  MACs    : hmac-md5,hmac-sha1,hmac-sha1-96,hmac-md5-96

  Ses Type     | Source IP                                  Port
  --- -------- + ---------------------------------------- -----
  1   console  |
  2   ssh      | 10.168.31.114                              1722
  3   inactive |
  4   inactive |
  5   inactive |
  6   inactive |
```

# ip ssh filetransfer

### Syntax

`ip ssh filetransfer`

`no ip ssh filetransfer`

### Description

Enables SSH on the switch to connect to an SCP or SFTP client application to transfer files to and from the switch.

### Usage

Use the `no ip ssh filetransfer` command to disable the switch's ability to perform secure file transfers with an SCP or SFTP client, without disabling SSH on the switch.

# Authorized IP managers for IPv6

The authorized IP managers feature uses IP addresses and masks to determine which stations (PCs or workstations) can access the switch through the network. This feature supports switch access through:

- Telnet and other terminal emulation applications
- SNMP (with a correct community name)
- SSH
- TFTP

## Features of authorized IP managers

As with the configuration of IPv4 management stations, the authorized IP managers for IPv6 feature allow you to specify the IPv6-based stations that can access the switch.

You can configure up to 100 authorized IPv4 and IPv6 manager addresses on a switch, where each address applies to either a single management station or a group of stations. Each authorized manager address consists of an IPv4 or IPv6 address and a mask that determines the individual management stations that are allowed access.

- You can configure up to 100 authorized IPv4 and IPv6 manager addresses on a switch, where each address applies to either a single management station or a group of stations. Each authorized manager address consists of an IPv4 or IPv6 address and a mask that determines the individual management stations that are allowed access.
    - Using the `ip authorized-managers`, you can configure authorized IPv4 manager addresses command. For more information, see "Using Authorized IP Managers" in the *Access Security Guide*.
    - Using the `ipv6 authorized-managers`, you can configure authorized IPv6 manager addresses command.
- You can block all IPv4-based or all IPv6-based management stations from accessing the switch by entering the following commands:
    - To block access to all IPv4 manager addresses while allowing access to IPv6 manager addresses, enter the `ip authorized-managers 0.0.0.0` command.
    - To block access to all IPv6 manager addresses while allowing access to IPv4 manager addresses, enter the `ipv6 authorized-managers ::` command. (The double colon represents an IPv6 address that consists of all zeros: **0:0:0:0:0:0:0:0**.)
- You configure each authorized manager address with manager- or operator-level privilege to access the switch.
    - Manager privilege allows full access to all console interface screens for viewing, configuring, and all other operations available in these interfaces.
    - Operator privilege allows read-only access from the console interfaces.
- When you configure station access to the switch using the authorized IP managers feature, the settings take precedence over the access configured with local passwords, TACACS+ servers, RADIUS-assigned settings, port-based (802.1X) authentication, and port security settings.

As a result, the IPv6 address of a networked management device must be configured with the authorized IP managers feature before the switch can authenticate the device using the configured settings from other access security features. If the authorized IP managers feature disallows access to the device, access is denied. Therefore, with authorized IP managers configured, logging in with the correct passwords is not sufficient to access a switch through the network unless the station requesting access is also authorized in the switch's authorized IP managers configuration.

# ipv6 mask

**Syntax**

```
ipv6 mask
```

**Description**

This parameter controls how the switch uses an IPv6 address to determine the IPv6 addresses of authorized manager stations on your network. For example, you can specify a mask that authorizes:

- Single station access
- Multiple station access

**Usage**

Mask configuration is a method for determining the valid IPv6 addresses that are authorized for management access to the switch. In the authorized IP managers feature, the mask serves a different purpose than an IPv6 subnet mask and is applied in a different manner.

## About configuring multiple station access

As shown in the following table, if a bit in any of the 4-bit binary representations of a hexadecimal value in a mask is "on" (set to 1), the corresponding bit in the IPv6 address of an authorized station must match the "on" or "off" setting of the same bit in the IPv6 address you enter with the `ipv6 authorized-managers` command.

Conversely, in a mask, a "0" binary bit means that either the "on" or "off" setting of the corresponding IPv6 bit in an authorized address is valid and does not have to match the setting of the same bit in the specified IPv6 address.

**Table 4:** *Hexadecimal mask values and binary equivalents*

| Hexadecimal value in an IPv6 mask | Binary equivalent |
|---|---|
| 0 | 0000 |
| 1 | 0001 |
| 2 | 0010 |
| 3 | 0011 |
| 4 | 0100 |
| 5 | 0101 |
| 6 | 0110 |
| 7 | 0111 |
| 8 | 1000 |
| 9 | 1001 |
| A | 1010 |
| B | 1011 |
| C | 1100 |

*Table Continued*

| Hexadecimal value in an IPv6 mask | Binary equivalent |
|---|---|
| D | 1101 |
| E | 1110 |
| F | 1111 |

#### Configuring multiple station access

The following table shows an example in which a mask that authorizes switch access to four management stations is applied to the IPv6 address: `2001:DB8:0000:0000:244:17FF:FEB6:D37D`.

The mask is: `FFFF:FFFF:FFFF:FFF8:FFFF:FFFF:FFFF:FFFC`.

### Table 5: *Mask for configuring a single authorized IPv6 manager station*

| | 1st block | 2nd block | 3rd block | 4th block | 5th block | 6th block | 7th block | 8th block | Manager- or operator-level access |
|---|---|---|---|---|---|---|---|---|---|
| IPv6 mask | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | The "F" value in the first 124 bits of the mask specifies that only the exact value of each corresponding bit in an authorized IPv6 address is allowed. However, the "C" value in the last four bits of the mask allows four possible combinations (D37C, D37D, D37E, and D37F) in the last block of an authorized IPv6 address. |
| IPv6 address | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37D | |

As shown in the table, if you use a mask of `FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFC` with an IPv6 address, you can authorize four IPv6-based stations to access the switch. In this mask, all bits except the last two are set to **1** ("on"); the binary equivalent of hexadecimal C is 1100.

### Table 6: *Mask for configuring a single authorized IPv6 manager station*

| | 1st block | 2nd block | 3rd block | 4th block | 5th block | 6th block | 7th block | 8th block |
|---|---|---|---|---|---|---|---|---|
| IPv6 mask | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF |
| IPv6 address entered with the `ipv6 authorized-managers` command | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37D |

*Table Continued*

| | 1st block | 2nd block | 3rd block | 4th block | 5th block | 6th block | 7th block | 8th block |
|---|---|---|---|---|---|---|---|---|
| Other authorized IPv6 addresses | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37C |
| | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37E |
| | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37F |

**Configuring multiple station access**

This table shows an example in which a mask that authorizes switch access to four management stations is applied to the IPv6 address: `2001:DB8:0000:0000:244:17FF:FEB6:D37D`.

The mask is: `FFFF:FFFF:FFFF:FFF8:FFFF:FFFF:FFFF:FFFC`.

**Table 7:** *Mask for configuring a single authorized IPv6 manager station*

| | 1st block | 2nd block | 3rd block | 4th block | 5th block | 6th block | 7th block | 8th block | Manager- or operator-level access |
|---|---|---|---|---|---|---|---|---|---|
| IPv6 mask | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | The "F" value in the first 124 bits of the mask specifies that only the exact value of each corresponding bit in an authorized IPv6 address is allowed. However, the "C" value in the last four bits of the mask allows four possible combinations (D37C, D37D, D37E, and D37F) in the last block of an authorized IPv6 address. |
| IPv6 address | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37D | |

As shown, if you use a mask of `FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFC` with an IPv6 address, you can authorize four IPv6-based stations to access the switch. In this mask, all bits except the last two are set to **1** (“on”); the binary equivalent of hexadecimal C is 1100.

**Table 8:** *How a mask determines four authorized IPv6 manager addresses*

| Last block in mask: FFFC<br><br>Last block in IPv6 address: D37D | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit numbers | Bit 15 | Bit 14 | | Bit 13 | Bit 12 | Bit 11 | Bit 10 | | Bit 9 | Bit 8 | Bit 7 | Bit 6 | | Bit 5 | Bit 4 | Bit 3 | Bit 2 | | Bit 1 | Bit 0 |
| Bit value | | | F | | | | | F | | | | | F | | | | | C | | |

*Table Continued*

| Last block in mask: FFFC<br><br>Last block in IPv6 address: D37D | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FFFC: Last block in mask | 1 | 1 | | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | | 0 | 0 |
| D37D: Last block in IPv6 address | 1 | 1 | | 0 | 1 | 0 | 0 | | 1 | 1 | 0 | 1 | | 1 | 1 | 1 | 1 | | 0 | 1 |
| Bit setting: | 1 = On | | | | 0 = Off | | | | | | | | | | | | | | | |

Therefore, this mask requires the first corresponding 126 bits in an authorized IPv6 address to be the same as in the specified IPv6 address: 2001:DB8:0000:0000:244:17FF:FEB6:D37C. However, the last 2 bits are set to **0** ("off") and allow the corresponding bits in an authorized IPv6 address to be either "on" or "off". As a result, only four IPv6 addresses are allowed access.

### **Table 9:** *Mask for configuring a single authorized IPv6 manager station*

| | 1st block | 2nd block | 3rd block | 4th block | 5th block | 6th block | 7th block | 8th block |
|---|---|---|---|---|---|---|---|---|
| IPv6 mask | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF |
| IPv6 address entered with the `ipv6 authorized-managers` command | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37D |
| Other authorized IPv6 addresses | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37C |
| | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37E |
| | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37F |

The table above shows an example in which a mask is applied to the IPv6 address: `2001:DB8:0000:0000:244:17FF:FEB6:D37D/64`. The specified mask `FFFF:FFFF:FFFF:FFF8:FFFF:FFFF:FFFF:FFFF` configures eight management stations as authorized IP manager stations.

In this example, the IPv6 mask is applied as follows:

- Eight management stations in different subnets are authorized by the value of the fourth block (`FFF8`) in the 64-bit prefix ID (`FFFF:FFFF:FFFF:FFF8`) of the mask. (The fourth block of the prefix ID is often used to define subnets in an IPv6 network.)The binary equivalent of `FFF8` that is used to specify valid subnet IDs in the IPv6 addresses of authorized stations is `1111 1111 1111 1000`.The three "off" bits (1000) in the last part of this block (`FFF8`) of the mask allow for eight possible authorized IPv6 stations: 2001:DB8:0000:0000:244:17FF:FEB6:D37D 2001:DB8:0000:0001:244:17FF:FEB6:D37D 2001:DB8:0000:0002:244:17FF:FEB6:D37D 2001:DB8:0000:0003:244:17FF:FEB6:D37D2001:DB8:0000:0004:244:17FF:FEB6:D37D2001:DB8:0000:0005:

244:17FF:FEB6:D37D2001:DB8:0000:0006:244:17FF:FEB6:D37D2001:DB8:0000:0007:244:17FF:FEB6:D37D

- Each authorized station has the same 64-bit device ID (`244:17FF:FEB6:D37D`), because the value of the last four blocks in the mask is `FFFF` (binary value 1111 1111).

`FFFF` requires all bits in each corresponding block of an authorized IPv6 address to have the same "on" or "off" setting as the device ID in the specified IPv6 address. In this case, each bit in the device ID (last four blocks) in an authorized IPv6 address is fixed and can be only one value: 244:17FF:FEB6:D37D.

**Table 10:** *Mask for configuring authorized IPv6 Manager stations in different subnets*

| | 1st block | 2nd block | 3rd block | 4th block | 5th block | 6th block | 7th block | 8th block | Manager- or operator-level access |
|---|---|---|---|---|---|---|---|---|---|
| IPv6 mask | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | FFFF | In this example, the IPv6 mask allows up to four stations in different subnets to access the switch. This authorized IP manager configuration is useful if only management stations are specified by the authorized IPv6 addresses. For how the bitmap of the IPv6 mask determines authorized IP manager stations, see fix this — Example of How an ACL Filters Packets — |
| IPv6 address | 2001 | DB8 | 0000 | 0000 | 244 | 17FF | FEB6 | D37D | |

This table shows the bits in the fourth block of the mask that determine the valid subnets in which authorized stations with an IPv6 device ID of `244:17FF:FEB6:D37D` reside.

**Table 11:** *How a mask determines authorized IPv6 manager addresses by subnet*

| Fourth block in mask: FFF8 | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Fourth Block in Prefix ID of IPv6 Address: 0000 | | | | | | | | | | | | | | | | | | | | |
| Bit numbers | Bit 15 | Bit 14 | | Bit 13 | Bit 12 | Bit 11 | Bit 10 | | Bit 9 | Bit 8 | Bit 7 | Bit 6 | | Bit 5 | Bit 4 | Bit 3 | Bit 2 | | Bit 1 | Bit 0 |
| Bit value | | | F | | | | | F | | | | | F | | | | | 8 | | |
| FFFC: Last block in mask | 1 | 1 | | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 0 | | 0 | 0 |
| D37D:Last block in IPv6 address | 0 | 0 | | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | | 0 | 0 |
| Bit setting: | 1 = On | | 0 = Off | | | | | | | | | | | | | | | | | |

`FFF8` in the fourth block of the mask means that bits 3 to 15 of the block are fixed and, in an authorized IPv6 address, must correspond to the "on" and "off" settings shown for the binary equivalent 0000 in the fourth block of the IPv6 address. Conversely, bits 0 to 2 are variable and, in an authorized IPv6 address, may be either "on" (1) or "off" (0).

As a result, assuming that the seventh and eighth bytes (fourth hexadecimal block) of an IPv6 address are used as the subnet ID, only the following binary expressions and hexadecimal subnet IDs are supported in this authorized IPv6 manager configuration:

**Table 12:** *Binary equivalents of authorized subnet IDs (in hexadecimal)*

| Authorized subnet ID in fourth hexadecimal block of IPv6 address | Binary equivalent |
|---|---|
| 0000 | 0000 0000 |
| 0001 | 0000 0001 |
| 0002 | 0000 0010 |
| 0003 | 0000 0011 |
| 0004 | 0000 0100 |
| 0005 | 0000 0101 |
| 0006 | 0000 0110 |
| 0007 | 0000 0111 |

# SCP and SFTP for IPv6

You can take advantage of the SCP and SFTP client applications to provide a secure alternative to TFTP for transferring sensitive switch information, such as configuration files and login information, between the switch and an administrator workstation. Because SCP and SFTP run over an encrypted SSH session, you can use a secure SSH tunnel to:

- Transfer files and update E Switch software images.
- Distribute new software images with automated scripts that make it easier to upgrade multiple switches simultaneously and securely.

## ip ssh filetransfer

**Syntax**

`ip ssh filetransfer`

**Description**

This command is used to perform secure file transfers to and from IPv4 and IPv6 client devices

**Restrictions**

Enabling SSH file transfer disables TFTP and auto-TFTP operation. The switch supports one SFTP session or one SCP session at a time.

All files on the switch have read-write permission. However, several SFTP commands, such as `create` or `remove`, are not supported and return an error.

This chapter describes MLD snooping and the CLI commands available for configuring it and for viewing its status.

# Multicast addressing

Multicast addressing allows one-to-many or many-to-many communication among hosts on a network. Typical applications of multicast communication include audio and video streaming, desktop conferencing, collaborative computing, and similar applications.

# Multicast Listener Discovery(MLD)

MLD is an IPv6 protocol used on a local link for multicast group management. MLD operates in a manner similar to IGMP in IPv4 networks. MLD is enabled per VLAN and is analogous to the IPv4 IGMP protocol. In the factory default state (MLD disabled), the switch floods all IPv6 multicast traffic it receives on a given VLAN through all ports on that VLAN except the port receiving the inbound multicast traffic. Enabling MLD imposes management controls on IPv6 multicast traffic to reduce unnecessary bandwidth usage. MLD is configured per-VLAN. MLD can be configured using version 1 (MLDv1) or version 2 (MLDv2). MLDv2 introduces source-specific multicast in which the only packets delivered to the receiver are those that originate from a specified source address requested by the receiver. The receiver indicates interest in receiving traffic to a multicast address and additionally can indicate interest in receiving traffic from only one specified source sending to that multicast address. This reduces the amount of multicast routing information that should be maintained.These options are available for MLDv1 and MLDv2:

- Query interval-the time interval between general queries sent by the querier.
- Query Max Response Time-the amount of time to wait for a response to a query.
- Last Member Query Interval-the amount of time the querier waits to receive a response from members to a group-specific query message. It also specifies the amount of time between successive group-specific query messages.
- Robustness-the number of times to retry a query.
- Fast Learn-enables the port to learn group information when there is a topology change.

# MLD snooping

There are several roles that network devices may play in an IPv6 multicast environment:

**MLD host**

A network node that uses MLD to "join" (subscribe to) one or more multicast groups.

**Multicast router**

A router that routes multicast traffic between subnets.

**Querier**

A switch or multicast router that identifies MLD hosts by sending out MLD queries to which the MLD hosts respond.

A network node that acts as a source of IPv6 multicast traffic is only an indirect participant in MLD snooping—it just provides multicast traffic, and MLD does not interact with it. (However, in an application like desktop conferencing a network node may act as both a source and an MLD host, but MLD interacts with that node only in its role as an MLD host.)

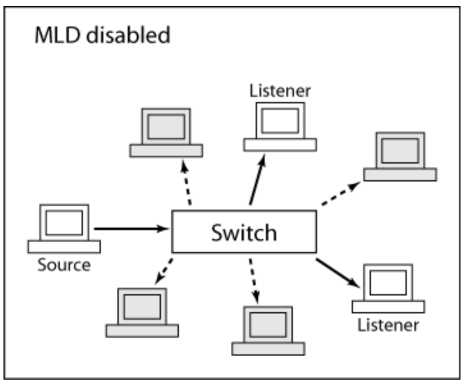
A source node creates multicast traffic by sending packets to a multicast address. In IPv6, addresses with the first 8 bits set (that is, "FF" as the first two characters of the address) are multicast addresses, and any node that listens to such an address will receive the traffic sent to that address. Application software running on the source and destination systems cooperates to determine what multicast address to use. (This is a function of the application software, not of MLD.)

For example, if several employees engage in a desktop conference across the network, they all need application software on their computers. At the start of the conference, the software on all the computers determines a multicast address of, for example, FF3E:30:2001:DB8::101 for the conference. Then any traffic sent to that address can be received by all computers listening on that address.

## MLD operation

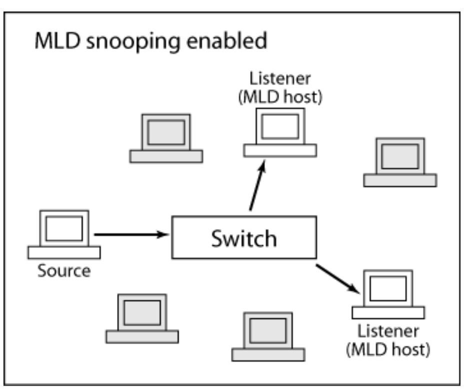
Multicast communication can take place without MLD, and by default, MLD is disabled. In that case, if a switch receives a packet with a multicast destination address, it floods the packet to all ports in the same VLAN (except the port that it came in on), as shown in the following figure. Any network nodes that are listening to that multicast address will see the packet; all other hosts ignore the packet.

**Figure 1:** *Without MLD, multicast traffic is flooded to all ports*




When MLD snooping is enabled on a VLAN, the switch acts to minimize unnecessary multicast traffic. If the switch receives multicast traffic destined for a given multicast address, it forwards that traffic only to ports on the VLAN that have MLD hosts for that address, as shown in the following figure. It drops that traffic for ports on the VLAN that have no MLD hosts (except for a few special cases explained below).

**Figure 2:** *With MLD snooping, traffic is sent to MLD hosts*




## Forwarding in MLD snooping

When MLD snooping is active, a multicast packet is handled by the switch as shown in the following list.

The packet is:

- Forwarded to ports that have nodes that have joined the packet's multicast address (that is, MLD hosts on that address packet)
- Forwarded toward the querier—If the switch is not the querier, the packet is forwarded out the port that leads to the querier.
- Forwarded toward any multicast routers—If there are multicast routers on the VLAN, the packet is forwarded out any port that leads to a router.
- Forwarded to administratively forwarded ports—The packet is forwarded through all ports set administratively to forward mode. (See the description of forward modes, below.)
- Dropped for all other ports.

Each individual port's forwarding behavior can be explicitly set using a CLI command to one of these modes:

**Auto (the default mode)**

The switch forwards packets through this port based on the MLD rules and the packet's multicast address. In most cases, this means that the switch forwards the packet only if the port connects to a node that is joined to the packet's multicast address (that is, to an MLD host). There is seldom any reason to use a mode other than "auto" in normal operation (though some diagnostics may use "forward" or "block" mode).

**Forward**

The switch forwards all IPv6 multicast packets through the port. This includes IPv6 multicast data and MLD protocol packets.

**Block**

The switch drops all MLD packets received by the port and blocks all outgoing IPv6 multicast packets through the port, except those packets destined for well-known IPv6 multicast addresses. This prevents IPv6 multicast traffic from moving through the port.

The switch floods all packets with "well-known" IPv6 multicast destination addresses through all ports. Well-known addresses are permanent addresses defined by the Internet Assigned Numbers Authority (IANA). IPv6 standards define any address beginning with FF0x/12 (binary 1111 1111 0000) as a well-known address.

# ipv6 mld

### Syntax

`ipv6 mld [enable|disable]`

`no ipv6 mld [enable|disable]`

### Description

The `no` form disables MLD snooping on a VLAN.

### Options

**`no ipv6 mld`**

This command will remove that option from the configuration and apply the default if one exists.

### Usage

This command must be issued in a **VLAN** context.

MLDv2 is disabled by default.

enable: Enables MLDv2 on a VLAN.

disable: Disables MLDv2 on a VLAN. The last-saved or the default MLD configuration is saved, whichever is most recent.

### Example input

This example shows how to enable or disable MLD snooping on a VLAN

To enable MLD snooping on VLAN 8:

```
Switch(vlan-8)# ipv6 mld enable
Switch# config
```

To disable MLD snooping on VLAN 8:

```
Switch(vlan-8)# no ipv6 mld
```

# ipv6 mld version

### Syntax

`ipv6 mld version` *`1-2`* `strict`

`no ipv6 mld version` *`1-2`* `strict`

### Description

This command sets the MLD protocol version to use. The `no` version of the command resets the version to the default, version 2.

### Usage

This command must be issued in a **VLAN** context.

### Example input

To set MLD to version 1 for VLAN 8 and version 2 for VLAN 9

```
Switch(vlan-8)# ipv6 mld version 1
Switch(vlan-8)# exit
Switch(config)# vlan 9
Switch(vlan-9)# ipv6 mld version 2
```

# ipv6 mld port

### Syntax

`ipv6 mld [auto` *`port-list`*`|blocked` *`port-list`*`|forward` *`port-list`*`]`

### Description

This command sets per-port traffic filters, which specify how each port should handle MLD traffic.

### Options

The following settings are allowed:

**`auto`**

Follows MLD snooping rules: packets are forwarded for joined groups.

**`blocked`**

All multicast packets are dropped, except that packets for well-known addresses are forwarded.

**`forward`**

All multicast packets are forwarded.

***`port-list`***

Specifies the affected port or range of ports.

The default value of the filter is `auto`

# Queries

The querier is a multicast router or a switch that periodically asks MLD hosts on the network to verify their multicast join requests. There is one querier for each VLAN, and all switches on the VLAN listen to the responses of MLD hosts to multicast queries and forward or block multicast traffic accordingly.

All of the HPE switches have the querier function enabled by default. If there is another device on the VLAN that is already acting as querier, the switch defers to that querier. If there is no device acting as querier, the switch enters an election state and negotiates with other devices on the network (if any) to determine which one will act as the querier.

The querier periodically sends general queries to MLD hosts on each multicast address that is active on the VLAN. The time that the querier waits between sending general queries is known as the query interval; the MLD standard sets the default query interval to 125 seconds.

Network nodes that wish to remain active as MLD hosts respond to the queries with join requests; in this way they continue to assert their presence as MLD hosts. The switch through which any given MLD host connects to the VLAN sees the join requests and continues forwarding traffic for that multicast address to the MLD host's port.

## ipv6 mld querier

**Syntax**

`ipv6 mld querier`

`no ipv6 mld querier`

**Description**

Enables the switch to act as querier on a VLAN. The querier function is enabled by default. If another switch or a multicast router is acting as the MLD querier on the VLAN, this switch defers to that device. If an acting querier stops performing the querier function, all querier-enabled switches and multicast routers on the VLAN enter an election to determine the next device to act as querier. The `no` form disables the switch from acting as querier on a VLAN.

**Usage**

This command must be issued in a **VLAN** context.

**Example input**

Configuring the querier

To disable the switch from acting as querier on VLAN 500:

```
Switch(vlan-500)# no ipv6 mld querier
```

To enable the switch to act as querier on VLAN 500:

```
Switch(vlan-500)# ipv6 mld querier
```

## ipv6 mld query-interval

**Syntax**

`ipv6 mld query-interval` *`60-31744`*

`no ipv6 mld query-interval` *`60-31744`*

**Description**

This command specifies the number of seconds between membership queries. The `no` form of the command sets the interval to the default of 125 seconds.

**Specifiers**

Default: 125 seconds.

**Usage**

This command must be issued in a **VLAN** context.

**Example input**

To set the query-interval to 300 seconds on ports in VLAN 500:

```
Switch(vlan-500)# ipv6 mld query-interval 300
```

## ipv6 mld query-max-response-time

**Syntax**

`ipv6 mld query-max-response-time` *`10 - 128`*

`no ipv6 mld query-max-response-time` *`10 - 128`*

**Description**

This command specifies the maximum amount of time to wait for a response to a query. The `no` form of the command sets the interval to the default of 10 seconds.

**Specifiers**

Default: 10 seconds

**Usage**

This command must be issued in a **VLAN** context.

**Example input**

To set the `query-max-response-time` to 30 seconds on ports on VLAN 500:

```
Switch(vlan-500)# ipv6 mld query-max-response-time 30
```

## ipv6 mld robustness

**Syntax**

`ipv6 mld robustness` *`1 - 8`*

**Description**

Specifies the number of times to retry a query. The `no` form of the command sets the interval to the default of 2.

**Specifiers**

Default: 2

**Usage**

This command must be issued in a **VLAN** context.

**Example input**

```
Switch(vlan-500)# ipv6 mld robustness 4
```

## ipv6 mld last-member-query-interval

**Syntax**

`ipv6 mld last-member-query-interval` *`1 - 2`*

`no ipv6 mld last-member-query-interval` *`1 - 2`*

**Description**

Sets the amount of time that the querier waits to receive a response from members to a group-specific query message. It also specifies the amount of time between successive group-specific query messages. The `no` form of the command sets the interval to the default of 1 second.

**Specifiers**

Default: 1 second.

**Usage**

This command must be issued in a **VLAN** context.

**Example input**

```
switch(vlan-500)# ipv6 mld last-member-query-interval 2
```

# ipv6 mld fastlearn

The Fast Learn option allows fast convergence of multicast traffic after a topology change. When a new port joins or moves to a forwarding state, MLD sends joins for the groups it maintains.

For MLDv1, a join is transmitted for each group if the switch is a nonquerier. If the switch is a querier, an MLDv1 query is sent to learn the group on that port.

For MLDv2, an IS_EX report is sent when the switch is a nonquerier. If the switch is a querier, an MLDv2 query is sent on the port to learn the group.

**Syntax**

```
ipv6 mld fastlearn <port-list|all>
```

```
no ipv6 mld fastlearn <port-list|all>
```

**Description**

This command enables fast learn on the specified ports in a VLAN. The `no` form of the command disables the fast learn function on the specified ports. The `all` option enabled or disables all ports.

**Specifiers**

Default: Disabled

**Example input**

```
switch(config)# ipv6 mld fastlearn 5 - 6
```

# Leaves

A node acting as an MLD host can be disconnected from a multicast address in two ways:

- It can stop sending join requests to the querier.This may happen if the multicast application quits or the node is removed from the network. If the switch goes for slightly more than two query intervals without seeing a join request from the MLD host, it stops sending multicast traffic for that multicast address to the MLD host's port.
- It can issue a "leave" request.This is done by the application software running on the MLD host. If the MLD host is the only node connected to its switch port, the switch sees the leave request and stops sending multicast packets for that multicast address to that port. (If there is more than one node connected to the port the situation is more complicated, as explained under **Fast leaves and forced fast leaves** on page 71.)

## Fast leaves and forced fast leaves

The fast leave and forced fast leave functions can help to prune unnecessary multicast traffic when an MLD host issues a leave request from a multicast address. Fast leave is enabled by default, and forced fast leave is disabled by default. Both functions are applied to individual ports.

Which function to use depends on whether a port has more than one node attached to it, as follows:

- If a port has only one node attached to it, when the switch sees a leave request from that node (an MLD host) it knows that it does not need to send any more multicast traffic for that multicast address to the host's port.
- If fast leave is enabled (the default setting), the switch stops sending the multicast traffic immediately.
- If fast leave is disabled, the switch continues to look for join requests from the host in response to group-specific queries sent to the port.
- The interval during which the switch looks for join requests is brief and depends on the forced fast leave setting:
- If forced fast leave is enabled for the port, it is equal to the "forced fast leave interval" (typically several seconds or less).
- If forced fast leave is disabled for the port, the period is about 10 seconds (governed by the MLD standard).
- When this process has completed, the multicast traffic for the group will stop (unless the switch sees a new join request).
- If a single port has multiple nodes attached to it, a leave request from one of those nodes (an MLD host) does not provide enough information for the switch to stop sending multicast traffic to the port. In this situation, the fast leave function does not operate. The switch continues to look for join requests from any MLD hosts connected to the port in response to group-specific queries sent to the port.As in the case described above for a single-node port that is not enabled for fast leave, the interval during which the switch looks for join requests is brief and depends on the forced fast leave setting:
- If forced fast leave is enabled for the port, it is equal to the "forced fast leave interval" (typically several seconds or less).
- If forced fast leave is disabled for the port, the period is about 10 seconds (governed by the MLD standard).
- When this process has completed, the multicast traffic for the group will stop unless the switch sees a new join request. This reduces the number of multicast packets forwarded unnecessarily.

## ipv6 mld fastleave

**Syntax**

`ipv6 mld fastleave` *`port-list`*

`no ipv6 mld fastleave` *`port-list`*

**Description**

Enables the fast leave function on the specified ports in a VLAN. The `no` form disables the fast leave function on the specified ports in a VLAN.

**Usage**

This command must be issued in a **VLAN** context.

**Example input**

To disable fast leave on ports in VLAN 8:

```
switch(vlan-8)# no ipv6 mld fastleave 14-15
```

To enable fast leave on ports in VLAN 8:

```
switch(vlan-8)# ipv6 mld fastleave 14-15
```

## ipv6 mld forcedfastleave

**Syntax**

`ipv6 mld forcedfastleave` *`port-list`*

`no ipv6 mld forcedfastleave` *`port-list`*

**Description**

Enables the forced fast leave function on the specified ports in a VLAN. The `no` form disables the forced fast leave function on the specified ports in a VLAN

**Specifiers**

Default: Disabled

**Usage**

This command must be issued in a **VLAN** context.

**Example input**

To enable forced fast leave on ports in VLAN 8:

```
switch(vlan-8)# ipv6 mld forcedfastleave 19-20
```

To disable forced fast leave on ports in VLAN 8:

```
switch(vlan-8)# no ipv6 mld forcedfastleave 19-20
```

# Current MLD status

The following information is shown for each VLAN that has MLD snooping enabled:

- VLAN ID number and name
- Querier address: IPv6 address of the device acting as querier for the VLAN.
- Querier up time: Length of time in seconds that the querier has been acting as querier.
- Querier expiry time: If this switch is the querier, this is the amount of time until the switch sends the next general query. If this switch is not the querier, this is the amount of time in seconds until the current querier is considered inactive (after which a new querier election is held).
- Ports with multicast routers: Ports on the VLAN that lead toward multicast routers (if any).
- Multicast group address information for each active group on the VLAN, including:
    - Multicast group address.
    - Type of tracking for multicast joins: standard or filtered.
        - If MLD snooping is enabled, port-level tracking results in filtered groups.
        - If MLD snooping is not enabled, joins result in standard groups being tracked by this device.
        - In addition, if hardware resources for multicast filtering are exhausted, new joins may result in standard groups even though MLD snooping is enabled.
        - MLD version number (MLDv2 display only)
        - Mode—INCLUDE or EXCLUDE (MLDv2 only): when INCLUDE is displayed, the host has requested specific source/group pairs. When EXCLUDE is displayed, the host has requested all sources for a group except for a specified list of sources to exclude.
- Uptime: The length of time the group has been joined.
- Expire time: Time until the group expires if no joins are seen.
- The ports that have joined the multicast group.

## show ipv6 mld

**Syntax**

```
show ipv6 mld
```

**Description**

Displays MLD status information for all VLANs on the switch that have MLD configured.

**Syntax**

```
show ipv6 mld vlan <vid>
```

**Description**

Displays MLD status for the specified VLAN.

**Example output**

Displaying the MLD configuration for all static VLANs on the switch

```
Switch# show ipv6 mld

  MLD Service Protocol Info

  Total vlans with MLD enabled               : 1
  Current count of multicast groups joined   : 6

  VLAN ID              : 500
  VLAN Name            : VLAN500
  MLD Version          : 2
  MLD Interface State  : Querier
  Querier Address      : fe80::b25a:daff:fe97:6280 [this switch]
          Version      : 2
          Uptime       : 18m 38s
          Expires      : 0m 37s  Querier Up Time : 1h:37m:20s

  Ports with multicast routers :


  Active Group Addresses                    Tracking  Vers  Mode  Uptime  Expires
  ----------------------------------------  ---- ----------  -------------------------
  ff1e::5                                   Filtered  2     EXC   3m 7s   2m 57s
  ff1e::6                                   Filtered  2     EXC   3m 8s   3m 0s
  ff1e::7                                   Filtered  2     EXC   3m 8s   2m 56s
  ff1e::8                                   Filtered  2     EXC   3m 8s   2m 57s
  ff1e::9                                   Filtered  2     EXC   3m 8s   2m 58s
  ff1e::a                                   Filtered  2     EXC   3m 8s   2m 52s

Switch#
```

# Current MLD configuration

The following information applies to all MLD-enabled VLANs:

**Control unknown multicast**

If this is set to YES, any IPv6 multicast packets that are not joined by an MLD host are sent only to ports that have detected a multicast router or ports that are administratively forwarded.

If this is set to NO (or if MLD snooping is disabled), unjoined IPv6 multicast packets are flooded out all ports in the VLAN.

**Forced fast leave timeout**

Interval between an address-specific query and a forced fast leave (assuming no response), in tenths of seconds.

**For each VLAN that has MLD enabled:**

- Whether MLD is enabled on the VLAN (default NO, but the VLAN will not show up on this list unless MLD is enabled).
- Whether the switch can act as querier for the VLAN (default YES).
- The MLD version (1 or 2)

**MLD configuration for a specific VLAN**

```
Switch# show ipv6 mld vlan 500 config

 MLD Service Vlan Config

  VLAN ID                                : 500
  VLAN NAME                              : VLAN500
  MLD Enabled [No]                       : Yes
  Querier Allowed [Yes]                  : Yes
  MLD Version                            : 2
  Strict Mode                            : No
  Last Member Query Interval (seconds)   : 10
  Robustness-Count                       : 2
  Port Type         Port Mode    Forced Fast Leave  Fast Leave  Fast Learn
  ---- ------------------- ----------------- ----------------------------

  1   100/1000T    forward      No                 Yes         No
  2   100/1000T    forward      No                 Yes         No
  3   100/1000T    forward      No                 Yes         No
  46  100/1000T    blocked      No                 Yes         No
  47  100/1000T    blocked      No                 Yes         No
  48  100/1000T    auto         No                 Yes         No

Switch#
```

- VLAN ID and Name
- MLD enabled: whether MLD is enabled on the VLAN (default NO, but the information for this VLAN will be listed only if MLD is enabled)
- Querier Allowed: whether the switch is allowed to act as querier on the VLAN
- MLD version
- Strict Mode: whether strict mode is enabled
- Last Member Query Interval: showing the amount of time the querier waits for a response from members, in seconds
- Query Interval showing the length of time between membership queries, in seconds
- Query Max. Response Time displaying the number of seconds to wait for a response to a query, in seconds
- Robustness-Count displaying the number of times to retry a query
- Port information for each IPv6 multicast group address in the VLAN (general group command) or for the specified IPv6 multicast group address (specific group command):
    - Group multicast address.
    - Last reporter: Last MLD host to send a join to the group address.
    - Group expiry: Time until the group expires if no further joins are seen.
    - Port name for each port.
    - Port type for each port: Ethernet connection type.
    - Port mode for each port:
        - `auto`

          (follows MLD snooping rules, that is, packets are forwarded for joined groups)
        - `forward`

          (all multicast packets are forwarded to this group)
        - `blocked`

          (all multicast packets are dropped, except that packets for well-known addresses are forwarded)
        - Expiry time for each port: Amount of time until this port is aged out of the multicast address group, unless a join is received.

- whether Forced Fast Leave is enabled or disabled
- whether Fast Leave is enabled or disabled
- whether Fast Learn is enabled or disabled - not in sw commands

## show ipv6 mld config

**Syntax**

`show ipv6 mld config`

**Description**

Displays current global MLD configuration for all MLD-enabled VLANS on the switch.

**Example output**

Configuring the current MLD

```
Switch# show ipv6 mld config

 MLD Service Config

  Control unknown multicast [Yes]   : Yes
  Forced fast leave timeout [4]     : 4
  Send Router Alert Option          : Default
  VLAN ID VLAN NAME  MLD Enabled Querier Allowed Version
  ------- ---------- ----------- -----------------------
  500     VLAN500    Yes         Yes             2

Switch#
```

# Commands to list currently joined ports

**show ipv6 mld vlan *vid* group**

Lists the ports currently joined for all IPv6 multicast group addresses in the specified VLAN.

**show ipv6 mld vlan *vid* group *ipv6-addr***

Lists the ports currently joined for the specified IPv6 multicast group address in the specified VLAN.

**show ipv6 mld vlan *vid* group *port-num***

Shows a list of all the MLD groups on the specified port.

**show ipv6 mld vlan *vid* group *ipv6-addr* source *ipv6-addr***

Only for MLDv2. Specify the source IPv6 address.

**Ports Joined to multicast groups in a specific VLAN**

The general form of the command is:

```
Switch# show ipv6 mld vlan 500 group

 MDL Service Protocol Group Info
 VLAN ID   : 500
 VLAN Name : VLAN500
 Group Address : ff1e::5
 Last Reporter : fe80::200:1ff:fe1b:c3a1
 Group Type    : Filtered

 Port Vers Mode Uptime   Expires  V1     Filter  Sources    Sources
```

```
                                  Timer  Timer   Forwarded  Blocked
 ---- -----------------------------------------------------------------
 2    2    EXC  32m 18s  2m 59s   -      2m 59s  0          0

 Group Address : ff1e::6
 Last Reporter : fe80::200:1ff:fe1b:c3a1
 Group Type    : Filtered

 Port Vers Mode Uptime   Expires  V1     Filter  Sources    Sources
                                  Timer  Timer   Forwarded  Blocked
 ---- ------------------------------------------------------------------
 2    2    EXC  32m 19s   2m 56s   -     2m 56s  0          0

 Port Vers Mode Uptime   Expires  V1     Filter  Sources    Sources
                                  Timer  Timer   Forwarded  Blocked
 ---- ------------------------------------------------------------------
 1    2    INC  0m 19s  4m 13s  -       -       1          0

 Group Address  : ff1e::7
 Source Address : 5555::1
 Source Type    : Filtered

 Port Mode Uptime   Expires  Configured Mode
 ---- ---------------------------------------
 1    INC  0m 19s   4m 13s   Auto

 Group Address : ff1e::8
 Last Reporter : fe80::200:1ff:fe1b:c3a1
 Group Type    : Filtered

 Port Vers Mode Uptime   Expires  V1     Filter  Sources    Sources
                                  Timer  Timer   Forwarded  Blocked
 ---- ------------------------------------------------------------------
 1    2    INC  0m 24s  4m 15s  -       -       1          0

Switch#
```

# show ipv6 mld statistics

### Syntax

`show ipv6 mld vlan *vid* statistics`

### Description

The general form of the command shows the total number of MLD-enabled VLANs and a count of multicast groups currently joined. Both forms of the command show VLAN IDs and names, as well as the number of filtered and standard multicast groups and the total number of multicast groups.

### Example output

This example shows MLD statistics for all VLANs configured.

```
Switch# show ipv6 mld 500 statistics

 MLD Statistics
  VLAN ID   : 500
  VLAN NAME : VLAN500
  Number of Filtered Groups             : 6
```

```
  Number of Standard groups             : 0
  Total Multicast Groups Joined         : 6


 Mode      EXCLUDE     INCLUDE
 ------- ------------ ------
 Filtered  6           1
 Standard  0           0
 Total     6           0

Switch#
```

# Counters

The following information is shown:

- VLAN number and name
- For each VLAN, number of:
    - general queries(MLDv1) received and sent
    - general queries (MLDv2) received and sent
    - version 1 group-specific queries received and sent
    - version 2 group-specific queries received and sent
    - group and source-specific queries received and sent
    - MLD version2 member reports (joins) received
    - MLD version 1 member reports (joins) received
    - version 1 leaves received and sent
    - packets forwarded to routers on this VLAN received and sent
    - packets forwarded to all ports on this VLAN received and sent
- Errors, number of:
    - MLD packets of unknown type received
    - malformed packets received
    - packets with bad checksums
    - packets from a martian source (the wrong subnet on an interface)
    - packets received on an MLD-disabled interface
    - queries—when a VLAN is configured as MLDv2 and an MLDv1 query is received from another switch for that VLAN, this counter is incremented. The reverse also applies.
- Port Counters, number of:
    - fast leaves that have occurred
    - forced fast leaves that have occurred
    - times a join has timed out on this VLAN

## show ipv6 mld vlan counters

**Syntax**

`show ipv6 mld vlan` *`vid`* `counters`

**Description**

Displays MLD counters for the specified VLAN.

**Example output**

MLD counters for a single VLAN

```
Switch# show ipv6 mld vlan 500 counters

 MLD Service Vlan Counters

  VLAN ID   : 500
  VLAN NAME : VLAN500


                                           Rx   Tx
                                           -------
 V1 All Hosts Query                        0    2
 V2 All Hosts Query                        0    24
 V1 Group Specific Query                   0    0
 V2 Group Specific Query                   0    12
 Group and Source Specific Query           0    0
 V2 Member Report                          220  0
 V1 Member Join                            12   0
 V1 Member Leave                           6    0
 Forward to Routers                        0    238
 Forward to VLAN                           0    26

 Errors:

 Unknown MLD Type                          0
 Unknown Packet                            0
 Malformed Packet                          12
 Bad Checksum                              0
 Martian Source                            0
 Packet received on MLD-disabled Interface 0
 Interface Wrong Version Query             0

 Port Counters:

 Fast Leave                                0
 Forced Fast Leave                         0
 Membership Timeout                        0

Switch#
```

The following information is shown:

- VLAN number and name
- For each VLAN:
  - number of general queries received
  - number of general queries sent
  - number of group-specific queries received
  - number of group-specific queries sent
  - number of MLD version 1 member reports (joins) received
  - number of MLD version 2 member reports (joins) received
  - number of leaves received
  - number of MLD packets of unknown type received
  - number of packets of unknown type received
  - number of malformed packets
  - number of packets received on MLD-disabled Interface

- number of packets forwarded to routers on this VLAN
- number of times a packet has been forwarded to all ports on this VLAN
- number of fast leaves that have occurred
- number of forced fast leaves that have occurred
- number of times a join has timed out on this VLAN

## mld reload

**Syntax**

`mld reload`

**Description**

This command resets the MLD state on all the interfaces.

**Example input**

```
Switch(vlan-500)# show ipv6 mld
```

MLD application is in Error State as System Resources are exhausted. Traffic will flood. Disable MLD on all VLANs or Issue the Command "mld reload" to take it out of Error. Refer to your product manual for information on MLD resource consumption.

```
Switch(vlan-500)#
```

## ipv6 mld send-router-alert

**Syntax**

`ipv6 mld send-router-alert default`

`ipv6 mld send-router-alert alternative-padding`

**Parameters**

**`send-router-alert`**

Enables/Disables insertion of the Router Alert option.

**`default`**

Enable insertion of the Router Alert option

**`alternative-padding`**

Enable insertion of the Router Alert option with padding in between the header length and the option type for interoperability.

# Listeners and joins

The "snooping" part of MLD snooping arises because a switch must track which ports have network nodes that are MLD hosts for any given multicast address. It does this by tracking "joins" on a per-port basis.

A network node establishes itself as an MLD host by issuing a multicast "join" request (also called a multicast "report") for a specific multicast address when it starts an application that listens to multicast traffic. The switch to which the node is connected sees the join request and forwards traffic for that multicast address to the node's port.

# Overview

An access control list (ACL) contains one or more access control entries (ACEs) specifying the criteria the switch uses to either permit (forward) or deny (drop) IP packets traversing the switch's interfaces. This chapter describes how to configure, apply, and edit static IPv6 ACLs for filtering IPv6 traffic in a network populated with the switches and how to monitor IPv6 ACL actions.

Because the switches operate in an IPv4/IPv6 dual stack mode, IPv6 and IPv4 ACLs can operate simultaneously in these switches. However:

- Static IPv6 ACLs and IPv4 ACLs do not filter each other's traffic.
- IPv6 and IPv4 ACEs cannot be configured in the same static ACL.
- RADIUS-assigned ACLs can be configured to filter either IPv4 traffic only, or both IPv4 and IPv6 traffic.

IPv6 traffic filtering with ACLs can help to improve network performance and restrict network use by creating policies for:

- Switch management accessPermits or denies in-band management access. This includes limiting and/or preventing the use of designated protocols that run on top of IPv6, such as TCP, UDP, ICMP, and others. Also included are the use of DSCP criteria and control for application transactions based on source and destination IPv6 addresses and transport layer port numbers.
- Application access securityEliminates unwanted IPv6 traffic in a path by filtering IPv6 packets where they enter or leave the switch on specific VLAN interfaces.

The ACLs described in this chapter can filter IPv6 traffic to or from a host, a group of contiguous hosts, or entire subnets.

CAUTION

The ACLs described in this chapter can enhance network security by blocking selected IPv6 traffic and can serve as part of your network security program. However, because ACLs do not provide user or device authentication or protection from malicious manipulation of data carried in IPv6 packet transmissions, they should not be relied upon for a complete security solution.

Static IPv6 ACLs on the switches do not screen non-IPv6 traffic such as IPv4, AppleTalk, and IPX packets.

## Types of IPv6 ACLs

A permit or deny policy for IPv6 traffic you want to filter is based on source and destination IPv6 address, plus other IPv6 protocol factors such as TCP/UDP, ICMP, and DSCP.

## Concurrent IPv4 and IPv6 ACLs

The switches support concurrent configuration and operation of IPv4 and IPv6 ACLs. For information on IPv4 ACLs, see the *ArubaOS-Switch Access Security Guide* for your switch.

## Multiple ACL assignments on an interface

The switch simultaneously supports IPv6, IPv4, and RADIUS-assigned ACLs on the same interface (subject to internal resource availability.) This means that traffic on a port belonging to a given VLAN "X" can simultaneously be subject to all of the ACLs listed in this table:

**Table 13:** *Per-interface multiple ACL assignments*

| ACL type | ACL application |
|---|---|
| RADIUS-assigned (dynamic) ACLs | One port-based ACL (for first client to authenticate on the port) or up to 32 user-based ACLs (one per authenticated client)<br>NOTE: If one or more user-based RADIUS-assigned ACLs are assigned to a port, the only traffic allowed inbound on the port is from authenticated clients. |
| IPv6 static ACLs | One static VACL for IPv6 traffic for VLAN "X" entering or leaving the switch through the port.One static port ACL for IPv6 traffic entering or leaving the switch on the port.One inbound and one outbound RACL filtering routed IPv6 traffic moving through the port for VLAN "X." (Also applies to inbound, switched traffic on VLAN "X" that has a destination on the switch itself.) |
| IPv4 static ACLs | One static VACL for IPv4 traffic for VLAN "X" entering or leaving the switch through the port. One static port ACL for any IPv4 traffic entering or leaving the switch on the port.One connection-rate ACL for inbound IPv4 traffic for VLAN "X" on the port (if the port is configured for connection-rate filtering).One inbound and one outbound RACL filtering routed IPv4 traffic moving through the port for VLAN "X". (Also applies to inbound, switched traffic on VLAN "X" that has a destination on the switch itself.) |

**More Information**

**Planning an ACL application** on page 93

## About filtering inbound traffic with multiple ACLS

When traffic inbound on a port is subject to multiple ACL assignments, and a RADIUS-assigned, user-based ACL is present, this traffic must satisfy the following conditions to be permitted on the switch:

| | |
|---|---|
| 1 | Originate with an authenticated client associated with the RADIUS-assigned ACL (if present). |
| 2 | Be permitted by the RADIUS-assigned ACL (if present). Includes both IPv4 and IPv6 traffic—unless the ACL is configured to exclude (drop) IPv6 traffic. |
| 3 | For IPv4-only traffic, be permitted by connection-rate ACL filtering. |
| 4 | Be permitted by a VACL configured on a VLAN to which the port is assigned.IPv4 VACLs and PACLs ignore IPv6 traffic, and the reverse. |
| 5 | Be permitted by a PACL assigned to the port. |
| 6 | For IPv4 traffic only, be permitted by a RACL assigned inbound to the port, if the traffic is subject to RACL rules. Be permitted by a RACL assigned inbound to the port, if the traffic is subject to RACL rules. |

## Filtering outbound traffic

Outbound IPv4 traffic can be filtered either by RACL or VACL/PACL assigned outbound on the port.

## Permitting traffic filtered through multiple ACLs

On a given interface where multiple ACLs apply to the same traffic, a packet having a match with a `deny` ACE in any applicable ACL on the interface (including an implicit `deny any any`) is dropped.

For example, suppose that the following is true:

- Ports 10 and 12 belong to VLAN 100.
- A static port ACL filtering inbound IPv6 traffic is configured on port 10.
- A VACL (with a different set of ACEs) is configured on VLAN 100.
- An RACL is also configured for inbound, routed traffic on VLAN 100.

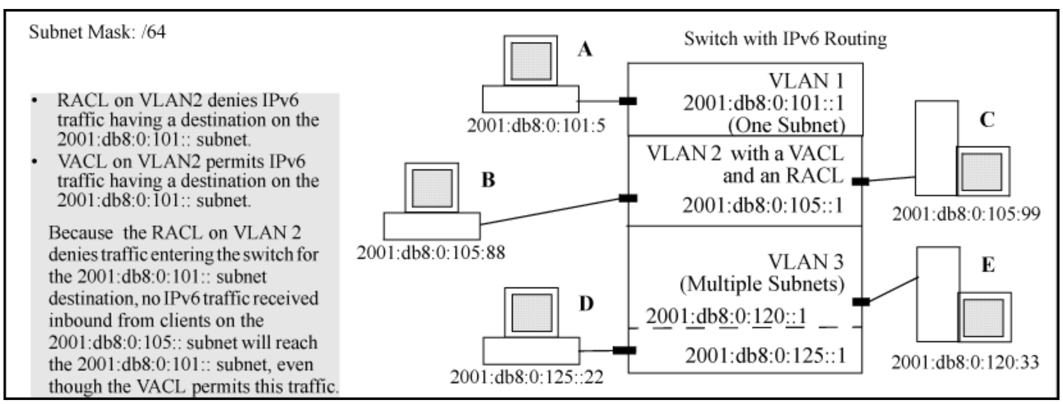
An inbound switched packet entering on port 10, with a destination on port 12, will be screened first by the VACL and then by the static port ACL and the RACL. A match with a `deny` action (including an implicit deny) in any of the applicable ACLs causes the switch to drop the packet. If the packet has a match with explicit `deny` ACEs in multiple ACLs and the log option is included in these ACEs, a log event for that denied packet occurs in each ACL where there is an applicable "deny" ACE. Logging can also be enabled for matches with "permit" ACEs.

However, in this case, suppose that VLAN 2 in the figure is configured with the following:

- A VACL permitting IPv6 traffic having a destination on the 2001:db8:0:101:: subnet
- An RACL that denies inbound IPv6 traffic having a destination on the 2001:db8:0:101:: subnet

In this case, no routed IPv6 traffic received on the switch from clients on the 2001:db8:0:105:: subnet will reach the 2001:db8:0:101:: subnet, even though the VACL allows such traffic. This is because the RACL is configured with a `deny` ACE that causes the switch to drop the traffic regardless of whether the VACL permits the traffic.

**Figure 3:** *Order of application for multiple ACLs on an interface*




Software release K.15.01 supports connection-rate ACLs for inbound IPv4 traffic, but not for IPv6 traffic.

In cases where an RACL and any type of port or VLAN ACL are filtering traffic entering the switch, the switched traffic explicitly permitted by the port or VLAN ACL is not filtered by the RACL (except when the traffic has a destination on the switch itself). However, routed traffic explicitly permitted by the port or VLAN ACL (and switched traffic having a destination on the switch itself) must also be explicitly permitted by the RACL, or it will be dropped.

A switched packet is not affected by an outbound RACL assigned to the VLAN on which the packet exits the switch.

Beginning with software release K.14.01, static ACL mirroring and static ACL rate-limiting are deprecated in favor of classifier-based mirroring and rate-limiting features that do not use ACLs. If ACL mirroring or ACL rate-limiting is already configured in a switch running software version K.13.xx, downloading and booting from release K.14.01 or greater automatically modifies the deprecated configuration to conform to the classifier-based mirroring and rate-limiting supported in release K.14.01 or greater. For more information on this topic, see the latest *Aruba-OS Switch Advanced Traffic Management Guide* for your switch.

## Features common to all ACL applications

- Any ACL can have multiple entries (ACEs).
- You can apply any one ACL to multiple interfaces.
- All ACEs in an ACL configured on the switch are automatically sequenced (numbered). For an existing ACL, entering an ACE without specifying a sequence number automatically places the ACE at the end of the list. Specifying a sequence number inserts the ACE into the list at the specified sequential location.
    - Automatic sequence numbering begins with "10" and increases in increments of 10. You can renumber the ACEs in an ACL and also change the sequence increment between ACEs.
    - The CLI `remark` command option allows you to enter a separate comment for each ACE.
- A source or destination IPv6 address and a prefix length, together, can define a single host, a range of hosts, or all hosts.
- Every ACL populated with one or more explicit ACEs automatically includes an implicit deny as the last entry in the list. The switch applies this action to packets that do not match other criteria in the ACL.
- In any ACL, you can apply an ACL log function to ACEs that have an explicit "deny" or “permit” action. (The logging occurs when there is a match on a "deny" or “permit” ACE that includes the `log` keyword.) The switch sends ACL logging output to syslog, if configured, and optionally, to a console session.

You can create ACLs for the switch configuration using either the CLI or a text editor. The text-editor method is recommended when you plan to create or modify an ACL that has more entries than you can easily enter or edit using the CLI alone.

## IPv6 ACLs

IPv6 ACLs enable filtering on the following:

- Source and destination IPv6 addresses (required), in one of the following options:
    - Specific host IPv6
    - Subnet or contiguous set of IPv6 addresses
    - Any IPv6 address
- Choice of any IPv6 protocol
- Optional packet-type criteria for ICMP traffic
- Optional source and/or destination TCP or UDP port, with a further option for comparison operators
- TCP flag (control bit) options
- Filtering for TCP traffic based on whether the subject traffic is initiating a connection ("established" option)
- Optional DSCP (IP precedence and ToS) criteria

The total number of ACLs each for IPv4 and IPv6 is determined from the number of unique identifiers in the configuration. For example, configuring two IPv6 ACLs results in an ACL total of two, even if neither is assigned to an interface. If you then assign a nonexistent IPv6 ACL to an interface, the new total is three, because the switch now has three unique IPv6 ACL names in its configuration.

For information on determining the current resource availability and usage, see the *Aruba-OS Switch Management and Configuration Guide* for your switch.

For ACL resource limits, see the latest *Aruba-OS Switch Management and Configuration Guide* for your switch.

### Create, enter and configure an ACL

For a match to occur with an ACE, a packet must have the source and destination IPv6 address criteria specified by the ACE, as well as any IPv6 protocol-specific criteria included in the command.

**Procedure**

1. Create and/or enter the context of a given ACL.
2. Enter the first ACE in a new ACL, or append an ACE to the end of an ACL.

**More Information**

**ipv6 access-list ascii-str** on page 104

## IPv6 ACL applications

ACL filtering is applied to IPv6 traffic as follows:

- Routed ACL (RACL) on a VLAN configured with an RACL, filters:
  - Routed IPv6 traffic entering or leaving the switch. (Routing can be between different VLANs or between different subnets in the same VLAN. IPv6 routing must be enabled.)
  - Routed IPv6 traffic having a DA on the switch itself. In **Figure 4: RACL filter applications on routed IPv6 traffic** on page 86, any of the IPv6 addresses shown in VLANs "A", "B", and "C". (IPv6 routing need not be enabled.)
  - If the ACL is applied to outbound traffic, filters outbound traffic generated by the switch itself
- VLAN ACL (VACL): On a VLAN configured with a VACL, filters inbound or outbound IPv6 traffic, regardless of whether it is switched or routed. On a multinetted VLAN, this includes inbound IPv6 traffic from any subnet.
- Static port ACL: Filters outbound IPv6 traffic on the port.
- RADIUS-assigned ACL: On a port having an ACL assigned by a RADIUS server to filter an authenticated client's traffic, filters inbound IPv4 and IPv6 traffic (or IPv4-only traffic) from that client

### RACL applications

RACLs filter routed IPv6 traffic entering or leaving the switch on VLANs configured with the "in" and/or "out" ACL option:

**Syntax**

```
vlan vid ipv6 access-group identifier [in|out|vlan-in|vlan-out]
```

```
interface tunnel tunnel-id ipv6 access-group identifier [in|out]
```

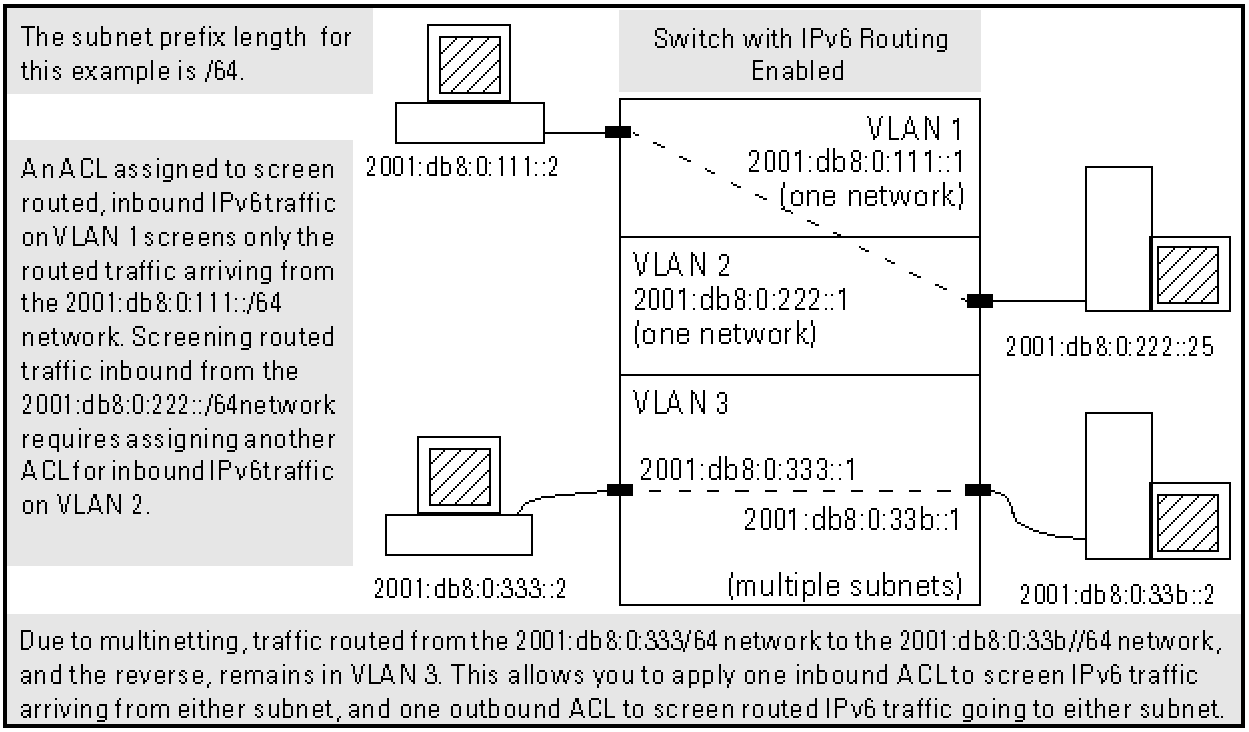
**RACL filter applications on routed IPv6 Traffic**

In the following figure:

- You would assign either an inbound ACL on VLAN 1 or an outbound ACL on VLAN 2 to filter a packet routed between subnets on different VLANs, that is, a packet sent from the workstation 2001:db8:0:111::2 on VLAN 1 to the server at 2001:db8:0:222::25 on VLAN 2. (An outbound ACL on VLAN 1 or an inbound ACL on VLAN 2 would not filter the packet.)

- If the traffic source and destination IP addresses are on devices external to the switch where multiple subnets are configured on the same VLAN, you can use either inbound or outbound ACLs to filter routed IPv6 traffic between the subnets on the VLAN

**Figure 4:** *RACL filter applications on routed IPv6 traffic*




### More information

The switch allows one inbound IPv6 RACL assignment and one outbound IPv6 RACL assignment configured per IP routing interface. This is in addition to any other IPv6 ACL assigned to the IP routing interface or to any ports on the VLAN. You can use the same RACL or different RACLs to filter inbound and outbound routed IPv6 traffic on an IP routing interface.

IPv6 RACLs do not filter traffic that remains in the same subnet from source to destination (switched traffic) unless the destination address (DA) or source address (SA) is on the switch itself.

## VACL applications

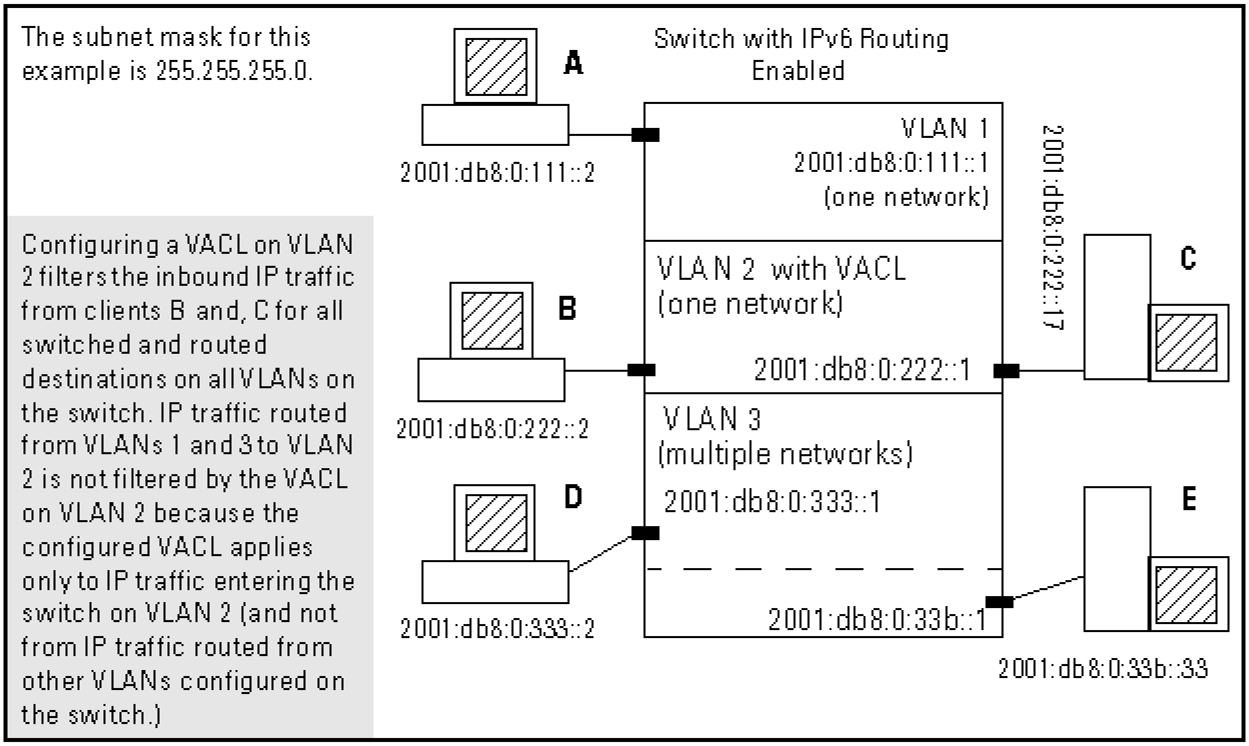
IPv6 VACLs filter traffic entering or leaving the switch on a VLAN configured with the "VLAN" ACL option:

### Syntax

```
vlan vid ipv6 access-group vacl-identifier <vlan-in|vlan-out>
```

**VACL filter applications on IPv6 traffic**
In the following figure ,you would assign a VACL to VLAN 2 to filter all inbound switched or routed IPv6 traffic received from clients on the 2001:db8 :0:222:: network. In this instance, routed IPv6 traffic received on VLAN 2 from VLANs 1 or 3 would not be filtered by the VACL on VLAN 2.

**Figure 5:** *Example of VACL filter applications on IPv6 traffic entering the switch*




The switch allows one IPv6 VACL assignment configured per VLAN. This is in addition to any other IPv6 ACL applications assigned to the IP routing interface or to ports in the VLAN.

## IPv6 static port ACL applications

An IPv6 static port ACL filters IPv6 traffic inbound or outbound on the designated ports, regardless of whether the traffic is switched or routed.

An IPv6 static port ACL filters IPv6 traffic inbound or outbound on the designated ports.

## RADIUS-assigned (dynamic) port ACL applications

Dynamic (RADIUS-assigned) port ACLs are configured on RADIUS servers and can be configured to filter IPv4 and IPv6 traffic inbound from clients authenticated by such servers. For example, in **Figure 5: Example of VACL filter applications on IPv6 traffic entering the switch** on page 87, client "A" connects to a given port and is authenticated by a RADIUS server. Because the server is configured to assign a dynamic ACL to the port, the IPv4 and IPv6 traffic inbound on the port from client "A" is filtered. See also **Operating notes for IPv6 applications** on page 88.

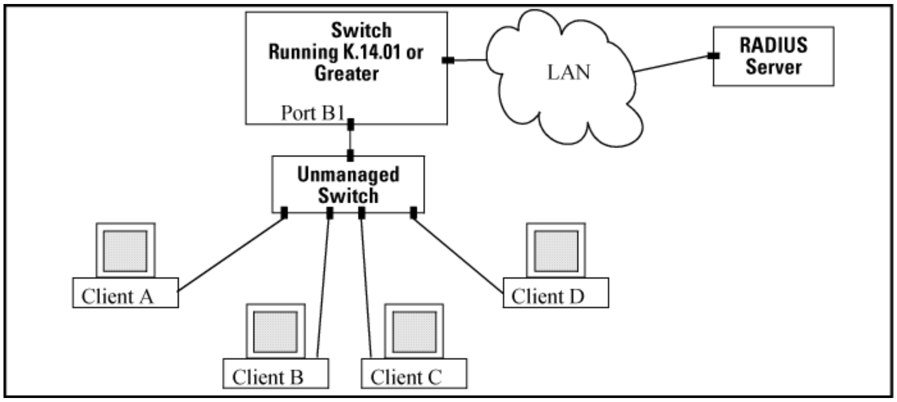
NOTE

Beginning with software release K.14.01, IPv6 support is available for RADIUS-assigned port ACLs configured to filter inbound IPv4 and IPv6 traffic from an authenticated client. Also, the implicit deny in RADIUS-assigned ACLs applies to both IPv4 and IPv6 traffic inbound from the client. For information on enabling RADIUS-assigned ACLs, see chapter "Configuring RADIUS Support for Switch Services" in this guide.

## Effect of RADIUS-assigned ACLs when multiple clients are using the same port

Some network configurations may allow multiple clients to authenticate through a single port where a RADIUS server assigns a separate, RADIUS-assigned ACL in response to each client's authentication on that port. In such cases, a given client's inbound traffic is allowed only if the RADIUS authentication response for that client includes a RADIUS-assigned ACL. Clients authenticating without receiving a RADIUS-assigned ACL are immediately de-authenticated. For example, in the following figure, clients A through D authenticate through the same port (B1) on an E switch running software release K.14.01 or greater.

**Figure 6:** *Example of Multiple Clients Authenticating Through a Single Port*




In this case, the RADIUS server must be configured to assign an ACL to port B1 for any of the authorized clients authenticating on the port.

## 802.1X user-based and port-based applications

User-Based 802.1X access control allows up to 32 individually authenticated clients on a given port. Port-Based access control does not set a client limit and requires only one authenticated client to open a given port (and is recommended for applications where only one client at a time can connect to the port).

- If you configure 802.1X user-based security on a port and the RADIUS response includes a RADIUS-assigned ACL for at least one authenticated client, the RADIUS response for all other clients authenticated on the ports must also include a RADIUS-assigned ACL. Inbound IP traffic on the port from a client that authenticates without receiving a RADIUS-assigned ACL is dropped and the client de-authenticated.
- Using 802.1X port-based security on a port where the RADIUS response to a client authenticating includes a RADIUS-assigned ACL, different results can occur, depending on whether any additional clients attempt to use the port and whether these other clients initiate an authentication attempt. This option is recommended for applications where only one client at a time can connect to the port, and not recommended for instances where multiple clients may access the same port at the same time. For more information, see the latest *Aruba-OS Switch Access Security Guide* for your switch.

**More Information**

**Operating notes for IPv6 applications** on page 88

## Operating notes for IPv6 applications

- For RADIUS ACL applications using software release K.14.01 or greater, the switch operates in a dual-stack mode, and a RADIUS-assigned ACL filters both IPv4 and IPv6 traffic. At a minimum, a RADIUS-assigned ACL automatically includes the implicit deny for both IPv4 and IPv6 traffic. Thus, an ACL configured on a RADIUS server to filter IPv4 traffic also denies inbound IPv6 traffic from an authenticated client unless the ACL includes ACEs that permit the desired IPv6 traffic. The reverse is true for a dynamic ACL configured on RADIUS server to filter IPv6 traffic. (ACLs are based on the MAC address of the authenticating client.) See the latest *Aruba-OS Switch Access Security Guide* for your switch.
- To support authentication of IPv6 clients:

- The VLAN to which the port belongs must be configured with an IPv6 address.
- Connection to an IPv6-capable RADIUS server must be supported.

- For 802.1X or MAC authentication methods, clients can authenticate regardless of their IP version (IPv4 or IPv6).
- For the web-based authentication method, clients must authenticate using IPv4. However, this does not prevent the client from using a dual stack, or the port receiving a RADIUS-assigned ACL configured with ACEs to filter IPv6 traffic.
- The RADIUS server must support IPv4 and have an IPv4 address. RADIUS clients can be dual stack, IPv6-only, or IPv4-only.
- 802.1X rules for client access apply to both IPv6 and IPv4 clients for RADIUS-assigned ACLs.

**More Information**

**802.1X user-based and port-based applications** on page 88

## General ACL operations

- ACLs do not provide DNS host name support. ACLs cannot be configured to screen host name IP traffic between the switch and a DNS.
- ACLs do not affect serial port access. ACLs do not apply to the switch's serial port.
- ACL screening of IPv6 traffic generated by the switch. Outbound IPv6 RACL applications on a switch do not screen IPv6 traffic (such as broadcasts, Telnet, Ping, and ICMP replies) generated by the switch itself. All ACLs applied on the switch do screen this type of traffic when other devices generate it. Similarly, all ACL applications can screen responses from other devices to unscreened IPv6 traffic the switch generates.
- ACL logging
    - The ACL logging feature generates a message only when packets are explicitly denied as the result of a match, and not when explicitly permitted or implicitly denied. To help test ACL logging, configure the last entry in an ACL as an explicit `deny` statement with a `log` statement included and apply the ACL to an appropriate port or VLAN.
    - A detailed event will be logged for the first packet that matches a "deny" or "permit" ACL logged entries with the appropriate action specified.
    - Subsequent packets matching ACL logged entries will generate a new event that summarizes the number of packets that matched each specific entry (with the time period).
    - Logging enables you to selectively test specific devices or groups. However, excessive logging can affect switch performance. For this reason, E recommends that you remove the logging option from ACEs for which you do not have a present need.
    - Also, avoid configuring logging where it does not serve an immediate purpose. (ACL logging is not designed to function as an accounting method.)
- When configuring logging, you can reduce excessive resource use by configuring the appropriate ACEs to match with specific hosts instead of entire subnets. For more information on resource usage, see page **Deleting an ACL in the Running Configuration** on page 90.
- Minimum number of ACEs in an IPv6 ACL. An IPv6 ACL must include at least one ACE to enable traffic screening. An IPv6 ACL can be created "empty", that is, without any ACEs. However, if an empty ACL is applied to an interface, the Implicit Deny function does not operate, and the ACL has no effect on traffic.
- Monitoring shared resources. Applied ACLs share internal switch resources with several other features. However, if the internal resources become fully subscribed, additional ACLs cannot be applied until the necessary resources are released from other applications. For information on determining current resource availability and usage, see the latest *Aruba-OS Switch Management and Configuration Guide* for your switch.
- Protocol support. ACL criteria does not include use of MAC address information or QoS.
- Replacing or adding to an active IPv6 ACL policy. If you assign an IPv6 ACL to an interface and subsequently add or replace ACEs in that ACL, each new ACE becomes active when you enter it. If the ACL is configured on multiple interfaces when the change occurs, the switch resources must accommodate all applications of the ACL. If there are insufficient resources to accommodate one of several ACL applications affected by the

change, the change is not applied to any of the interfaces and the previous version of the ACL remains in effect.
- "Strict" IPv6 TCP and UDP.When the IPv6 ACL configuration includes TCP or UDP options, the switch operates in "strict" TCP and UDP mode for increased control. In this case, the switch compares all IPv6 TCP and UDP packets against the IPv6 ACLs.

### Deleting an ACL in the Running Configuration

Attempting to delete an ACL that is assigned to an interface removes all configured ACEs from the ACL, but leaves an "empty" ACL in the configuration. To delete an ACL that is assigned to an interface, do the following:

**Procedure**

1. Use the `no ipv6 access-group` command in the **interface** context to remove the ACL from the interface.
2. Use the `no ipv6 access-list` *`name-str`* command. to delete the ACL.

## Procedures for planning and configuring ACLs

**Procedure**

1. Identify the ACL action to apply.
2. Determine the best points at which to apply specific ACL controls. For example, you can improve network performance by filtering unwanted IPv6 traffic at the edge of the network instead of in the core. Also, on the switch itself, you can improve performance by filtering unwanted IPv6 traffic where it is inbound to the switch instead of outbound.

| Traffic source | ACL application |
|---|---|
| IPv6 traffic from a specific, authenticated client | RADIUS-assigned ACL for inbound IPv6 traffic from an authenticated client on a portFor more information, see chapter "Configuring RADIUS Server Support for Switch Services" in the latest version of the *Access Security Guide* for your switch. See also the documentation for your RADIUS server. |
| IPv6 traffic entering or leaving the switch on a specific port | Static port ACL (static-port assigned) for inbound or outbound IPv6 traffic on a port from any source |
| Switched or routed IPv6 traffic entering or leaving the switch on a specific VLAN | VACL (VLAN ACL) |
| Routed IPv6 traffic entering or leaving the switch on a specific VLAN | RACL (routed ACL) |

3. Identify the IPv6 traffic types to filter:
   a. The SA and/or the DA of IPv6 traffic you want to permit or deny; this can be a single host, a group of hosts, a subnet, or all hosts.
   b. IPv6 traffic of a specific protocol type (0 to 255).
   c. TCP traffic (only) for a specific TCP port or range of ports, including optional control of connection traffic based on whether the initial request should be allowed.
   d. UDP traffic (only) or UDP traffic for a specific UDP port.
   e. ICMP traffic (only) or ICMP traffic of a specific type and code.
   f. Any of the above with specific DSCP settings.

4. Design the ACLs for the control points (interfaces) you have selected. Where you are using explicit "deny" or “permit” ACEs, you can optionally use the ACL logging feature for notification that the switch is denying unwanted packets, or permitting packets that you want to go through.
5. Configure the ACLs on the selected switches.
6. Assign the ACLs to the interfaces you want to filter, using the ACL application (static port ACL or VACL) appropriate for each assignment.
7. If you are using a routed ACL (RACL), ensure that IPv6 routing is enabled on the switch.
8. Test for desired results.

# IPv6 ACL operation

An ACL is a list of one or more ACEs, where each ACE consists of a matching criteria and an action (permit or deny). An ACL applies only to the switch in which it is configured. ACLs operate on assigned interfaces, and offer the below traffic filtering options:

- IPv6 traffic inbound or outbound on a port.
- IPv6 traffic inbound or outbound on a VLAN.
- Routed IPv6 traffic entering or leaving the switch on a VLAN. (ACLs do not screen traffic at the internal point where traffic moves between VLANs or subnets within the switch.)

The following table lists the range of interface options:

| Interface | ACL application | Application point | Filter action |
|---|---|---|---|
| Port | Static port ACL (switch configured) | Inbound on the switch port | Inbound IPv6 traffic |
| | RADIUS-assigned ACLThis chapter describes ACLs statically configured on the switch. For information on RADIUS-assigned ACLs, see the *Aruba-OS Switch Access Security Guide* for your switch. | Inbound on the switch port used by authenticated client | Inbound IPv6 traffic from the authenticated client |
| VLAN | VACL | Entering the switch on the VLAN | Inbound IPv6 traffic |
| IP routing interface (VLAN or tunnel) | RACLSupports one inbound and/or one outbound RACL. When both are used, one RACL can be assigned to filter both inbound and outbound, or different RACLs can be assigned to filter inbound and outbound. | Entering the switch on the VLAN | Routed IPv6 traffic entering the switch and IPv6 traffic with a destination on the switch itself |
| | | Exiting from the switch on the VLAN | Routed IPv6 traffic exiting from the switch |

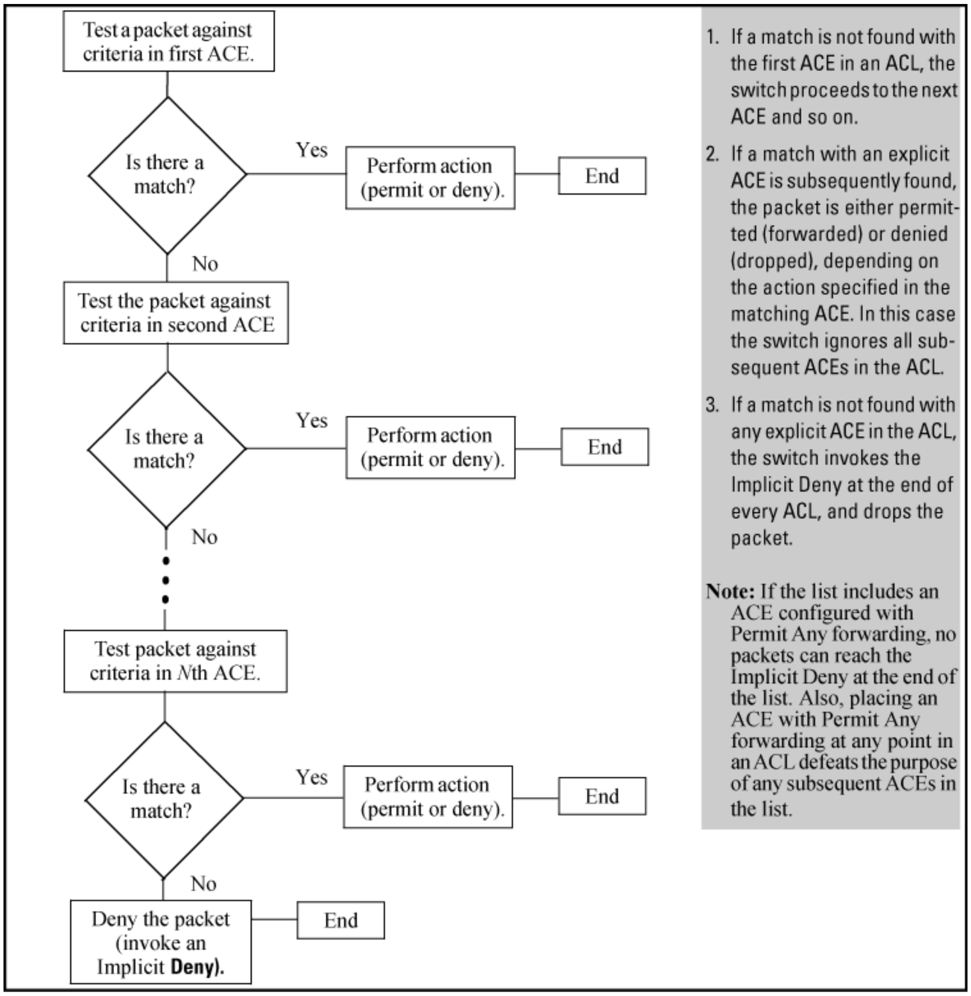
NOTE

After you assign an ACL to an interface, the default action on the interface is to implicitly deny any IPv6 traffic that is not permitted by the ACL. (This applies only in the direction of traffic flow filtered by the ACL.)

## Packet filtering process

When an ACL filters a packet, it sequentially compares each ACE's filtering criteria to the corresponding data in the packet until it finds a match. The action indicated by the matching ACE (deny or permit) is then performed on the packet.

**Figure 7:** *Packet-filtering process in an ACL with N entries (ACEs)*




NOTE

The order in which an ACE occurs in an ACL is significant. For example, if an ACL contains six ACEs, but the first ACE allows "Permit Any" forwarding, the ACL permits all IPv6 traffic, and the remaining ACEs in the list do not apply, even if they have a match with any traffic permitted by the first ACE.

## Packet-filtering

Suppose that you want to configure an ACL (with an ID of "Test-02") to invoke these policies for IPv6 traffic entering the switch on VLAN 100:

The following ACL, when assigned to filter inbound traffic on VLAN 100, supports the above case:

**How an ACL filters packets**

```
ipv6 access-list "Test-02"
10 permit ipv6 2001:db8:0:fb::11:42/128 ::/0
20 deny tcp 2001:db8:0:fb::11:101/128 eq 23 ::/0
```

```
30 permit ipv6 2001:db8:0:fb::11:101/128 ::/0
40 permit tcp 2001:db8:0:fb::11:33/128 ::/0 eq 23
Implicit Deny Any Any
```

**Line 10**

Permits IPv6 traffic from 2001:db8:0:fb::11:42. Packets matching this criterion are permitted and will not be compared to any later ACE in the list. Packets not matching this criterion will be compared to the next entry in the list.

**Line 20**

Denies IPv6 Telnet traffic from 2001:db8:0:fb::11:101. Packets matching this criterion are dropped and are not compared to later criteria in the list. Packets not matching this criterion are compared to the next entry in the list.

**Line 30**

Permits IPv6 traffic from 2001:db8:0:fb::11:101. Packets matching this criterion will be permitted and will not be compared to any later criteria in the list. Because this entry comes after the entry blocking Telnet traffic from this same address, there will not be any Telnet packets to compare with this entry; they have already been dropped as a result of matching the preceding entry.

**Line 40**

Permits IPv6 Telnet traffic from 2001:db8:0:fb::11:33. Packets matching this criterion are permitted and are not compared to any later criteria in the list. Packets not matching this criterion are compared to the next entry in the list.

**Implicit Deny Any Any**

This entry does not appear in an actual ACL, but is implicit as the last entry in every IPv6 ACL. Any IPv6 packets that do not match any of the criteria in the preceding ACL entries will be denied (dropped) from the VLAN.

NOTE

It is important to remember that ACLs configurable on the switch include an implicit `deny ipv6 any any`. That is, IPv6 packets that the ACL does not explicitly permit or deny will be implicitly denied, and therefore dropped instead of forwarded on the interface. If you want to pre-empt the implicit deny so that packets not explicitly denied by other ACEs in the ACL will be permitted, insert an explicit `permit ipv6 any any` as the last ACE in the ACL. Doing so permits any packet not explicitly denied by earlier entries (this solution would not apply in the preceding example, where the intention is for the switch to forward only the explicitly permitted packets entering the switch on VLAN 100.)

# Planning an ACL application

Before creating and implementing ACLs, define the policies you want your ACLs to enforce and understand how the ACL assignments will impact your network users.

NOTE

IPv6 traffic entering the switch on a given interface is filtered by the ACLs configured for inbound traffic on that interface. For this reason, an inbound packet is denied (dropped) if it has a match with an implicit (or explicit) `deny ipv6 any any` in any of the inbound ACLs applied to the interface.

**More Information**

**Multiple ACL assignments on an interface** on page 81

## IPv6 traffic management and improved network performance

You can use ACLs to block IPv6 traffic from individual hosts, workgroups, or subnets, and to block access to VLANs, subnets, devices, and services. Traffic criteria for ACLs include:

- Switched IPv6 traffic
- Switched and/or routed IPv6 traffic
- IPv6 traffic of a specific protocol type (0 to 255)
- TCP traffic (only) for a specific TCP port or range of ports, including optional control of connection traffic based on whether the initial request should be allowed
- UDP traffic (only) or UDP traffic for a specific UDP port
- ICMP traffic (only) or ICMP traffic of a specific type and code
- Any of the above with specific precedence and/or ToS settings

Depending on the source, destination, or both of a given IPv6 traffic type, you must also determine the ACL applications (VACL or static port ACL) to filter the traffic on the applicable switch interfaces. Depending on the source and/or destination of a given IPv6 traffic type, you must also determine the ACL applications (RACL, VACL, or static port ACL) to filter the traffic on the applicable switch interfaces. Answering the following questions can help you to design and properly position ACLs for optimum network usage:

- What are the logical points for minimizing unwanted IPv6 traffic, and what ACL applications should be used?In many cases, it makes sense to prevent unwanted IPv6 traffic from reaching the core of your network by configuring ACLs to drop unwanted IPv6 traffic at or close to the edge of the network. (The earlier in the network path you can deny unwanted traffic, the greater the benefit for network performance.)
- From where is the traffic coming?The source and destination of IPv6 traffic you want to filter determines the ACL application to use (VACL, static port ACL, and RADIUS-assigned ACL). The source and destination of IPv6 traffic you want to filter determines the ACL application to use (RACL, VACL, static port ACL, and RADIUS-assigned ACL).
- What IPv6 traffic should you explicitly deny?Depending on your network size and the access requirements of individual hosts, it can involve creating many ACEs in a given ACL (or many ACLs), which increases the complexity of your solution.
- What IPv6 traffic can you implicitly deny by taking advantage of the implicit `deny ipv6 any any` to deny IPv6 traffic that you have not explicitly permitted?It can reduce the number of entries needed in an ACL.
- What IPv6 traffic should you permit?In some cases, you should identify permitted IPv6 traffic. In others, depending on your policies, you can insert an ACE with "permit any" forwarding at the end of an ACL. This means that IPv6 traffic not matched by earlier entries in the list will be permitted.

## Security

ACLs can enhance security by blocking IPv6 traffic carrying an unauthorized source IPv6 address. This can include:

- Blocking access from specific devices or interfaces (port or VLAN)
- Blocking access to or from subnets in your network
- Blocking access to or from the Internet
- Blocking access to sensitive data storage or restricted equipment
- Preventing specific TCP, UDP, and ICMP traffic types, including unauthorized access using functions such as Telnet and SSH

You can also enhance switch management security by using ACLs to block IPv6 traffic that has the switch itself as the DA.

CAUTION

ACLs can enhance network security by denying selected IPv6 traffic, and they can serve as one aspect of maintaining network security. However, because ACLs do not provide user or device authentication, or protection from malicious manipulation of data carried in IPv6 packet transmissions, they should not be relied upon for a complete security solution.

NOTE

ACLs in the switches do not filter non-IPv6 traffic such as IPv4, AppleTalk, and IPX packets.

## Guidelines for planning the structure of an ACL

After determining the ACL application (VACL or static port ACL) to use at a particular point in your network, determine the order in which to apply individual ACEs to filter IPv6 traffic. After determining the ACL application (RACL, VACL, or static port ACL) to use at a particular point in your network, determine the order in which to apply individual ACEs to filter IPv6 traffic. For information on ACL applications, see **IPv6 ACL applications** on page 85.

**The sequence of ACEs is significant.**

When the switch uses an ACL to determine whether to permit or deny a packet, it compares the packet to the criteria specified in the individual ACEs in the ACL, beginning with the first ACE in the list and proceeding sequentially until a match is found. When a match is found, the switch applies the indicated action (permit or deny) to the packet.

**The first match in an ACL dictates the action on a packet.**

Subsequent matches in the same ACL are ignored. However, if a packet is permitted by one ACL assigned to an interface, but denied by another ACL assigned to the same interface, the packet will be denied on the interface.

**On any ACL, the switch implicitly denies IPv6 packets that are not explicitly permitted or denied by the ACEs configured in the ACL.**

If you want the switch to forward a packet for which there is not a match in an ACL, append an ACE that enables permit any forwarding as the last ACE in an ACL. This ensures that no packets reach the implicit deny case for that ACL.

**ACEs listing order**

Generally, you should list ACEs from the most specific (individual hosts) to the most general (subnets or groups of subnets), unless doing so permits IPv6 traffic that you want dropped.

For example, an ACE allowing a series of workstations to use a specialized printer should occur earlier in an ACL than an entry used to block widespread access to the same printer.

## ACL configuration considerations

**RACLs and routed IPv6 traffic**

Except for IPv6 traffic with a DA on the switch itself, RACLs filter only routed IPv6 traffic that is entering or leaving the switch on a given VLAN. Thus, if routing is not enabled on the switch, there is no routed IPv6 traffic for RACLs to filter.

RACLs screen routed IPv6 traffic entering or leaving the switch on a given VLAN interface is subject to ACL filtering. This means that the following traffic:

- IPv6 traffic arriving on the switch through one VLAN and leaving the switch through another VLAN.
- IPv6 traffic arriving on the switch through one subnet and leaving the switch through another subnet within the same, multinetted VLAN.Filtering the desired, routed IPv6 traffic requires assigning an RACL to screen IPv6 traffic inbound or outbound on the appropriate VLANs. If a multinetted VLAN, it means that IPv6 traffic inbound from different subnets in the same VLAN is screened by the same inbound RACL, and IPv6 traffic outbound from different subnets is screened by the same outbound RACL. See **Figure 4: RACL filter applications on routed IPv6 traffic** on page 86.

**RACLs do not filter switched IPv6 traffic unless the switch itself is the SA or DA**

RACLs do not filter IPv6 traffic moving between ports belonging to the same VLAN or subnet (if a subnetted VLAN). (IPv6 traffic moving between ports in different subnets of the same VLAN can be filtered by a RACL.)

NOTE

RACLs do filter routed or switched IPv6 traffic having an SA or DA on the switch itself.

**VACLs and switched or routed IPv6 traffic**

A VACL filters IPv6 traffic entering or leaving the switch on the VLANs to which it is assigned. These filter IPv6 traffic entering or leaving the switch through any port belonging to the designated VLAN. You can assign an ACL to any VLAN that is statically configured on the switch. ACLs do not operate with dynamic VLANs. A VACL assigned to a VLAN applies to all physical ports on the switch belonging to that VLAN, including ports that have dynamically joined the VLAN.

**Per switch ACL limits for all ACL types**

At a minimum, an ACL must have one, explicit "permit" or "deny" ACE. You can configure up to 2048 ACLs (IPv4 and IPv6 combined). Total ACEs in all ACLs depend on the combined resource usage by ACL and other features.

**Implicit deny**

In any static ACL, the switch implicitly (automatically) applies an implicit deny `ipv6 any any` that does not appear in `show` listings. This means that the ACL denies any packet it encounters that does not have a match with an entry in the ACL. Thus, if you want an ACL to permit any IPv6 packets that you have not expressly denied, you must enter a `permit ipv6 any any` as the last ACE in an ACL. Because, for a given packet, the switch sequentially applies the ACEs in an ACL until it finds a match. Any packet that reaches a `permit ipv6 any any` entry is permitted and does not encounter the implicit "Deny" ACE the switch automatically includes at the end of the ACL.

**Explicitly permitting IPv6 traffic**

Entering a `permit ipv6 any any` ACE in an ACL permits the IPv6 traffic not previously permitted or denied by that ACL. Any ACEs listed after that point are redundant.

**Explicitly denying IPv6 traffic**

Entering a `deny ipv6 any any` ACE in an ACL denies IPv6 traffic not previously permitted or denied by that ACL. Any ACEs listed after that point have no effect.

**Replacing one ACL with another of the same type**

For a specific interface, the most recent ACL assignment using a given application replaces any previous ACL assignment using the same application on the same interface. For example, if you assign a VACL named "Test-01" to filter inbound IPv6 traffic on VLAN 20, but later you assign another VACL named "Test-02" to filter inbound IPv6 traffic on this same VLAN, VACL "Test-02" replaces VACL "Test-01" as the ACL to use. For example, if you assign an RACL named "Test-01" to filter inbound routed IPv6 traffic on VLAN 20, but later you assign another RACL named "Test-02" to filter inbound routed IPv6 traffic on this same VLAN, RACL "Test-02" replaces RACL "Test-01" as the ACL to use.

**Static port ACLs**

These are applied per port, per port list, or per static trunk. Adding a port to a trunk applies the ACL trunk configuration to the new member. If a port is configured with an ACL, the ACL must be removed before the port is added to the trunk. In addition, removing a port from an ACL-configured trunk removes the ACL configuration from that port.

## How an ACE uses a prefix to screen packets for SA and DA matches

For an IPv6 ACL, a match with a packet occurs when both the protocol and the SA/DA configured in a given ACE within the ACL are a match with the same criteria in a packet being filtered by the ACL.

In IPv6 ACEs, prefixes define how many leading bits in the SA and DA to use for determining a match. That is, the switch uses IPv6 prefixes in CIDR format to specify how many leading bits in a packet's SA and DA must be an exact match with the same bits in an ACE. The bits to the right of the prefix are "wildcards" and are not used to determine a match.

| Prefix | Range of applicable addresses | Examples |
|---|---|---|
| /0 | Any IPv6 host | ::/0 |
| / 1-/127 | All IPv6 hosts within the range defined by the number of bits in the prefix | 2001:db8::/482001:db8::/64 |
| /128 | One IPv6 host | 2001:db8::218:71ff:fec4:2f00/128 |

**SA/DA prefix lengths**

The following ACE applies to Telnet packets from a source address where the leading bits are set to 2001:db8:10:1 and any destination address where the leading bits are set to 2001:db8:10:1:218:71ff:fec.

```
permit tcp 2001:db8:10:1::/64 eq 23 2001:db8:10:1:218:71ff:fec4::/112
```

**::/64**

Prefix Defining the Mask for the Leading Bits in the Source Address

**::/112**

Prefix Defining the Mask for the Leading Bits in the Destination Address

Thus, in the above example, if an IPv6 Telnet packet has an SA match with the ACE's leftmost 64 bits and a DA match with the ACE's leftmost 112 bits, there is a match and the packet is permitted. In this case, the source and destination addresses allowed are:

| Address | Prefix | Range of unicast addresses |
|---|---|---|
| Source (SA) | 2001:db8:10:1 | *prefix*<br>::0 to*prefix* :FFFF:FFFF:FFFF:FFFF |
| Destination (DA) | 2001:db8:10:1:218:71ff:fec4 | *prefix*<br>:0to*prefix* :FFFF |

To summarize, when the switch compares an IPv6 packet to an ACE in an ACL, it uses the subnet prefixes configured with the SA and DA in the ACE to determine how many leftmost, contiguous bits in the ACE's SA and DA must be matched by the same bits in the SA and DA carried by the packet. Thus, the subnet prefixes specified with the SA and DA in an ACE determine the ranges of source and destination addresses acceptable for a match between the ACE and a packet being filtered.

**More Information**

**[any|hostSA|SA/prefix-length]** on page 105

## Prefix usage differences between ACLs and other IPv6 addressing

For ACLs, the prefix is used to specify the leftmost bits in an address that are meaningful for a packet match. In other IPv6 usage, the prefix separates network and subnet values from the device identifier in an address.

| Prefix usage | Examples | Notes |
| --- | --- | --- |
| For an SA or DA in the ACE belonging to an IPv6 ACL, the associated prefix specifies how many consecutive, leading bits in the address are used to define a match with the corresponding bits in the SA or DA of a packet being filtered. | 2620:0:a03:e102:215:60ff:fe7a:adc0/128 2620:0:a03:e102:215/80::/0 | All bits. Used for a specific SA or DA. The first 80 bits. Used for an SA or DA having 2620:0:a03:e102:215 in the leftmost 80 bits of an address.Zero bits. Used to allow a match with "any" SA or DA. |
| For the IPv6 address assigned to a given device, the prefix defines the type of address and the network and subnet in which the address resides. In this case, the bits to the right of the prefix comprise the device identifier. | fe80::215:60ff:fe7a:adc0/642620:0:a03:e102:215:60ff:fe7a:adc0/64 | Link-Local address with a prefix of 64 bits and a device ID of 64 bits.Global unicast address with a prefix of 64 bits and a device ID of 64 bits. |
| For an RA, the included prefix defines the network or range of networks and the subnets the router is advertising. | 2620:0:a03::/482620:0:a03:e102::/64 | An RA with a 48-bit prefixan RA with a 64-bit prefix |

# Configuring and assigning an IPv6 ACL

**More Information**

**Permit/deny options** on page 98

## Implementing IPv6 ACLs

**Procedure**

1. Configure one or more ACLs.
2. This creates and stores the ACLs in the switch configuration.
3. Assign an ACL to an interface using one of the following applications:
   a. RACL (routed IPv6 traffic entering or leaving the switch on a given VLAN)
   b. VACL (IPv6 traffic entering or leaving the switch on a given VLAN)
   c. Static port ACL (IPv6 traffic entering or leaving the switch on a given port, port list, or static trunk)
4. If the ACL is applied as an RACL, IPv6 routing must be enabled. Except for instances where the switch is the traffic source or destination, assigned RACLs filter IPv6 traffic only when IPv6 routing is enabled on the switch.

## Permit/deny options

You can use the following criteria as options for permitting or denying a packet:

- Source IPv6 address
- Destination IPv6 address
- IPv6 protocol options:
  - All IPv6 traffic
  - IPv6 traffic of a specific protocol type (0 to 255)

- IPv6 traffic for a specific TCP port or range of ports, including:
  - Optional control of connection (established) traffic based on whether the initial request should be allowed
  - TCP flag (control bit) options
- IPv6 traffic for a specific UDP port or range of ports
- IPv6 traffic for a specific ICMP type and code
- Any of the above with specific DSCP precedence or ToS settings

**More Information**

**Configuring and assigning an IPv6 ACL** on page 98

## Implicit deny override

If a packet does not have a match with the criteria in any of the ACEs in the ACL, the ACL denies (drops) the packet. If you have to override the implicit deny so that a packet that does not have a match will be permitted, configure `permit ipv6 any any` as the last ACE in the ACL. This directs the ACL to permit (forward) packets that do not have a match with any earlier ACE listed in the ACL and prevents these packets from being filtered by the implicit `deny ipv6 any any`.

**Overriding an implicit deny**

Suppose the following ACL with five ACEs is assigned to filter the IPv6 traffic from an authenticated client on a given port in the switch:

```
10 permit ipv6 ::/0 fe80::136:24/128
20 permit ipv6 ::/0 fe80::156:7/128
30 deny ipv6 ::/0 fe80::156:3/128
40 deny tcp ::/0 ::/0 eq 23
50 permit ipv6 ::/0 ::/0
(deny ipv6 ::/0 ::/0)
```

For an inbound packet with a destination IP address of FE80::156:3, the ACL:

1. Compares the packet to the first ACE first (line 10).
2. Since there is not a match with the first ACE, the ACL compares the packet to the second ACE, where there is also not a match (line 20).
3. The ACL compares the packet to the third ACE. There is an exact match, so the ACL denies (drops) the packet (line 30).
4. The packet is not compared to the fourth ACE (line 40).
5. The last line demonstrates the "deny any any" ACE implicit in every IPv6 ACL. Inbound IPv6 traffic from an authenticated client that does not have a match with any of the five explicit ACEs in this ACL will be denied by the implicit "deny any any".

As shown above, the ACL tries to apply the first ACE in the list. If there is not a match, it tries the second ACE, and so on. When a match is found, the ACL invokes the configured action for that entry (permit or drop the packet) and no further comparisons of the packet are made with the remaining ACEs in the list. This means that when an ACE whose criteria matches a packet is found, the action configured for that ACE is invoked, and any remaining ACEs in the ACL are ignored. Because of this sequential processing, successfully implementing an ACL depends in part on configuring ACEs in the correct order for the overall policy you want the ACL to enforce.

## ACL configuration

After you enter an ACL command, you may want to inspect the resulting configuration. This is especially true where you are entering multiple ACEs into an ACL. Also, it is helpful to understand the configuration structure when using later sections in this chapter. The basic ACL structure includes four elements:

1. ACL identity This is a string of up to 64 characters specifying the ACL name.
2. Optional `remark` entries.

3. One or more deny/permit list entries (ACEs): One entry per line.

| Element | Notes |
|---|---|
| Identifier | Alphanumeric; up to 64 characters, including spaces |
| Remark | Allows up to 100 alphanumeric characters, including blank spaces. (If any spaces are used, the remark must be enclosed in a pair of single or double quotes.).A remark is associated with a particular ACE and has the same sequence number as the ACE. (One remark is allowed per ACE.) See **Remarks** on page 119. |
| Maximum ACEs per switch | The maximum number of ACEs supported by the switch is up to 3072 for IPv6 ACEs and up to 3072 for IPv4 ACEs. The maximum number of ACEs applied to a VLAN or port depends on the concurrent resource usage by multiple configured features. For more information, use the `show qos|access-list resources` command. |

4. Implicit deny

## ACL Configuration Structure

Individual ACEs in an IPv6 ACL include:

- Optional remark statements
- A permit/deny statement
- Source and destination IPv6 addressing
- Choice of IPv6 criteria
- Optional ACL `log` command (for `deny` or `permit` entries)

**General structure options for an IPv6 ACL**

```
ipv6 access-list identifier

  [ seq-# ]

  [ remark remark-str ]

   permit | deny
   0 - 255
   esp
   ah
   sctp
   icmp
       SA  [operator  value ]
       DA  [operator  value ]
               [type [code] | icmp-msg ]
               [dscp  codepoint | precedence ]
               ipv6
               tcp
        SA  [operator  value ]
        DA  [operator  value ]
               [dscp codepoint | precedence]
               [established]
               [ack | fin | rst | syn]
               udp
         SA  [operator  value ]
         DA  [operator  value ]
      [log] (Allowed only with "deny" or "permit" ACEs.)
         Implicit Deny Any Any
   exit
```

**Displayed ACL configuration**

```
Switch# show run
.
.
.
ipv6 access-list "Sample-List-1"
   10 permit ipv6 2001:db8:0:130::55/128 2001:db8:0:130::240/128
   20 permit tcp ::/0 ::/0 eq 23
   30 remark "ALLOWS HTTP FROM SINGLE HOST."
   30 permit tcp 2001:db8:0:140::14/128 eq 80 ::/0 eq 3871
   40 remark "DENIES HTTP FROM ANY TO ANY."
   40 deny tcp ::/0 ::/0 eq 80 log
   50 deny udp 2001:db8:0:150::44/128 eq 69 2001:db8:0:120::19/128 range 3680 3690
log
   60 deny udp ::/0 2001:db8:0:150::121/128 log
   70 permit ipv6 2001:db8:0:01::/56 ::/0
   exit
```

| Line | Action |
|---|---|
| 10 | Permits all IPv6 traffic from the host at 2001:db8:0:130::55 to the host at 2001:db8:0:130::240. |
| 20 | Permits all Telnet traffic from any source to any destination. |
| 30 | Includes a remark and permits TCP port 80 traffic received at any destination as port 3871 traffic. |
| 40 | Includes a remark and denies TCP port 80 traffic received at any destination, and causes a log message to be generated when a match occurs. |
| 50 | Denies UDP port 69 (TFTP) traffic sent from the host at 2001:db8:0:150::44 to the host at 2001:db8:0:120::19 with a destination port number in the range of 3680 to 3690 and causes a log message to be generated when a match occurs. |
| 60 | Denies UDP traffic from any source to the host at 2001:db8:0:150::121 and causesa log message to be generated when a match occurs. |
| 70 | Permits all IPv6 traffic with an SA prefix of 2001:db8:0:01/56 that is not already permitted or denied by the preceding ACEs in the ACL. |

NOTE

An implicit `deny IPv6 any any` is automatically applied following the last line (70, in this case) and denies all IPv6 traffic not already permitted or denied by the ACEs in lines 10 through 70.

# ACL configuration considerations

## The sequence of entries in an ACL is significant

When the switch uses an ACL to determine whether to permit or deny a packet, it compares the packet to the criteria specified in the individual ACEs in the ACL, beginning with the first ACE in the list and proceeding sequentially until a match is found. When a match is found, the switch applies the indicated action (permit or deny) to the packet. This is significant because, once a match is found for a packet, subsequent ACEs in the same ACL are not applied to that packet, regardless of whether they match the packet.

**ACE that permits all IPv6 traffic not implicitly denied**

Suppose that you have applied the ACL shown, to inbound IPv6 traffic on VLAN 1 (the default VLAN):

```
ipv6 access-list "Sample-List-2"
10 deny ipv6 2001:db8::235:10/128 ::/0
20 deny ipv6 2001:db8::245:89/128 ::/0
30 permit tcp 2001:db8::18:100/128 2001:db8::237:1/128
40 deny tcp 2001:db8::18:100/128 ::/0
50 permit ipv6 ::/0 ::/0
(Implicit deny ipv6 any any)
exit
```

After the last explicit ACE, there is always an Implicit Deny. However, in this case it will not be used because the last permit ipv6 ACL allows all IPv6 packets that earlier ACEs have not already permitted or denied.

| Line # | Action |
|---|---|
| N/A | Shows IP type (IPv6) and ID (Sample-List-2). |
| 10 | A packet from source address 2001:db8:235:10 will be denied (dropped). This ACE filters out all packets received from 2001:db8:235:10. As a result, IPv6 traffic from that device will not be allowed, and packets from that device will not be compared against any later entries in the list. |
| 20 | A packet from IPv6 source address 2001:db8::245:89 will be denied (dropped). This ACE filters out all packets received from 2001:db8::245:89. As the result, IPv6 traffic from that device will not be allowed, and packets from that device will not be compared against any later entries in the list. |
| 30 | A TCP packet from SA 2001:db8::18:100 with a DA of 2001:db8::237:1 will be permitted (forwarded). Since no earlier ACEs in the list have filtered TCP packets from 2001:db8::18:100 with a destination of 2001:db8::237:1, the switch will use this ACE to evaluate such packets. Any packets that meet this criteria will be forwarded. (Any packets that do not meet this TCP source-destination criteria are not affected by this ACE.) |
| 40 | A TCP packet from source address 2001:db8::18:100 to any destination address will be denied (dropped). Since, in this example, the intent is to block TCP traffic from 2001:db8::18:100 to any destination except the destination stated in the ACE at line 30, this ACE must follow the ACE at line 30. (If their relative positions were exchanged, all TCP traffic from 2001:db8::18:100 would be dropped, including the traffic for the 2001:db8::237:1 destination.) |
| 50 | Any packet from any IPv6 source address to any IPv6 destination address will be permitted (forwarded). The only traffic filtered by this ACE will be packets not permitted or denied by the earlier ACEs. |

*Table Continued*

| Line # | Action |
|---|---|
| 60 | The implicit deny (`deny ipv6 any any`) is a function the switch automatically adds as the last action in all IPv6 ACLs. It denies (drops) traffic from any source to any destination that has not found a match with earlier entries in the ACL. In this example, the ACE at line 50 permits (forwards) any traffic not already permitted or denied by the earlier entries in the list, so there is no traffic remaining for action by the implicit deny function. |
| exit | Defines the end of the ACL. |

### Implied deny function

In any ACL having one or more ACEs, there is always a packet match. This is because the switch automatically applies the implicit deny as the last ACE in any ACL. This function is not visible in ACL listings, but is always present; see **ACE that permits all IPv6 traffic not implicitly denied** on page 102. This means that if you configure the switch to use an ACL for filtering either inbound or outbound traffic on a VLAN, any IPv6 packets not permitted or denied by the explicit entries you create is denied by the implicit deny action. If you want to pre-empt the implicit deny (so that IPv6 traffic not addressed by earlier ACEs in a given ACL is permitted), insert an explicit `permit ipv6 any any` as the last explicit ACE in the ACL.

### Assignment of an ACL to an interface

The switch stores ACLs in the configuration file. Until you actually assign an ACL to an interface, it is present in the configuration, but not used (and does not use any of the monitored resources.) See the latest version of the *Aruba-OS Switch Management and Configuration Guide* for your switch

### Assignment of an ACL name to an interface

In this case, if you subsequently create an ACL with that name, the switch automatically applies each ACE as soon as you enter it in the running-config file. Similarly, if you modify an existing ACE in an ACL you already applied to an interface, the switch automatically implements the new ACE as soon as you enter it. For example, if you configure two ACLs, but assign only one of them to a VLAN, the ACL total is two, for the two unique ACL names. If you then assign the name of an empty ACL to a VLAN, the new ACL total is three, because the switch now has three unique ACL names in its configuration. (RADIUS-based ACL resources are drawn from the IPv4 allocation.)

## Creating an ACL using the CLI

You can use either the switch CLI or an offline text editor to create an ACL. This section describes the CLI method, which is recommended for creating short ACLs.

## General ACE rules

These rules apply to all ACEs you create or edit using the CLI.

### Adding or inserting an ACE in an ACL

To add an ACE to the end of an ACL:

**Procedure**

1. Use the `ipv6 access-list` *`name-str`* command to enter the context for a specific IPv6 ACL. (If the ACL does not exist in the switch configuration, this command creates it.)
2. Enter the text of the ACE without specifying a sequence number.
3. For example, the following pairs of commands enter the context of an ACL named "List-1" and add a "permit" ACE to the end of the list. This new ACE permits the IPv6 traffic from the device at 2001:db8:0:a9:8d:100 to go to all destinations.

```
Switch(config)# ipv6 access-list List-1
Switch(config-ipv6-acl)# permit host 2001:db8:0:a9::8d:100 any
```

To insert an ACE anywhere in an existing ACL:

Enter the context of the ACL and specify a sequence number.

**To insert a new ACE**
To insert a new ACE as line 15 between lines 10 and 20 in an existing ACL named "List-2" to deny traffic from the device at 2001:db8:0:a9::8d:77:

```
Switch(config)# ipv6 access-list List-2
Switch(config-ipv6-acl)# deny host 2001:db8:0:a9::8d:77 any
```

## Deleting an ACE

**Procedure**

1. Enter the **ACL** context and delete the sequence number for the unwanted ACE.
2. To view the sequence numbers of the ACEs in a list, use `show access-list` *`acl-name-str`* `config` .

**To delete an ACE**
To delete the ACE at line 40 in an ACL named "List-2", enter the following commands:

```
Switch(config)# ipv6 access-list List-2 config
Switch(config-ipv6-acl)# no 40
```

### Duplicate ACE sequence numbers

Duplicate sequence numbering for ACEs are not allowed in the same ACL. Attempting to enter a duplicate ACE displays the `Duplicate sequence number` message.

# ipv6 access-list ascii-str

This command is a prerequisite for entering or editing ACEs in an ACL.

**Syntax**

`ipv6 access-list` *`ascii-str`*

**Description**

Places the CLI in the IPv6 ACL (**ipv6-acl**) context specified by the *ascii-str* alphanumeric identifier. This enables entries of individual ACEs in the specified ACL. If the ACL does not exist, this command creates it.

**Options**

`ascii-str`

Specifies an alphanumeric identifier for the ACL and consists of an alphanumeric string of up to 64 case-sensitive characters. If you include spaces in the string, you must enclose the string in single or double quotes. For example: "Accounting ACL".

**Example input**

Entering the ACL context

```
Switch(config)# ip access-list Sample-List
Switch(config-ipv6-acl)#
```

**More Information**

**Create, enter and configure an ACL** on page 84

# Commands to configure ACEs in an ACL

Configuring ACEs is done after using the `ipv6 access-list` *`ascii-str`* command described on page **ipv6 access-list ascii-str** on page 104 to enter the IPv6 ACL (**ipv6_acl**) context of an ACL.

## [deny|permit]ipv6

**Syntax**

`[deny|permit][`*`ipv6|ipv6 protocol|ipv6-protocol-nbr`*`]`

**Description**

To insert a new ACE between two existing ACEs in an ACL, precede `deny` or `permit` with an appropriate sequence number.

**Options**

**`[deny|permit]`**

These keywords are used in the IPv6 ACL (**ipv6-acl**) context to specify whether the ACE denies or permits a packet matching the criteria in the ACE, as described below.

**`[ipv6|ipv6-protocol|ipv6 protocol-nbr]`**

Used after `deny` or `permit` to specify the packet protocol type required for a match. An ACL must include one of the following:

**`ipv6`**

Any IPv6 packet.

**`ipv6-protocol`**

Any one of the following IPv6 protocol names:

`esp ah sctp icmp`[1] `tcp`[2] `udp`[3]

***`ipv6-protocol-nbr`***

The protocol number of an IPv6 packet type, such as "8" for Exterior Gateway Protocol or 121 for Simple Message Protocol. Range: 0 to 255

[1,2,3] For ICMP, TCP, and UDP, additional (optional) criteria can be specified, as described in **Filtering ICMP traffic**.

## [any|hostSA|SA/prefix-length]

**Syntax**

`[any|host`*`SA`*`|`*`SA/prefix-length`*`]`

**Description**

This is the first instance of IPv6 addressing in an ACE. It follows the protocol specifier and defines the source IPv6 address (SA) a packet must carry for a match with the ACE.

**Options**

**`any`**

Allows IPv6 packets from any IPv6 SA.

**`host` *`SA`***

Specifies only packets having a single address as the SA. Use this criterion when you want to match only the IPv6 packets from a single SA.

**`SA` *`prefix-length`***

Specifies packets received from one or more contiguous subnets or contiguous addresses within a single subnet. The prefix length is in CIDR format and defines the number of leftmost bits to use in determining a match.

**Example**

Prefix-length applications:

- 2001:db8:0:e102::10:100/120 matches any IPv6 address in the range of 2001:db8:0:e102::10:<0100 - 01FF>
- 2001:db8:a0:e102::/64 matches any IPv6 address having a prefix of 2001:db8:a0:e102.
- FE80::/16 matches any link-local address on an interface.

**More Information**

**How an ACE uses a prefix to screen packets for SA and DA matches** on page 96

## [any|host DA|DA/prefix-length]

**Syntax**

`[any|host DA|DA/prefix-length]`

**Description**

This is the second instance of addressing in an IPv6 ACE. It follows the first (SA) instance, described earlier in this section, and defines the destination IPv6 address (DA) that a packet must carry to have a match with the ACE.

**Options**

**`any`**

Allows IPv6 packets to any IPv6 DA.

**`host` *`DA`***

Specifies only packets having *`DA`* as the destination address. Use this criterion when you want to match only the IPv6 packets for a single DA.

***`DA/prefix-length`***

Specifies packets intended for one or more contiguous subnets or contiguous addresses within a single subnet. The prefix length is in CIDR format and defines the number of leftmost bits to use in determining a match.

**More Information**

**CIDR notation usage to enter the IPv6 ACL prefix length** on page 137

## [ dscp codepoint/precedence ]

**Syntax**

`[ dscp codepoint/precedence ]`

**Description**

This option follows the DA to include a DSCP codepoint or precedence as a matching criteria.

**Options**

***`codepoint`***

Supports these codepoint selection options:

***`0 - 63`***

Select a specific DSCP codepoint by entering its decimal equivalent.

Assured Forwarding (AF) codepoint matches:

| AF | DSCPMatch |
|---|---|
| af11 | 001010 |
| af12 | 001100 |
| af13 | 001110 |
| af21 | 010010 |
| af22 | 010100 |
| af23 | 010110 |
| af31 | 011010 |
| af32 | 011100 |
| af33 | 011110 |
| af41 | 100010 |
| af42 | 100100 |
| af43 | 100100 |

**Default**

Matches with the **000000** (default) DSCP.

**ef**

Expedited forwarding (EF) (**000000**) DSCP match.

**precedence**

Supports selection of a precedence setting in the DSCP.

| Option | Precedence Bits | Name |
|---|---|---|
| cs1 | 001 | priority |
| cs2 | 010 | immediate |

*Table Continued*

| Option | Precedence Bits | Name |
|---|---|---|
| cs3 | 011 | flash |
| cs4 | 100 | flash-override |
| cs5 | 101 | critical |
| cs6 | 110 | Internet (for internetwork control) |
| cs7 | 111 | network (for network control) |

`[log]`

This option can be used after the DA to generate an Event Log message if:

- The action is `deny` or `permit`.
- There is a match.
- ACL logging is enabled.

For a given ACE, if `log` is used, it must be the last keyword entered.

**Table 14:** *DSCP codepoints with decimal equivalents*

| DSCP bits | Decimal | DSCP bits | Decimal | DSCP bits | Decimal |
|---|---|---|---|---|---|
| 000000 | 0 (default) | 010110 | 22 | 101011 | 43 |
| 000001 | 1 | 010111 | 23 | 101100 | 44 |
| 000010 | 2 | 011000 | 24 | 101101 | 45 |
| 000011 | 3 | 011001 | 25 | 101110 | 46 |
| 000100 | 4 | 011010 | 26 | 101111 | 47 |
| 000101 | 5 | 011011 | 27 | 110000 | 48 |
| 000110 | 6 | 011100 | 28 | 110001 | 49 |
| 000111 | 7 | 011101 | 29 | 110010 | 50 |
| 001000 | 8 | 011110 | 30 | 110011 | 51 |
| 001001 | 9 | 011111 | 31 | 110100 | 52 |
| 001010 | 10 | 100000 | 32 | 110101 | 53 |
| 001011 | 11 | 100001 | 33 | 110110 | 54 |
| 001100 | 12 | 100010 | 34 | 110111 | 55 |

*Table Continued*

| DSCP bits | Decimal | DSCP bits | Decimal | DSCP bits | Decimal |
|---|---|---|---|---|---|
| 001101 | 13 | 100011 | 35 | 111000 | 56 |
| 001110 | 14 | 100100 | 36 | 111001 | 57 |
| 001111 | 15 | 100101 | 37 | 111010 | 58 |
| 010000 | 16 | 100110 | 38 | 111011 | 59 |
| 010001 | 17 | 100111 | 39 | 111100 | 60 |
| 010010 | 18 | 101000 | 40 | 111101 | 61 |
| 010011 | 19 | 101001 | 41 | 111110 | 62 |
| 010100 | 20 | 101010 | 41 | 111111 | 63 |

# ACL Bypass

Enable or disable hardware configuration to block packets with nonclassifiable layer 4 packets.

## ipv6 access-list deny-non-classifiable-layer4-header

**Syntax**

```
ipv6 access-list deny-non-classifiable-layer4-header
```

**Description**

Configure the switch to drop all IPv6 packets for which the layer 4 header cannot be processed by ACL hardware.

## show running-config

**Syntax**

```
show running-config
```

**Description**

Enabled and disabled configuration settings can be viewed using this command.

**Example output**

This example shows enabled and disabled configuration settings.

```
Running configuration:
; Stack Configuration Editor; Created on release #KA.15.16.0000x
; Ver #05:19.ff.ff.3f.ef:cc


 ipv6 access-list deny-non-classifiable-layer4-header
.
```

## show statistics aclv6

**Syntax**

```
show statistics aclv6 <deny-non-classifiable-layer4-header/ACL-NAME-STR>
```

**Description**

Shows the number of packets dropped due to a nonclassifiable Layer 4 header.

**Options**

**ACL-NAME-STR**

The ACL to show statistics for.

**deny-non-classifiable-layer4-header**

Shows the number of packets dropped due to a nonclassifiable Layer 4 header.

**Example output**

This example shows the number of packets dropped.

```
Switch(config)# show statistics aclv6 deny-non-classifiable-layer4-header
Hit Counts for deny-non-classifiable-layer4-header
Total
(0) 5 deny-non-classifiable-layer4-header
```

## show access list

**Syntax**

```
show access-list
```

**Description**

This command shows the enabled and disabled access control lists.

**Example output**

This example shows Access Control Lists.

```
Access Control Lists
 deny-fragmented-tcp-header          : Enabled/Disabled
 deny-non-classifiable-layer4-header : Enabled/Disables

 Type  Appl  Name
 ----  ----  ----
 ext   no    100
 ipv6  no    300
```

### show access-list config

**Syntax**

```
show access-list config
```

**Description**

To view details of the access control lists.

**Example output**

```
deny-fragmented-tcp-header
deny-non-classifiable-layer4-header
ip access-list extended "100"
5 deny tcp 0.0.0.0 255.255.255.255 20.0.20.1 0.0.0.255 eq 23
10 deny tcp 0.0.0.0 255.255.255.255 20.0.20.1 0.0.0.255 eq 23
20 permit tcp 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
30 deny icmp 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255 log
exit
```

```
ipv6 access-list "300"
10 deny icmp ::/0 2001::1/64
50 deny icmp ::/0 2001:2::50/64
60 permit icmp ::/0 ::/0
exit
```

## clear statistics

### Syntax

```
clear statistics aclv6 deny-non-classifiable-layer4-header
```

## show access-list resources

### Syntax

```
show access-list resources
```

### Description

Displays rules, meters, and ports used by different ACL and classifier features. This rule takes up six ACL entries.

### Example output

```
Resource usage in Policy Enforcement Engine
               |   Rules    |  Rules Used
         Ports | Available | ACL | QoS | IDM |  VT | Mirr | PBR |  OF  | Other |
 --------------+-----------+-----+-----+-----+-----+------+-----+------+-------|
 1/1-24,49-50  |      8166 |   6 |   0 |   0 |   0 |    0 |   0 |    0 |     0 |
 1/25-48,51-52 |      8176 |   0 |   0 |   0 |   0 |    0 |   0 |    0 |     0 |

               |   Meters   |  Meters Used
         Ports | Available | ACL | QoS | IDM |  VT | Mirr | PBR |  OF  | Other |
 --------------+-----------+-----+-----+-----+-----+------+-----+------+-------|
 1/1-24,49-50  |      2047 |     |   0 |   0 |     |      |     |    0 |     0 |
 1/25-48,51-52 |      2047 |     |   0 |   0 |     |      |     |    0 |     0 |
               |Application|
               |Port Ranges|  Application Port Ranges Used
         Ports | Available | ACL | QoS | IDM |  VT | Mirr | PBR |  OF  | Other |
 --------------+-----------+-----+-----+-----+-----+------+-----+------+-------|
 1/1-24,49-50  |        60 |   0 |   0 |   0 |     |    0 |   0 |      |     0 |
 1/25-48,51-52 |        60 |   0 |   0 |   0 |     |    0 |   0 |      |     0 |

               |    PBR     |
               | Next-hops |  PBR Next-hops Used
         Ports | Available | ACL | QoS | IDM |  VT | Mirr | PBR |  OF  | Other |
 --------------+-----------+-----+-----+-----+-----+------+-----+------+-------|
 1/1-24,49-50  |      1024 |     |     |     |     |      |   0 |      |   0   |
 1/25-48,51-52 |      1024 |     |     |     |     |      |   0 |      |   0   |
 2 of 32 Policy Engine management resources used.

Key:
ACL = Access Control Lists
QoS = Device & Application Port Priority, QoS Policies, ICMP rate limits
IDM = Identity Driven Management
VT  = Virus Throttling blocks
Mirr = Mirror Policies, Remote Intelligent Mirror endpoints
PBR = Policy Based Routing Policies
OF = OpenFlow
Other = Management VLAN, DHCP Snooping, ARP Protection, Jumbo IP-MTU, RA Guard,
Control Plane
Protection, Service Tunnel.

Resource usage includes resources actually in use, or reserved for future use by
```

```
the listed feature.
Internal dedicated-purpose resources, such as port bandwidth limits or VLAN QoS
priority,
are not included.
```

# TCP and UDP traffic configuration in IPv6 ACLs

An ACE designed to permit or deny TCP or UDP traffic can optionally include port number criteria for either the source, the destination, or both. Use of TCP criteria also allows the `established` option for controlling TCP connection traffic.

## Commands to configure TCP

`[deny|permit] tcp`

*`SA [ comparison-operator tcp-src-port ]`*

*`DA [ comparison-operator tcp-dest-port ]`*

`[ established ]`

`[ ack ][ fin ][ rst ][ syn ]`

## Commands to configure UDP

`[deny|permit] udp`

*`SA [ comparison-operator udp-src-port ]`*

*`DA [ comparison-operator udp-dest-port ]`*

# comparison-operator tcp/udp-src-port

**Syntax**

`[ comparison-operator tcp/udp-src-port ]`

**Description**

In an IPv6 ACL using either `tcp` or `udp` as the IP packet protocol type, you can optionally apply comparison operators specifying TCP or UDP source and/or destination port numbers or ranges of numbers to define the criteria for a match.

**Options**

**`eq`** ***`tcp/udp-port-nbr`***

"Equal To" — to have a match with the ACE entry, the TCP or UDP source port number in a packet must be equal to *`tcp/udp-port-nbr`* .

**`gt`** ***`tcp/udp-port-nbr`***

"Greater Than" — to have a match with the ACE entry, the TCP or UDP source port number in a packet must be greater than *`tcp/udp-port-nbr`* .

**`lt`** ***`tcp/udp-port-nbr`***

"Less Than" — to have a match with the ACE entry, the TCP or UDP source port number in a packet must be less than *`tcp/udp-port-nbr`* .

**`neq`** ***`tcp/udp-port-nbr`***

"Not Equal" — to have a match with the ACE entry, the TCP or UDP source port number in a packet must not be equal to *`tcp/udp-port-nbr`* .

**range** ***start-port-nbr end-port-nbr***

For a match with the ACE entry, the TCP or UDP source-port number in a packet must be in the range *start-port-nbr end-port-nbr* .

**Usage**

To specify a TCP or UDP source port number in an ACE:

1. Select a comparison operator from the following list.
2. Enter the port number or a well-known port name.

## Port number or well-known port name

Use the TCP or UDP port number required by your application. The switch also accepts these well-known TCP or UDP port names as an alternative to their port numbers:

- TCPbgp, dns, ftp, http, imap4, ldap, nntp, pop2, pop3, smtp, ssl, telnet
- UDPbootpc, bootps, dns, ntp, radius, radius-old, rip, snmp, snmp-trap, tftp

To list the above names, press the **[Shift]+ [?]** key combination after entering an operator.

## Comparison operators and well-known port names

This option applies only where TCP is the configured IPv6 protocol type. It blocks the synchronizing packet associated with establishing a new TCP connection, while allowing all other IPv6 traffic for existing connections.

For example, a Telnet connect requires TCP traffic to move both ways between a host and the target device. Simply applying a deny to inbound Telnet traffic on a VLAN prevents Telnet sessions in either direction, because responses to outbound requests are blocked. However, by using the established option, inbound Telnet traffic arriving in response to outbound Telnet requests are permitted, but inbound Telnet traffic trying to establish a new connection is denied.

The `established` and `dscp` options are mutually exclusive in a given ACE.

Configuring `established` and any combination of TCP control bits in the same ACE is supported, but `established` must precede any TCP control bits configured in the ACE.

**TCP control bits**

In a given ACE for filtering TCP traffic you can configure one or more of these options:

**[ ack ]**

Acknowledgement

**[ fin ]**

Sender finished

**[ rst]**

Connection reset

**[ syn]**

TCP control bit: sequence number synchronize

# Filtering ICMP traffic

This option allows configuring an ACE to selectively permit some types of ICMP traffic, while denying other types. An ACE designed to permit or deny ICMP traffic can optionally include an ICMP type and code value to permit or deny an individual type of ICMP packet, while not addressing other ICMP traffic types in the same ACE. As a further option, the ACE can include the name of an ICMP packet type.

## Commands to filter ICMP traffic

**Syntax**

`[deny|permit] icmp` *`SA DA icmp-type icmp-code`*

**Options**

*`icmp-type icmp-code`*

This option identifies an individual ICMP packet type as criteria for permitting or denying that type of ICMP traffic in an ACE.

- icmp-type— This value is in the range of 0 to 255 and corresponds to an ICMP packet type.
- icmp-code— This value corresponds to an ICMP code for an ICMP packet type. It is optional and needed only when a particular ICMP subtype is needed as a filtering criterion.Range: 0 to 255

**Example**

Showing two ACEs entered in an ACL context:

```
#permit icmp any any 1 3
#permit icmp any any destination-unreachable
```

**Options**

*`icmp-type name`*

These name options are an alternative to the [ icmp-type [ icmp-code ] ] methodology described above.

| | |
|---|---|
| cert-path-advertise | mobile-advertise |
| cert-path-solicit | mobile-solicit |
| destination-unreachable | nd-na |
| echo-reply | nd-ns |
| echo-request | node-info |
| home-agent-reply | node-query |
| home-agent-request | packet-too-big |
| inv-nd-na | parameter-problem |
| inv-nd-ns | redirect |
| mcast-router-advertise | router-advertisement |
| mcast-router-solicit | router-renum |
| mcast-router-terminate | router-solicitation |
| mld-done | time-exceeded |
| mld-query | ver2-mld-report |

# vlan ipv6 access-group identifier

You can assign the same ACL to filter both inbound and outbound routed traffic, and to filter traffic on multiple VLANs.

### Syntax

`vlan` *`vid`* `ipv6 access-group` *`identifier`* `[in|out]`

`no vlan` *`vid`* `ipv6 access-group` *`identifier`* `[in|out]`

### Description

Assigns an ACL to a VLAN as an RACL to filter routed IP traffic entering or leaving the switch on that VLAN. You can use either the global configuration level or the **VLAN** context level to assign or remove an RACL.

### Options

***vid***

VLAN identification number

**`tunnel tunnel-id`**

Tunnel Identification

***identifier***

The alphanumeric name by which the ACL can be accessed. An identifier can have up to 64 characters.

**`in`**

Keyword for assigning the ACL to filter routed traffic entering the switch on the specified VLAN

**`out`**

Keyword for assigning the ACL to filter routed traffic leaving the switch on the specified VLAN

### Usage

The switch allows you to assign an "empty" ACL to a VLAN. In this case, if you later populate the empty ACL with one or more ACEs for that same identifier, the ACL automatically becomes active on the assigned VLAN. Also, where a given ACL is assigned to an interface, if you delete the ACL from the running configuration without also using the `no` form of this command to remove the assignment to the interface, the ACL becomes "empty," but remains assigned to the interface and continues to exist (as an empty ACL) in the running configuration. In this case, if you later repopulate the ACL with an explicit ACE, the ACL immediately reactivates and begins filtering traffic (which includes use of the implicit deny).

### Example output

Methods for enabling and disabling RACLs

```
Switch(config)# vlan 20 ipv6 access-group List-001 in

Switch(config)# vlan 20
Switch(vlan-20)# ipv6 access-group List-005 out
Switch(vlan-20)# exit

Switch(config)# no vlan 20 ipv6 access-group List-001 in

Switch(config)# vlan 20
Switch(vlan-20)# no ipv6 access-group List-005 out
Switch(vlan-20)# exit
```

# Filtering routed or switched IPv6 traffic inbound on a VLAN

For a given port, port list, or static port trunk, you can assign an ACL as a static port ACL to filter switched or routed IPv6 traffic entering the switch on that interface. You can use the same ACL for assignment to multiple VLANs.

### Syntax

`vlan *vid* ipv6 access-group *identifier* <vlan-in|vlan-out>`

`no vlan *vid* ipv6 access-group *identifier* <vlan-in|vlan-out>`

### Description

Assigns an ACL as a VACL to a VLAN to filter switched or routed IPv6 traffic entering or leaving the switch on that VLAN. You can use either the global configuration level or the **VLAN** context level to assign or remove a VACL.

### Options

***vid***

VLAN identification number.

***identifier***

The alphanumeric name by which the ACL can be accessed. An identifier can have up to 64 characters.

### Usage

The switch allows you to assign an "empty" ACL identifier to a VLAN. In this case, if you later populate the ACL with ACEs, the new ACEs automatically become active on the assigned VLAN as they are created. Also, if you delete an assigned ACL from the switch without also using the `no` form of this command to remove the assignment to a VLAN, the ACL assignment remains as an "empty" ACL.

### Example output

Methods for enabling and disabling VACLs

```
Switch(config)# vlan 20 ipv6 access-group List-010 vlan-in

Switch(config)# vlan 20
Switch(vlan-20)# ipv6 access-group List-015 vlan-in
Switch(vlan-20)# exit

Switch(config)# no vlan 20 ipv6 access-group List-010 vlan-in

Switch(config)# vlan 20
Switch(vlan-20)# no ipv6 access-group 015 vlan-in
Switch(vlan-20)# exit
```

## Filtering inbound IPv6 traffic per port and trunk

You can use the same ACL for assignment to multiple interfaces.

### Syntax

`interface [*port-list*|trkx] access-group *identifier* in`

`no interface [*port-list*|trkx] access-group *identifier* in`

### Description

Assigns an ACL as a static port ACL to a port, port list, or static trunk to filter switched or routed IPv6 traffic entering the switch on that interface. You can use either the global configuration level or the **interface** context level to assign or remove a static port ACL.

**Options**

***identifier***

The alphanumeric name by which the ACL can be accessed. An identifier can have up to 64 characters.

***[port-list|trkx]***

The port, trunk, or list of ports and/or trunks on which to assign or remove the specified ACL.

**Usage**

The switch allows you to assign an "empty" ACL identifier to an interface. If you later populate the empty ACL with one or more ACEs, it automatically becomes active on the assigned interfaces. Also, if you delete an assigned ACL from the running config file without also using the `no` form of this command to remove the assignment to an interface, the ACL assignment remains and automatically activates any new ACL you create with the same identifier.

**Example output**

Methods for enabling and disabling ACLs

```
Switch(config)# interface b10 ipv6 access-group List-1 in 1

Switch(config)# interface b10
Switch(eth-b10)# ipv6 access-group List-4 in 2
Switch(eth-b10)# exit

Switch(config)# no interface b10 ipv6 access-group List-1 in 3

Switch(config)# interface b10
Switch(eth-b10)# no ipv6 access-group List-4 in 4
Switch(eth-b10)# exit
```

[1] Enables a static port ACL from the Global Configuration level

[2] Enables a static port ACL from a port

[3] Disables a static port ACL from the Global Configuration level

[4] Uses a VLAN context to disable a static port

# Deleting an ACL

**Syntax**

`no ipv6 access-list` *`identifier`*

**Description**

Used in the **global config** context to remove the specified IPv6 ACL from the switch's running-config file.

**Options**

*identifier*

The alphanumeric name assigned to an ACL.

**Usage**

If an ACL name is assigned to an interface before the ACL itself has been created, the switch creates an "empty" version of the ACL in the running configuration and assigns the empty ACL to the interface. Later adding explicit

ACEs to the empty ACL causes the switch to automatically activate the ACEs as they are created and to implement the implicit deny at the end of the ACL. If an ACL name is assigned to an interface before the ACL itself has been created, the switch creates an "empty" version of the ACL in the running configuration and assigns the empty ACL to the interface. Later adding explicit ACEs to the empty ACL causes the switch to automatically activate the ACEs as they are created and to implement the implicit deny at the end of the ACL.

# [permit|deny] ipv6 -ACE-criteria

**Syntax**

`1 - 2147483647 [permit|deny] ipv6 -ACE-criteria`

**Description**

Used in the context of a given ACL, this command inserts an ACE into the ACL.

**Options**

***1 - 2147483647***

The range of valid sequence numbers for an ACL.

***ipv6-ACE-criteria***

The various traffic selection options described earlier in this chapter.

**Usage**

Entering an ACE that would result in an out-of-range sequence number is not allowed. Use the `resequence` command to free up ACE numbering availability in the ACL.

**Example**

**Inserting a New ACE in an Existing ACL**

From the **global configuration** context:

1. Insert a new ACE with a sequence number of 45 between the ACEs numbered 40 and 50 in **Appending an ACE to an existing list** on page 132.
2. From within the context of an IPv6 ACL named "List-01", insert a new ACE between two existing ACEs.
3. In this example, the first command creates a new ACL and enters the **ACL** context. The next two ACEs entered become lines 10 and 20 in the list. The third ACE entered is inserted between lines 10 and 20 by using the sequence command with a sequence number of 11.

# Deleting an ACE from an existing ACL

**Procedure**

1. To find the sequence number of the ACE you want to delete, use `show access-list` *`identifier`* or `show access-list config` to view the ACL.
2. Use `ipv6 access-list` *`identifier`* `config` to enter the IPv6 ACL (**config-ipv6-acl**) context of the specified ACE.
3. In the IPv6 ACL (**config-ipv6-acl**) context, type `no` and enter the sequence number of the ACE you want to delete.

# ipv6 access-list resequence

**Syntax**

`ipv6 access-list resequence identifier starting-seq-# interval`

**Description**

Resets the sequence numbers for all ACEs in the ACL.

**Options**

***starting-seq-#***

Specifies the sequence number for the first ACE in the list.

Default: 10; Range: 1 – 2147483647

***interval***

Specifies the interval between consecutive sequence numbers for the ACEs in the list.

Default: 10; Range: 1 – 2147483647

**Example output**

Viewing and Resequencing an ACL

```
Switch(config)# ipv6 access-list resequence My-List 100
100
Switch(config)# show access-list config
 ipv6 access-list "My-List"
      100 permit ipv6 fe80::100/128 ::/0
      200 deny ipv6 fe80::110/128 fe80::/124
      300 permit ipv6 ::/0 ::/0
    exit
```

# Remarks

A remark is numbered in the same way as an ACE and uses the same sequence number as the ACE to which it refers. This operation requires that the remark for a given ACE be entered before entering the ACE itself.

## remark remark-str

**Syntax**

`remark` *`remark-str`*

*`1 - 2147483647`* `remark` *`remark-str`*

`no` *`seq-# remark`*

**Description**

These commands are used in the **ACL** context to enter a comment related to an adjacent ACE. To associate a remark with a specific ACE, do one of the following:

- Enter the remark first (without a sequence number) and immediately follow it with the ACE (also without a sequence number). The remark and the following ACE will have the same (automatically generated) sequence number.
- Enter the ACE with or without a sequence number, then use *`1 - 2147483647`* `remark` *`remark-str`* to enter the remark, where a number in the range of *1 - 2147483647* matches the sequence number of the related ACE. This method is useful when you want to enter a remark at some time after you have entered the related ACE.

**Options**

*`remark-str`*

The text of the remark. If spaces are included in the remark, the remark string must be delimited by either single quotes or double quotes. For example:

```
remark Permits_Telnet_from_2001:db8:0:1ab_subnet
remark "Permits Telnet from 2001:db8:0:1ab subnet"
remark 'Permits Telnet from 2001:db8:0:1ab subnet'
```

*1 - 2147483647*

The range of valid sequence numbers for an ACL.

The `no` form of the command deletes the indicated remark, but does not affect the related ACE.

## Appending remarks and related ACEs to the end of an ACL

To include a remark for an ACE that will be appended to the end of the current ACL:

### Procedure

1. Enter the remark first.
2. Then enter the related ACE.
3. This results in the remark and the subsequent ACE having the same sequence number.

**Appending remarks and related ACEs to the end of an ACL**
To append an ACE with an associated remark to the end of an ACL named "List-100," enter remarks from the **CLI** context for the desired ACL:

```
Switch(config)# ipv6 access-list List-100
Switch(config-ipv6-acl)# permit tcp host 2001:db8:0:b::100:17 eq telnet any
Switch(config-ipv6-acl)# permit tcp host 2001:db8:0:b::100:23 eq telnet any
Switch(config-ipv6-acl)# remark "BLOCKS UNAUTH TELNET TRAFFIC FROM SUBNET B"
Switch(config-ipv6-acll)# deny tcp 2001:db8:0:a::/64 eq telnet any
Switch(config-ipv6-acl)# show access-list List-100 config

ipv6 access-list "List-100"
    10 remark "TEXT"
    10 permit tcp 2001:db8:0:b::100:17/128 eq 23 ::/0
    20 permit tcp 2001:db8:0:b::100:23/128 eq 23 ::/0
    30 remark "BLOCKS UNAUTH TELNET TRAFFIC FROM SUBNET B"
    30 deny tcp 2001:db8:0:b::/64 eq 23 ::/0
  exit
 Switch(config-ipv6-acl)#
```

## Inserting remarks and related ACEs within an existing list

To insert an ACE with a remark within an ACL by specifying a sequence number:

### Procedure

1. Insert the numbered remark first
2. Then, using the same sequence number, insert the ACE.

```
Switch(config-ipv6-acl)# 15 remark "PERMIT HTTP; STATION 23; SUBNET 1D"
Switch(config-ipv6-acl)# 15 permit tcp host 2001:db8:0:1d::23 eq 80
2001:db8:0:2f::/64


Switch(config-ipv6-acl)# show access config
. . .
```

```
ipv6 access-list "List-105"
    10 permit tcp 2001:db8:0:1f::/64 eq 80 2001:db8:0:2f::/64
    15 remark "PERMIT HTTP; STATION 23; SUBNET 1D"
    15 permit tcp 2001:db8:0:1d::23/128 eq 80 2001:db8:0:2f::/64 1
    20 deny tcp 2001:db8:0:1d::/64 eq 80 2001:db8:0:2f::/64
  exit
```

[1] The above two commands insert a remark with its corresponding ACE (same sequence number) between two previously configured ACEs.

## Inserting a remark for an ACE that exists in an ACL

If an ACE exists in a given ACL, you can insert a remark for that ACE by simply configuring the remark to have the same sequence number as the ACE.

## Replacing an existing remark

**Procedure**

1. Use `ipv6 access-list` *`identifier`* to enter the desired **ACL** context.
2. Configure the replacement remark with the same sequence number as the remark you want to replace.
3. This step overwrites the former remark text with the new remark text.

---

**Replacing an existing remark**

To change the text of the remark at line 15 in **Inserting remarks and related ACEs within an existing list** on page 120 to "PERMIT HTTP FROM ONE STATION", use the following command:

```
Switch(config)# ipv6 access-list List-105
Switch(config-ipv6-acl)# 15 remark "PERMIT HTTP FROM ONE STATION"
```

## Removing a remark from an existing ACE

If you want to remove a remark, but want to retain the ACE:

**Procedure**

1. Use `ipv6 access-list` *`identifier`* to enter the desired **ACL** context.
2. Use `no` *`1 - 2147483647`* `remark .`

Using the `no` *`1 - 2147483647`* command without the remark keyword deletes both the remark and the ACE to which it is attached.

## Operating notes for remarks

- An "orphan" remark is a remark that does not have an ACE counterpart with the same sequence number. The `resequence` command renumbers an orphan remark as a sequential, stand-alone entry without a permit or deny ACE counterpart.

```
ipv6 access-list "XYZ"
     10 remark "Permits HTTP"
     10 permit tcp 2001:db8::2:1/120 eq 80 ::/0
     12 remark "Denies HTTP from subnet 1."
     18 remark "Denies pop3 from 1:157."
     18 deny tcp 2001:db8::1:157/128 eq 110 ::/0 log
     50 permit ipv6 ::/0 ::/0
  exit
 Switch# ipv6 access-list resequence XYZ 100 10
```

```
 Switch# show access-list XYZ config
ipv6 access-list "XYZ"
     100 remark "Permits HTTP"
     100 permit tcp 2001:db8::2:1/120 eq 80 ::/0
     110 remark "Denies HTTP from subnet 1."
     120 remark "Denies pop3 from 1:157."
     120 deny tcp 2001:db8::1:157/128 eq 110 ::/0 log
     130 permit ipv6 ::/0 ::/0
  exit
```

- Entering either an unnumbered remark followed by a manually numbered ACE (using *1 - 2147483647* ), or the reverse (an unnumbered ACE followed by a manually numbered remark) can result in an "orphan" remark.
- Configuring two remarks without including either sequence numbers or an intervening, unnumbered ACE results in the second remark overwriting the first.

**Overwriting one remark with another**

```
Switch(config-ipv6-acl)# permit ipv6 host fe80::a1:121 fe80::/104
Switch(config-ipv6-acl)# deny tcp any eq ftp 2001:db8:0:a1::/64
Switch(config-ipv6-acl)# remark Marketing
Switch(config-ipv6-acl)# remark Channel_Mktg
Port_1_5400(config-ipv6-acl)# show access-list Accounting config

ipv6 access-list "Accounting"
    10 permit ipv6 fe80::a1:121/128 fe80::/104
    20 deny tcp ::/0 eq 21 2001:db8:0:a1::/64
    30 remark "Channel_Mktg"
  exit
```

NOTE

Where multiple remarks are sequentially entered for automatic inclusion at the end of an ACL, each successive remark replaces the previous one until an ACE is configured for automatic inclusion at the end of the list.

# Displaying ACL data

## show access-list

**Syntax**

```
show access-list
```

**Description**

Lists a summary table of the name, type, and application status of all ACLs (IPv4 and IPv6) configured on the switch.

**Example output**

```
Switch(config)# show access-list

Access Control Lists

 Type  Appl Name
 ----- ---- ------------------------
 ext   yes 101 1
 std   yes 55 2
 ext   yes Marketing 3
 ipv6  no  Accounting
```

```
 ipv6  no  List-01-Inbound
 ipv6  yes List-02-Outbound
 ipv6  yes Test-1
```

[1] IPv4 extended ACL “101”

[2] IPv4 standard ACL “55”

[3] IPv4 extended named ACL "Marketing"

| Term | Meaning |
|---|---|
| Type | Shows whether the listed ACL is an IPv6 (`ipv6`) ACL or one of two IPv4 ACL types:<br>• `std`<br>(Standard; source-address only)<br>• `ext`<br>(Extended; protocol, source, and destination data) |
| Appl | Shows whether the listed ACL has been applied to an interface (`yes/no`). |
| Name | Shows the identifier assigned to each ACL configured in the switch. |

## show access-list config

### Syntax

`show access-list config`

### Description

Lists the configuration details for every IPv4 and IPv6 ACL in the running-config file, regardless of whether any are assigned to filter traffic on specific interfaces.

### Usage

You can use the output from this command for input to an offline text file in which you can edit, add, or delete ACL commands.

This information also appears in the `show running` output. If you execute `write memory` after configuring an ACL, it appears in the `show config` output.

### Example output

This example shows the ACLs on a switch configured with two IPv6 ACLs named "Accounting" and "List-01-Inbound", and one extended IPv4 ACL named "101":

```
Switch(config)# show access-list config

ip access-list extended "101"
     10 permit tcp 10.30.133.27 0.0.0.0 0.0.0.0 255.255.255.255
     20 permit tcp 10.30.155.101 0.0.0.0 0.0.0.0 255.255.255.255
     30 deny ip 10.30.133.1 0.0.0.0 0.0.0.0 255.255.255.255 log
     40 deny ip 10.30.155.1 0.0.0.255 0.0.0.0 255.255.255.255
   exit
ipv6 access-list "Accounting"
     10 permit tcp 2001:db8:0:1af::10:14/128 ::/0 eq 23
     20 permit tcp 2001:db8:0:1af::10:23/128 ::/0 eq 23
     30 deny tcp 2001:db8:0:1af::10/116 ::/0 log
```

```
     40 permit ipv6 2001:db8:0:1af::10/116 ::/0
     50 deny ipv6 ::/0 ::/0 log
   exit
ipv6 access-list "List-01-Inbound"
     10 permit icmp fe80::10:60/128 ::/0 dscp 38
     20 permit icmp fe80::10:77/128 ::/0 dscp 38
     30 permit icmp fe80::10:83/128 ::/0 dscp 38
     40 deny icmp ::/0 ::/0 dscp 38
     50 permit ipv6 fe80::10/112 ::/0
     60 deny ipv6 fe80::/64 ::/0
   exit
```

## show access-list vlan

### Syntax

`show access-list vlan` *`vid`*

### Description

Lists the current IPv4 and IPv6 ACL assignments to the specified VLAN (in the running config file).

### Usage

This information also appears in the `show running` output. If you execute `write memory` after configuring an ACL, it also appears in the `show config` output.

### Example output

Displaying the IPv4 and IPv6 VACL assignments for a VLAN

```
Switch(config)# show access-list vlan 20

 Access Lists for VLAN 20

 Ipv6 Inbound Access List: Accounting 1

 Inbound Access List: None 2

 Ipv6 Outbound Access List: None 3

 Outbound Access List: 101 4

 Type: Extended

 Ipv6 VACL Access List: None 5

 VACL Access List: None

 Connection Rate Filter Access List: None 6
```

[1] An IPv6 ACL named “Accounting” is assigned to filter routed IPv6 traffic entering the switch on VLAN 20

[2] There is no filtering of routed IPv4 traffic entering the switch on VLAN 20

[3] There is no filtering of routed IPv6 traffic leaving the switch on VLAN 20.

[4] An extended ACL named “101” is assigned to filter routed IPv4 traffic exiting from the switch on VLAN 20

[5] There are no per-VLAN IPv6 or IPv4 ACLs assigned to VLAN 20

[6] Applies to IPv4 Connection Rate Filter ACLs. See “Virus Throttling (Connection-Rate Filtering)” in the *Access Security Guide* for your switch.

## show access-list tunnel

### Syntax

`show access-list [vlan *vid*|tunnel *tunnel-id*]`

### Description

Lists the identifiers and types of RACLs and VACLs currently assigned to a particular VLAN in the running-config file. (The switch allows one inbound and one outbound RACL assignment per VLAN, plus one VACL assignment.)

### Example output

The following output shows that inbound, routed IPv6 traffic and outbound, routed IPv4 traffic are both filtered on VLAN 20.

```
Switch(config)# show access-list vlan 20

 Access Lists for VLAN 20

 Ipv6 Inbound Access List: Accounting 1

 Inbound Access List: None 2

 Ipv6 Outbound Access List: None 3

 Outbound Access List: 101 4

 Type: Extended

 Ipv6 VACL Access List: None 5

 VACL Access List: None 6

 Connection Rate Filter Access List: None 7
```

[1] An IPv6 ACL named "Accounting" is assigned to filter routed IPv6 traffic entering the switch on VLAN 20

[2] There is no filtering of routed IPv4 traffic entering the switch on VLAN 20

[3] There is no filtering of routed IPv6 traffic leaving the switch on VLAN 20

[4] An extended ACL named "101" is assigned to filter routed IPv4 traffic exiting from the switch on VLAN 20

[5,6] There are no per-VLAN IPv6 or IPv4 ACLs assigned to VLAN 20

[7] Applies to IPv4 Connection Rate Filter ACLs. See "Virus Throttling (Connection-Rate Filtering)" in the *Access Security Guide* for your switch.

## show access-list ports

### Syntax

`show access-list ports [all|*port-list*]`

### Description

Lists the identification and types of current static port ACL assignments to individual switch ports and trunks, as configured in the running-config file. (The switch allows one static port ACL assignment per port.)

### Example output

The following output shows IPv4 and IPv6 ACLs configured on various ports and trunks on the switch:

```
Switch(config)# show access-list ports all
```

```
Access Lists for Port 1 1
Inbound Ipv6: List-01-Inbound

Access Lists for Port 12 2
Inbound : 101
Type    : Extended
Inbound Ipv6: Accounting

Access Lists for Port Trk2 3
Inbound Ipv6: Accounting

Access Lists for Port Trk5 4
Inbound : Marketing
Type    : Extended
```

[1] An IPv6 ACL is filtering inbound traffic on port 1

[2] Both an IPv4 ACL and an IPv6 ACL are filtering inbound IPv4 and IPv6 traffic, respectively, on port 12

[3] An IPv6 ACL is filtering inbound IPv6 traffic on Trunk 2 (Trk2)

[4] An IPv4 ACL is filtering inbound IPv4 traffic on Trunk 5 (Trk5)

## show access-list identifier

**Syntax**

`show access-list` *`identifier`* `[ config ]`

**Description**

Displays detailed information on the content of a specific ACL configured in the running-config file.

NOTE

This information also appears in the `show running` display. If you execute `write memory` after configuring an ACL, it also appears in the `show config` display.

For information on IPv4 ACL operation, see the latest version of the *Aruba-OS Switch Access Security Guide* for your switch.

**Example**

Viewing the content of a specific ACL

Suppose you configured the following two ACLs in the switch:

| Identifier | Type | Desired action |
|---|---|---|
| Accounting | IPv6 | • Permit Telnet traffic from these two IPv6 addresses:<br>◦ 2001:db8:0:1af::10: 14<br>◦ 2001:db8:0:1af::10: 24<br>• Deny Telnet traffic from all other devices in the same subnet.<br>• Permit all other IPv6 traffic from the subnet.<br>• Deny and log any IPv6 traffic from any other source. |
| List-120 | IPv4 Extended | • Permit any TCP traffic from 10.30.133.27 to any destination.<br>• Deny any other IP traffic from 10.30.133.(*1 - 255*).<br>• Permit all other IP traffic from any source to any destination. |

**Example**

Listing an IPv6 ACL

```
Switch(config)# show access-list Accounting
Access Control Lists
  Name: Accounting
  Type: ipv6
  Applied: Yes 1

 SEQ  Entry
--------------------------------------------------------
 10   Action: permit
      Remark: Telnet Allowed 2
      3 Src IP: 2001:db8:0:1af::10:14     4 Prefix Len: 128
      5 Dst IP: ::                        6 Prefix Len: 0
      7 Src Port(s):  8 Dst Port(s): eq 23
      9 Proto : TCP  Option(s):
      10 Dscp : -

 20   Action: permit
      Src IP: 2001:db8:0:1af::10:23     Prefix Len: 128
      Dst IP: ::                        Prefix Len: 0
      Src Port(s):  Dst Port(s): eq 23
      Proto : TCP  Option(s):
      Dscp : -

 30   Action: deny (log)
      Src IP: 2001:db8:0:1af::10        Prefix Len: 116
      Dst IP: ::                        Prefix Len: 0
      Src Port(s):  Dst Port(s):
      Proto : TCP  Option(s):
      Dscp : -

 40   Action: permit
      Src IP: 2001:db8:0:1af::10        Prefix Len: 116
      Dst IP: ::                        Prefix Len: 0
      Src Port(s):  Dst Port(s):
      Proto : IPV6
      Dscp : -
```

[1] Indicates whether the ACL is applied to an interface

[2] Remark Field (Appears if remark configured.)

[3] Source Address

[4] Source and Destination Prefix Lengths

[5,6] Destination Address

[7] TCP Source Port

[8] TCP Destination Port

[9] Protocol Data

[10] DSCP Codepoint or Precedence

**An ACL listed with the `config` option**

```
Port-1(config)# show access-list List-120 config
ip access-list extended "List-120"
 10 remark "Telnet Allowed"
 10 permit tcp 10.30.133.27 0.0.0.0 eq 23 0.0.0.0 255.255.255.255 precedence 0
established
 20 deny ip 10.30.133.1 0.0.0.255 0.0.0.0 255.255.255.255 log
 30 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
 exit
```

## Descriptions of data types included in show access-list command output

**Table 15:** *Descriptions of data types included in show access-list `acl-id` output*

| Field | Description |
|---|---|
| Name | The ACL identifier. For IPv6 ACLs, is an alphanumeric name. For IPv4 ACLs, can be a number from 1 to 199 or an alphanumeric name. |
| Type | IPv6, Standard, or Extended. IPv6 ACLs use a source and a destination address, plus IPv6 protocol specifiers.<br>• Standard ACLs are IPv4 only, and use only a source IP address.<br>• Extended ACLs are available in IPv4 only, and use both source and destination IP addressing, as well as other IP protocol specifiers. |
| Applied | `Yes`<br>means that the ACL has been applied to an interface. `No` means that the ACL exists in the switch configuration, but has not been applied to any interface, and is therefore not in use. |
| SEQ | The sequential number of the ACE in the specified ACL. |
| Entry | Lists the content of the ACEs in the selected ACL. |
| Action | Permit (forward) or deny (drop) a packet when it is compared to the criteria in the applicable ACE and found to match. Includes the optional `log` option, if used, in `deny` or `permit` actions. |
| Remark | Displays any optional remark text configured for the selected ACE. |
| IP | Used for IPv4 standard ACEs: The source IPv4 address to which the configured mask is applied to determine whether there is a match with a packet. |
| Src IP | Used for IPv6 ACEs and IPv4 extended ACEs: The source IPv6 or IPv4 address to which the configured mask is applied to determine whether there is a match with a packet. |
| Dst IP | Used for IPv6 ACEs and IPv4 extended ACEs: The source and destination IP addresses to which the corresponding configured masks are applied to determine whether there is a match with a packet. |

*Table Continued*

| Field | Description |
|---|---|
| Mask | Used in IPv4 ACEs, the mask is configured in an ACE and applied to the corresponding IP address in the ACE to determine whether a packet matches the filtering criteria. |
| Prefix Len (source and destination) | Used in IPv6 ACEs to specify the number of consecutive high-order (leftmost) bits of the source and destination addresses configured in an ACE to be used to determine a match with a packet being filtered by the ACE. |
| Proto | Used in IPv6 ACEs and IPv4 extended ACEs to specify the packet protocol type to filter. |
| Port | Used in IPv4 extended ACEs to show any TCP or UDP operator and port numbers included in the ACE. |
| Src Ports Dst Ports | Used in IPv6 ACEs to show TCP or UDP source and destination operator and port numbers included in the ACE. |
| DSCP | Used in IPv6 ACEs to show the DSCP precedence or codepoint setting, if any. |
| TOS | Used in IPv4 extended ACEs to indicate type-of-service setting, if any. |
| Precedence | Used in IPv4 extended ACEs to indicate the IP precedence setting, if any. |

## Options for applying IPv6 ACLs on the switch

To apply IPv6 ACL filtering, assign a configured IPv6 ACL to the interface on which you want the traffic filtering to occur. VLAN IPv6 traffic ACLs can be applied statically using the switch configuration. Port traffic ACLs can be applied either statically or dynamically (using a RADIUS server).

### Static ACLS

Static ACLs are configured on the switch. To apply a static ACL, assign it to an interface (VLAN or port). The switch supports three static ACL types:

- Routed IPv6 traffic ACL (RACL)An ACL configured on a VLAN to filter routed IPv6 traffic entering or leaving the switch on that interface, as well as IPv6 traffic having a destination on the switch itself. (Except for filtering IPv6 traffic to an address on the switch itself, IPv6 RACLs can operate only while IPv6 routing is enabled.
- VLAN ACL (VACL)An ACL to a VLAN to filter IPv6 traffic entering or leaving the switch on that VLAN interface and having a destination on the same VLAN. The traffic can be either switched or routed.
- Static Port ACLAn ACL assigned to a port to filter IPv6 traffic entering or leaving the switch on that port, regardless of whether the traffic is routed, switched, or addressed to a destination on the switch itself.

### RADIUS-assigned ACLs

A RADIUS-assigned ACL for filtering traffic from a specific client or group of clients is configured on a RADIUS server. When the server authenticates a client associated with that ACL, the ACL is assigned to filter the inbound IP traffic received from the authenticated client through the port on which the client is connected to the switch. If the RADIUS server supports both IPv4 and IPv6 ACEs, the ACL assigned by the server can be configured to filter both traffic types, or just the IPv4 traffic. When the client session ends, the ACL is removed from the port. The switch allows as many RADIUS-assigned ACLs on a port as it allows authenticated clients. For information on RADIUS-assigned ACLs, see the latest *Aruba-OS Switch Access Security Guide* for your switch.

## Static ACL performance monitoring

ACL statistics counters provide a means for monitoring ACL performance by using counters to display the current number of matches the switch has detected for each ACE in an ACL assigned to a switch interface. This can help, for example, to determine whether a particular traffic type is being filtered by the intended ACE in an assigned list, or if traffic from a particular device or network is being filtered as intended.

### Commands for monitoring static ACL performance

This section lists the commands for monitoring static ACL performance. To monitor RADIUS-assigned ACL performance, use either of the following commands:

**Syntax**

`show access-list radius [all|`*`port-list`*`]`

`show access-list radius [authenticator|mac-based|web-based] clients` *`port-list`* `detailed`

### [show|clear] statistics

**Syntax**

`[show|clear] statistics`

**Options**

**`show`**

Displays the current match (hit) count per ACE for the specified IPv6 or IPv4 static ACL assignment on a specific interface.

**`clear`**

Resets ACE hit counters to zero for the specified IPv6 or IPv4 static ACL assignment on a specific interface.

**Subcommands**

`aclv4` *`acl-name-str`* `port` *`port-#`*

`aclv4` *`acl-name-str`* `vlan`*`vid`* `[in|out|vlan-in|vlan-out]`

`aclv6` *`acl-name-str`* `port` *`port-#`*

`aclv6` *`acl-name-str`* `vlan`*`vid`* `[in|out|vlan-in|vlan-out]`

`aclv6` *`acl-name-str`* `tunnel` *`tunnel-id`* `[in|out]`

**Example output**

IPv6 and IPv4 ACL activity

```
Switch# show statistics aclv6 TEST-01 vlan 20 <vlan-in|vlan-out>
 HitCounts for ACL IPV6-ACL
 Total
(12) 10 permit icmp ::/0 fe80::20:2/128 128
(6)  20 deny tcp ::/0 fe80::20:2/128 eq 23 log
(41) 30 permit ipv6 ::/0 ::/0

Switch# show statistics aclv4 102 vlan 20 <vlan-in|vlan-out>

 HitCounts for ACL 102
 Total
(4)   10 permit icmp 10.10.20.3 0.0.0.0 10.10.20.2 0.0.0.0 8
(8)   20 deny icmp 0.0.0.0 255.255.255.255 10.10.20.2 0.0.0.0 8
(2)   30 permit tcp 10.10.20.3 0.0.0.255 10.10.20.2 0.0.0.255 eq 23
```

```
(2)   55 deny tcp 0.0.0.0 255.255.255.255 10.10.20.2 0.0.0.0 8
(125) 60 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
```

## ACE counter operation

For a given ACE in an assigned ACL, the counter increments by 1 each time the switch detects a packet that matches the criteria in that ACE, and it maintains a running total of the matches since the last counter reset.

**ACE counter operation**

In ACL line 10 below, there has been a total of 37 matches on the ACE since the last time the ACL's counters were reset.

```
Total
(37)        10 permit icmp ::/0 fe80::20:2/128 128
```

## About resetting ACE hit counters to zero

- Using the `clear statistics` command, see **Static ACL performance monitoring** on page 130
- Removing an ACL from an interface zeros the ACL's ACE counters for that interface only.
- For a given ACL, either of the following actions clear the ACE counters to zero for all interfaces to which the ACL is assigned:
  - Adding or removing a permit or deny ACE in the ACL.
  - Rebooting the switch.

**Resetting ACE hit counters to zero**

The following example uses the counter activity to demonstrate using `clear statistics` to reset the counters to zero.

```
Switch# show statistics aclv6 V6-02 vlan 20 vlan-in

HitCounts for ACL V6-02

Total

(5)   10 permit icmp ::/0 fe80::20:2/128 128
(4)   20 permit icmp ::/0 fe80::20:3/128 128
(136) 30 permit tcp fe80::20:1/128 ::/0 eq 23
(2)   40 deny icmp ::/0 fe80::20:1/128 128
(10)  50 deny tcp ::/0 ::/0 eq 23
(8)   60 deny icmp ::/0 ::/0 133
(155) 70 permit ipv6 ::/0 ::/0

Switch# clear statistics aclv6 V6-02 vlan 20 <vlan-in|vlan-out>
Switch# show statistics aclv6 V6-02 vlan 20 <vlan-in|vlan-out>

HitCounts for ACL V6-02

Total
(0)   10 permit icmp ::/0 fe80::20:2/128 128
(0)   20 permit icmp ::/0 fe80::20:3/128 128
(0)   30 permit tcp fe80::20:1/128 ::/0 eq 23
(0)   40 deny icmp ::/0 fe80::20:1/128 128
(0)   50 deny tcp ::/0 ::/0 eq 23
(0)   60 deny icmp ::/0 ::/0 133
(0)   70 permit ipv6 ::/0 ::/0
```

# Editing ACLs

## General editing rules

The CLI provides the capability for editing in the switch by using sequence numbers to insert or delete individual ACEs. An offline method is also available. This section describes using the CLI for editing ACLs. You can use the CLI to delete individual ACEs from anywhere in an ACL, append new ACEs to the end of an ACL, and insert new ACEs anywhere within an ACL.

- When you enter a new ACE in an ACL without specifying a sequence number, the switch inserts the ACE as the last entry in the ACL.
- When you enter a new ACE in an ACL and include a sequence number, the switch inserts the ACE according to the position of the sequence number in the current list of ACEs.
- You can delete an ACE by using the `ipv6 access-list` *`identifier`* command to enter the ACL's context, and then `no` *`seq-#`*
- Deleting the last ACE from an ACL leaves the ACL in the configuration as an "empty"ACL placeholder that cannot perform any filtering tasks. (In any ACL, the implicit deny does not apply unless the ACL includes at least one explicit ACE.

### Sequence numbering in ACLs

The ACEs in any ACL are sequentially numbered. In the default state, the sequence number of the first ACE in a list is "10," and subsequent ACEs are numbered in increments of 10. The following `show run` output shows that an ACL named "My-list" using the default numbering scheme:

**Default sequential numbering for ACEs**

```
ipv6 access-list "My-list"
     10 permit ipv6 2001:db8:0:5ad::25/128 ::/0
     20 permit ipv6 2001:db8:0:5ad::111/128 ::/0
     30 permit icmp 2001:db8:0:5ad::115/128 ::/0 135
     40 deny ipv6 2001:db8:0:5ad::/64 ::/0
   exit
```

### Appending a new ACE to the end of an ACL

```
Switch(config)# ipv6 access-list My-list permit esp host 2001:db8:0:5ad::19 any 1
Switch(Config)# ipv6 access-list My-list 2
Switch(config-ipv6-acl)# permit ipv6 any host 2001:db8:0:5ad::1
```

[1] From the global configuration prompt, appends an ACE to the end of the ACL named My-list

[2] Enters the context of the “My-list”ACL and appends an ACE to the end of the list

### Appending an ACE to an existing list

To continue from **Appending a new ACE to the end of an ACL** on page 132 and append a final ACE to the end of the ACL:

```
Switch(config-ipv6-acl)# deny ipv6 2001:db8:0:5ad::/64 any 1
Switch (config-ipv6-acl)# permit ipv6 any any 2
Switch(config-ipv6-acl)# show run
. . .
ipv6 access-list "My-list"
     10 permit ipv6 2001:db8:0:5ad::25/128 ::/0
     20 permit ipv6 2001:db8:0:5ad::111/128 ::/0
     30 permit icmp 2001:db8:0:5ad::115/128 ::/0
```

```
    40 permit icmp 2001:db8:0:5ad::/64 ::/0
    50 permit 50 2001:db8:0:5ad::19/128 ::/0
    60 permit ipv6 ::/0 2001:db8:0:5ad::1/128
    70 deny ipv6 2001:db8:0:5ad::/64 ::/0
    80 permit ipv6 ::/0 ::/0
  exit
```

[1] Appended as line 70

[2] Appended as line 80

### About viewing all ACLs and their assignments in the switch startup-config file and running-config file

The `show config` and `show running` commands include in their listings any configured ACLs and any ACL assignments to VLANs.

Remember that `show config` lists the startup-config file and `show running` lists the running-config file.

## Creating or editing an ACL offline

Using the CLI to edit an ACL is applicable in most cases where the ACL is short or there is only a minor editing task to perform. The offline method provides a useful alternative to using the CLI for creating or extensively editing a large ACL.

For longer ACLs that may be difficult or time-consuming to accurately create or edit in the CLI, you can use the offline method.

### The offline process

**Procedure**

1. Begin by doing one of the following:
   a. To edit one or more existing ACLs, use `copy command-output tftp` to copy the current version of the ACL configuration to a file in your TFTP server. For example, to copy the ACL configuration to a file named `acl-001.txt` in the TFTP directory on a server at FE80::2a1:200:

      ```
      Switch# copy command-output 'show access-list config' tftp fe80::2a1:200
      acl-001.txt pc
      ```
   b. To create a new ACL, open a text (.txt) file in the appropriate directory on a TFTP server accessible to the switch.
2. Use a text editor to create or edit the ACLs in the `*.txt` ASCII file format.
3. If you are replacing an ACL on the switch with a new ACL that uses the same number or name syntax, begin the command file with a `no ip access-list` command to remove the earlier version of the ACL from the switch's running-config file. Otherwise, the switch will append the new ACEs in the ACL you download to the existing ACL. For example, if you planned to use the `copy` command to replace an ACL named "List-120", you would place this command at the beginning of the edited file: `no ipv6 access-list List-120`
4. An offline ACL file designed to replace an existing ACL

   ```
   no ipv6 access-list List-120
   ip access-list "List-120" 1
     10 remark "THIS ACE ALLOWS TELNET"
     10 permit tcp fe80::17/128 ::/0 eq 23
     20 deny ipv6 fe80::123/128 fe80::/125 log
     30 deny ipv6 fe80::255/128 fe80::/125 log
     40 remark "THIS IS THE FINAL ACE IN THE LIST"
   ```

```
  40 permit ipv6 ::/0 ::/0
 exit
```

[1] Removes an existing ACL and replaces it with a new version with the same identifier. To append new ACEs to an existing ACL instead of replacing it, you would omit the first line and ensure that the sequence numbering for the new ACEs begin with a number greater than the highest number in the existing list.

5. Use `copy tftp command-file` to download the file as a list of commands to the switch.
   a. You would create a `.txt` file.
   b. After you copy the above `.txt` file to the TFTP server at FE80::1ad:17, you would then execute the following command: `copy tftp command-file fe80::1ad:17 acl-001.txt pc`. In this example, the CLI would show output similar to the following to indicate that the ACL was successfully downloaded to the switch.

# Testing and troubleshooting ACLs

You can monitor ACL performance by using the logging option (which generates log messages when there is a "deny" or “permit” ACE match) and the ACE statistics counters (which maintain running totals of the packet matches on each ACE in an ACL).

## Enable IPv6 ACL "Deny" or “Permit” logging

ACL logging enables the switch to generate a message when IP traffic meets the criteria for a match with an ACE that results in an explicit "deny" or “permit” action. You can use ACL logging to help:

- To help ensure that your ACL configuration is detecting and denying the incoming IPv6 traffic you do not want to enter the switch, or permitting the traffic, test your network
- Receive notification when the switch denies inbound IPv6 traffic you have designed your ACLs to reject (deny), or permits traffic you have designed your ACLs to allow (permit).

## IPv6 ACL logging

### Prerequisites

**Procedure**

1. The switch configuration must include an ACL:
   a. Assigned to a port, trunk, or static VLAN interface
   b. Containing an ACE configured with the `deny` or `permit` action and the `log` option.
2. If the RACL application is used, IPv6 routing must be enabled on the switch.
3. For IPv6 ACL logging to a syslog server:
   a. The server must be accessible to the switch and identified in the running configuration.
   b. The logging facility must be enabled for syslog.
   c. Debug must be configured to:
      - Support ACL messages
      - Send debug messages to the desired debug destination

### ACL logging operation

When the switch detects a packet match with an ACE and the ACE includes the `deny` or `permit` action and the optional *`log`* parameter, an ACL log message is sent to the designated debug destination. The first time a packet matches an ACE with `deny` or `permit` and `log` configured, the message is sent immediately to the destination and the switch starts a wait-period of approximately five minutes. (The exact duration of the period depends on how the packets are internally routed.) At the end of the collection period, the switch sends a single-line summary of any additional "deny" matches for that ACE (and any other "deny" ACEs for which the switch detected a match). If no further log messages are generated in the wait-period, the switch suspends the timer and resets itself to send a message as soon as a new "deny" match occurs.

**Content of messages generated by an ACL-deny action**

```
Example Syslog report of the first deny event detected by the switch for this ACE:

ACL 12/01/08 10:04:45 List NO-TELNET, seq#10 denied tcp 2001:db8:0:1ae::1a:3(1612)
->2001:db8:0:1ad::1a:2(23) on vlan 1, port A7
```

Example of subsequent deny events detected by the switch for the same ACE.

```
Dec 1 10:04:45 2008:db8:0:1ad::1a:1 ACL:
ACL 12/01/08 10:04:45 : ACL NO-TELNET seq#10 denied 6 packets
```

# Enabling ACL logging on the switch

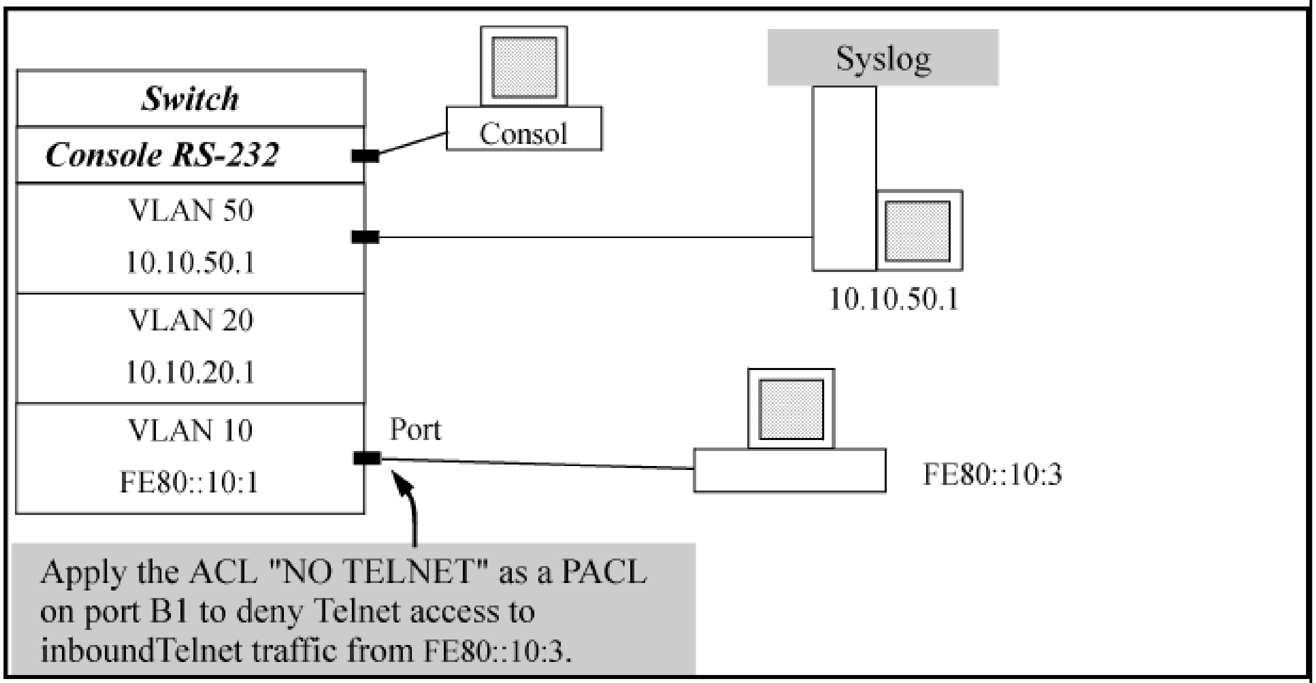
**Procedure**

1. If you are using a syslog server, use the `logging` *`ip-addr`* command to configure the syslog server IP addresses; ensure that the switch can access any syslog servers you specify.
2. Use `logging facility syslog` to enable the logging for syslog operation.
3. Use the `debug destination` command to configure one or more log destinations.
4. Destination options include logging and session. For more information on debug, see "Debug and Syslog Messaging Operation" in the appendix, "Troubleshooting", in the latest *Management and Configuration Guide* for your switch.
5. Use `debug acl` or `debug all` to configure the debug operation to include ACL messages.
6. Configure an ACL with the `deny` or `permit` action and the `log` option in one or more ACEs.

**Enabling ACL logging on the switch**
Suppose that you want to configure the following on a switch receiving IPv6 traffic and configured for IPv4 routing:

- For port B1 on VLAN 10, configure an IPv6 ACL with an ACL-ID of "NO-TELNET" and use the PACL `in` option to deny Telnet traffic entering the switch from IP address FE80::10:3.
- Configure the switch to send an ACL log message to the current console session and to a syslog server at 10.10.50.173 on VLAN 50 if the switch detects a packet match denying a Telnet attempt from FE80::10:3.

**Figure 8:** *Example of an ACL log application*




## Commands for applying an ACL with logging

```
Switch(config)# ipv6 access-list NO-TELNET
Switch(config-ipv6-acl)# remark "deny fe80::10:3 Telnet traffic."
Switch(config-ipv6-acl)# deny tcp host fe80::10:3 any eq telnet log
Switch(config-ipv6-acl)# permit ipv6 any any
Switch(config-ipv6-acl)# exit
Switch(config)# vlan 10 ipv6 access-group NO-TELNET vlan 1
Switch(config)# logging 10.10.50.173
Switch(config)# logging facility syslog
Switch(config)# debug destination logging
Switch(config)# debug destination session
Switch(config)# debug acl
Switch(config)# write mem
Switch(config)# show debug
 Debug Logging
 Destination:
 Logging --
 10.10.50.173
 Facility = syslog
 Severity = debug
 System Module = all-pass
 Priority Desc =
 Session

 Enabled debug types:
 event
 acl log
```

```
Switch(config)# show access-list NO-TELNET config

ipv6 access-list "NO-TELNET"
   10 remark "deny fe80::10:3 TELNET TRAFFIC"
   10 deny tcp fe80::10:3/128 ::/0 eq 23 log
   20 permit ipv6 ::/0 ::/0
 exit
```

[1] Assigns the ACL named “NOTELNET” as a VACL to filter Telnet traffic from FE80::10:3 entering the switch on VLAN 10

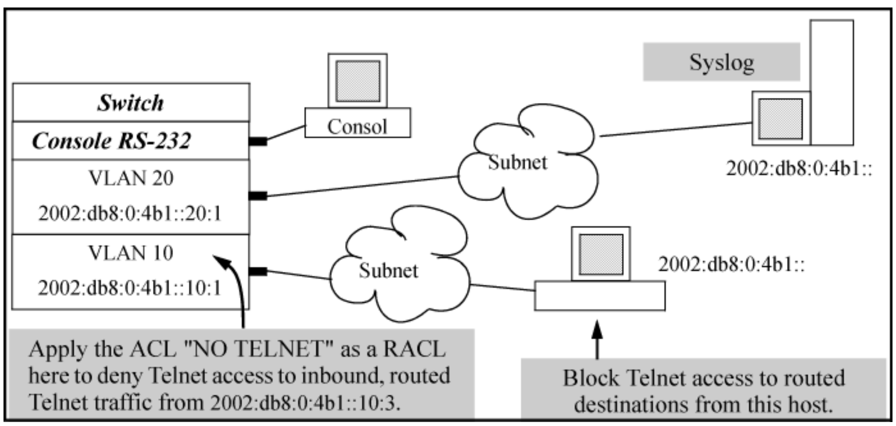
### ACL log application

Suppose that you want to configure the following operation:

- For VLAN 10, configure an ACL with an ACL-ID of "NO-TELNET" and use the RACL `in` option to deny Telnet traffic entering the switch from IP address 2001:db8:0:4b1::10:3 to any routed destination. (This assignment will not filter Telnet traffic from 2001:db8:0:4b1::10:3 to destinations on VLAN 10 itself.)
- Configure the switch to send an ACL log message to the current console session and to a syslog server at 2001:db8:0:4b1::20:3 on VLAN 20 if the switch detects a packet match denying a Telnet attempt from 2001:db8:0:4b1::10:3.

(This example assumes that IPv6 routing is already configured on the switch.)

**Figure 9:** *Example of an ACL log application*




# CIDR notation usage to enter the IPv6 ACL prefix length

CIDR (classless interdomain routing) notation is used to specify ACL prefix lengths. The switch compares the address bits specified by a prefix length for an SA or DA in an ACE with the corresponding address bits in a packet being filtered by the ACE. If the designated bits in the ACE and in the packet have identical settings, the addresses match.

**Table 16:** *Examples of CIDR notation for prefix lengths*

| SA or DA used in an ACL with CIDR notation | Resulting prefix length defining an address match | Meaning |
|---|---|---|
| 2620:0:a03:e102::/64 | 2620:0:a03:e102 | The leftmost 64 bits must match. The remaining 64 bits are wildcards. |
| 2620:0:a03:e102:215::/80 | 2620:0:a03:e102:215 | The leftmost 80 bits must match. The remaining 48 bits are wildcards. |
| 2620:0:a03:e102:215:60ff:fe7a:adc0/128 | 2620:0:a03:e102:215:60ff:fe7a:adc0 | All 128 bits must match. This specifies a single host address. |
| 2001:db8:a03:e102:0:ab4:100::/112 | 2001:db8:a03:e102:0:ab4:100 | The leftmost 112 bits must match. The remaining 16 bits are wildcards. |

**More Information**

[any|host DA|DA/prefix-length] on page 106

# IPv6 ACL configuration in a routed environment

Suppose that you want to implement these policies on a switch configured for IPv6 routing and membership in VLANs 15, 14, and 13:

Policy A

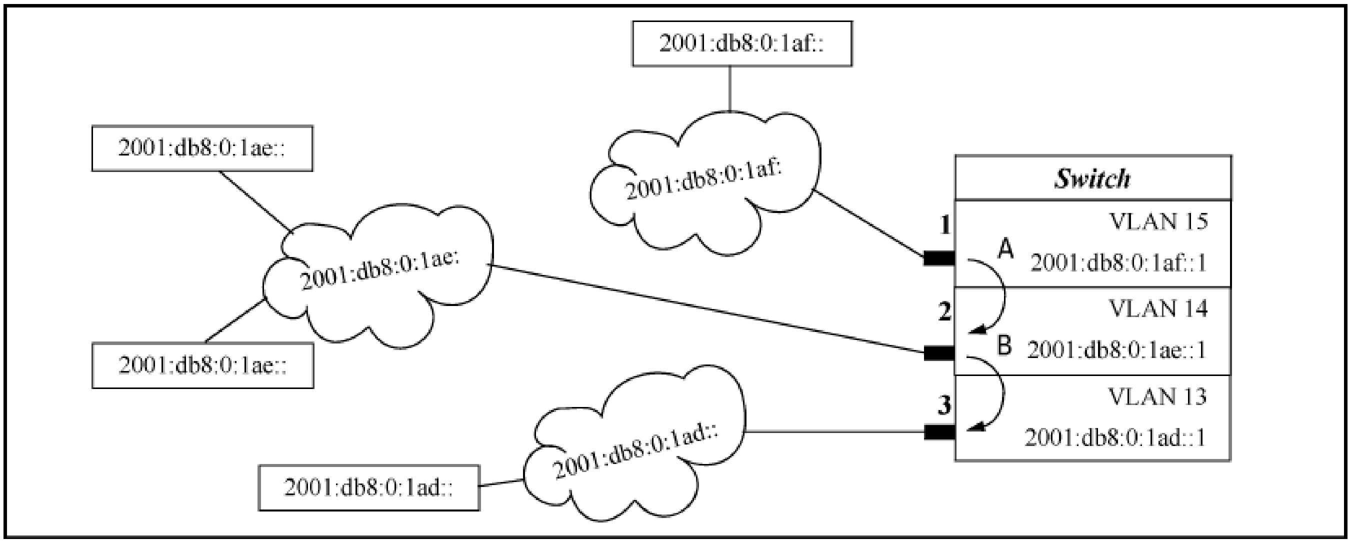
**Procedure**

1. Permit IPv6 Telnet traffic from 2001:db8:0:1af::144 to 2001:db8:0:1ae::178.
2. Deny all other IPv6 traffic from network 2001:db8:0:1af::/64 (VLAN 15) to 2001:db8:0:1ae::/64 (VLAN 14).
3. Permit all other IPv6 traffic from 2001:db8:0:1af::/64 (VLAN 15) to any destination.

Policy B

- Permit File Transfer Protocol traffic from IPv6 address 2001:db8:0:1ae::100 (on VLAN 14) to 2001:db8:0:1ad::55 (on VLAN 13). The TCP port number assigned for File Transfer Protocol traffic is "21".
- Deny File Transfer Protocol traffic from other hosts on network 2001:db8:0:1ae::/64 to any destination.
- Permit all other IPv6 traffic.

**Figure 10:** *Example of an IPv6 ACL application*




To implement the policies described above, configure ACLs on the switch as shown in below:

**Switch B shown in Example of an IPv6 ACL application**

**Switch A shown in Example of an IPv6 ACL application**

```
Switch(config-ipv6-acl)# permit tcp host 2001:db8:0:1ae::100 host 2001:db8:0:1ad::
55 eq 21
Switch(config-ipv6-acl)# deny tcp 2001:db8:0:1ae::/64 any
Switch(config-ipv6-acl)# permit ipv6 any any
Switch(config-ipv6-acl)# exit
Switch(config-ipv6-acl)# vlan 1 ipv6 access-group List-02 in
```

```
Switch(config-ipv6-acl)# permit tcp host 2001:db8:0:1af::144 host 2001:db8:0:1ae::
178 eq telnet
Switch(config-ipv6-acl)# deny ipv6 2001:db8:0:1af::/64 2001:db8:0:1ae::/64
Switch(config-ipv6-acl)# permit ipv6 2001:db8:0:1af::/64 any
Switch(config-ipv6-acl)# exit
Switch(config)# vlan 1 ipv6 access-group List-01 in
```

# IPv6 counter operation with multiple interface assignments

Where the same IPv6 ACL is assigned to multiple interfaces, the switch maintains a separate instance of each ACE counter in the ACL. When there is a match with traffic on one of the ACL's assigned interfaces, only the affected ACE counters for that interface are incremented. Other instances of the same ACL applied to other interfaces are not affected.

NOTE

The examples of counters in this section use small values to help illustrate counter operation. The counters in real-time network applications are much more active and show higher values.

**IPv6 counter operation with multiple interface assignments**

Suppose that:

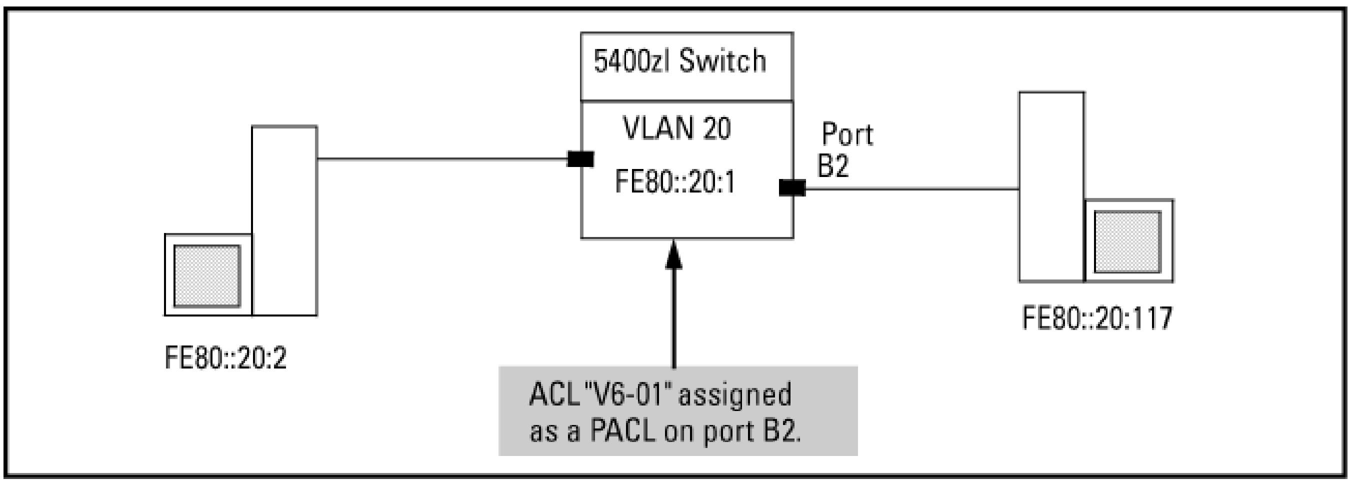
- An ACL named "V6-01" is configured as shown in the following example, to block Telnet access to a workstation at FE80::20:2, which is connected to a port belonging to VLAN 20.
- The ACL is assigned as a PACL (port ACL) on port 2, which is also a member of VLAN 20:

**ACL "V6-01" and command for PACL assignment on port 2**

```
Switch(config)# show access-list V6-01 config

ipv6 access-list "V6-01"
     10 permit icmp ::/0 fe80::20:2/128 128
     20 deny tcp ::/0 fe80::20:2/128 eq 23 log
     30 permit ipv6 ::/0 ::/0
   exit

 Switch(config)# int b2 ipv access-group V6-01 in1
```

[1] Assigns the ACL to port 2

**Figure 11:** *Application to filter traffic inbound on port B2*




Using the topology shown, a workstation at FE80::20:117 on port B2 attempting to ping and Telnet to the workstation at FE80::20:2 is filtered through the PACL instance of the "V6-01" ACL assigned to port B2, resulting in the following:

**Ping and Telnet from FE80::20:117 to FE80::20:2 filtered by the assignment of "V6-01" as a PACL on port B2**

```
Switch# ping6 fe80::20:2%vlan20
fe80:0000:0000:0000:0000:0000:0020:0002 is alive, time = 5 ms
Switch# telnet fe80::20:2%vlan20
Telnet failed: Connection timed out.
Switch#
```

**Resulting ACE hits on ACL “V6-01”**

```
Switch# show statistics aclv6 IP-01 port 2

 Hit Counts for ACL IPV6-ACL
```

```
 Total

(1)1   10 permit icmp fe80::20:3/128 fe80::20:2/128 128
(5)2   20 deny tcp ::/0 fe80::20:2/128 eq 23 log
(4)3   30 permit ipv6 ::/0 ::/0
```

[1] Shows the successful ping permitted by ACE 10

[2] Indicates denied attempts to Telnet to FE80::20:2 via the instance of the "V6-01" PACL assignment on port 2

[3] Indicates permitted attempts to reach any accessible destination via the instance of the “V6-01”PACL assignment on port 2

## IPv4 counter operation with multiple interface assignments

Where the same IPv4 ACL is assigned to multiple interfaces as a VLAN ACL (VACL) or port ACL (PACL), the switch maintains a separate instance of ACE counters for each interface assignment. Thus, when there is a match with traffic on one of the ACL's VACL- or PACL-assigned interfaces, only the ACE counter in the affected instance of the ACL is incremented. However, if an ACL has multiple assignments as an RACL, then a match with an ACE in any RACL instance of the ACL increments that same counter on all RACL-assigned instances of that ACL. (The ACE counters for VACL and PACL instances of an ACL are not affected by counter activity in RACL instances of the same ACL.)

**IPv4 counter operation with multiple interface assignments**

Suppose that an IPv4 ACL named "Test-1" is configured as shown in the following example, to block Telnet access to a server at 10.10.20.12 on VLAN 20, and that the Test-1 ACL is assigned to VLANs as follows:

- VLAN 20: VACL
- VLAN 50: RACL
- VLAN 70: RACL

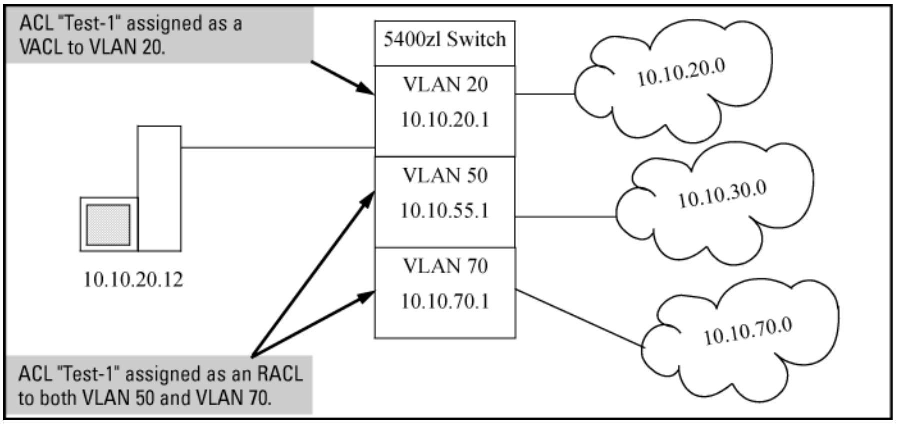
**ACL “Test-1” and interface assignment commands**

```
Switch(config)# show access-list Test1 config

ip access-list extended "Test1"
   10 deny tcp 0.0.0.0 255.255.255.255 10.10.20.12 0.0.0.0 eq 23 log
   20 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
   exit

Switch(config)# vlan 20 ip access-group Test-1 vlan-in 1
Switch(config)# vlan 50 ip access-group Test-1 in 2
Switch(config)# vlan 70 ip access-group Test-1 in
```

[1] Assigns the ACL as a VACL to VLAN 20

[2] Assigns the ACL as an RACL to VLANs 50 and 70

**Figure 12:** *Using the same IPv4 ACL for VACL and RACL applications*




In the above case:

- Matches with ACEs 10 or 20 that originate on VLAN 20 increment only the counters for the instances of these two ACEs in the Test-1 VACL assignment on VLAN 20. The same counters in the instances of ACL Test-1 assigned to VLANs 50 and 70 are not incremented.
- Any Telnet requests to 10.10.20.12 that originate on VLANs 50 or 70 are filtered by instances of Test-1 assigned as RACLs and increment the counters for ACE 10 on both RACL instances of the Test-1 ACL.

Using the network in the figure shown, a device at 10.10.20.4 on VLAN 20 attempting to ping and Telnet to 10.10.20.12 is filtered through the VACL instance of the "Test-1" ACL on VLAN 20 and results in the following:

**Ping and Telnet from 10.10.20.4 to 10.10.20.2 filtered by the assignment of "Test-1" as an IPv4 VACL on VLAN 20**

```
Switch(config)# ping 10.10.20.2
10.10.20.2 is alive, time = 5 ms
Switch(config)# telnet 10.10.20.2
Telnet failed: Connection timed out.
Switch(config)#
```

**Resulting ACE hits on IPv4 ACL “Test-1”**

```
Switch(config)# show statistics aclv4 Test-1 vlan 20 vlan-in
 Hit Counts for ACL Test-1
  Total
(5)[1]10 deny tcp 0.0.0.0 255.255.255.255 10.10.20.2 0.0.0.0 eq 23 log
(2)[2]20 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255

Switch# show statistics aclv4 Test-1 vlan 50 in
Hit Counts for ACL Test-1
Total
(0) [3]10 deny tcp 0.0.0.0 255.255.255.255 10.10.20.2 0.0.0.0 eq 23 log
(0)20 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
```

[1] Indicates denied attempts to Telnet to 10.10.20.12 filtered by the instance of the “Test-1” VACL assignment on VLAN 20

[2] Indicates permitted attempts to reach any accessible destination via the instance of the “Test- 1”VACL assignment on VLAN 20. This example shows the successful pings permitted by ACE

[3] Shows that the hits on the instance of the “Test-1” VACL assignment on VLAN 20 have no effect on the counters for the RACL assignment of “Test-1” on VLAN 50

However, using a device at 10.10.30.11 on VLAN 50 for attempts to ping and Telnet to 10.10.20.12 requires routing and filters the attempts through the RACL instance of the “Test-1” ACL on VLAN 50.

**Ping and Telnet from 10.10.30.11 to 10.10.20.2 filtered by the assignment of "Test-1" as an IPv4 RACL on VLAN 30**

```
Switch# ping 10.10.20.2
10.10.20.2 is alive, time = 25 ms
Switch# telnet 10.10.20.2
Telnet failed: Connection timed out.
Switch#
```

This action has an identical effect on the counters in all RACL instances of the "Test-1" ACL configured and assigned to interfaces on the same switch. In this example, it means that the RACL assignments of "Test-1" on VLANs 50 and 70 are incremented by the above action occurring on VLAN 50.

**Resulting ACE hits on the VLAN 30 IPv4 RACL assignment of the "Test-1" ACL**

```
Switch(config)# show statistics aclv4 Test-1 vlan 50 in
Hit Counts for ACL Test-1
Total

(6) 10 deny tcp 0.0.0.0 255.255.255.255 10.10.20.2 0.0.0.0 eq 23 log
(1) 20 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
Switch(config)#
```

NOTE

The `Total 6` indicates the same type of data for the VACL assignment of the “Test-1” ACL. That is, the Ping attempt incremented the counters for ACE 20 and the Telnet attempt incremented the counters for ACE 10 in the VLAN 50 RACL instance of the ACL.

**Resulting ACE hits on the VLAN 70 IPv4 RACL assignment of the "Test-1" ACL**

```
Switch(config)# show statistics aclv4 Test-1 vlan 70 in
HitCounts for ACL Test-1
Total
(6) 10 deny tcp 0.0.0.0 255.255.255.255 10.10.20.2 0.0.0.0 eq 23 log
(1) 20 permit ip 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255
Switch(config)#
```

# IPv6 routing overview

Beginning with software release K.15.01, the switches support these IPv6 routing features:

- **IPv6 static routing** on page 155
- **Router advertisements** on page 30
- **DHCPv6-relay** on page 151
- **OSPFv3 routing** on page 185

**More Information**

**Router access and default router selection** on page 30

## Dual stack IPv4/IPv6 operation

The switches support IPv4/IPv6 dual-stack operation. This allows full Ethernet link support for switching and routing both IPv4 and IPv6 traffic on the same VLAN interfaces configured on the switch without modifying current IPv4 network topologies. This enables you to use IPv6 devices on existing VLANs, manage the switches and other devices from IPv6 management stations, and create dedicated groups of IPv6 devices as needed to accommodate the need for the IPv6 network growth anticipated for the future.

# show ip ospf

**Syntax**

`show ip ospf`

**Description**

This command is used when OSPFv2 is enabled.

**Options**

`show ip ospf3`

This command is used when OSPFv3 is enabled.

**Usage**

If one of the following is true, the router ID is set to 0.0.0.0:

- No manual router ID, IPv4 network address, or IPv4 loopback interface address is configured on the routing switch.
- No dynamic routing protocol is enabled on the routing switch.

**Example**

Displaying the router ID when OSPFv3 is enabled

If a routing switch is using a router ID such as 10.10.10.1:

```
Switch(config)# show ipv6 ospf3

 OSPFv3 General Status
```

```
OSPFv3 Protocol    : Enabled
Router ID          : 10.10.10.1
```

# no ip router id

You can override the current router ID assignment by explicitly configuring the router ID to any valid IPv4 address. (This address must be unique; not configured on another device in the network.)

**Syntax**

`ip router-id` *n n n n*

`no ip router-id` *n n n n*

**Description**

This command configures a router ID for use by any dynamic routing protocol configured on the routing switch. The assignment applies across all routable VLANs on the routing switch, and uses the IPv4 32-bit dotted decimal format. The resulting IPv4 address must be unique in the network.

This `no` form of the command replaces a manually configured router ID with the first-detected IPv4 network address configured on a VLAN. If an IPv4 address is not detected on a VLAN, then the lowest-numbered IPv4 address of the lowest-numbered IPv4 loopback interface is used.

# Changing an existing router ID

**Procedure**

1. Go to the **global config** context.
2. The CLI prompt will appear similar to the following:
3. `Switch(config)#_`
4. Use `ip router-id` *n.n.n.n* to specify a new router ID.
5. This must be in the IPv4 (32-bit) dotted-decimal address format and must be unique in the routing switch configuration and your network. For example:
   - If both OSPFv2 and OSPFv3 are enabled, you will see the following:

     ```
     OSPFv3 and OSPFv2 are running. Router-id will be applied immediately.
     Protocol will be reset. Continue [y/n]?
     Press Y to continue, or press N to exit from the command without changing the
     ID.
     ```

   - If a routing protocol is not enabled, the new ID is entered and the global config prompt is displayed.

# Commands to view manually configured router ID

Use one of the following commands to view a manually configured router ID:

**Syntax**

`show run`

or

`show ip ospf` (if OSPFv2 is enabled)

or

`show ip ospf3` (if OSPFv3 is enabled)

# ipv6 hop-limit

### Syntax

`ipv6 hop-limit *1 - 255*`

`no ipv6 hop-limit *1 - 255*`

### Description

### Global config operation

This global config command sets the maximum number of routers (hops) through which packets originating on the routing switch can pass before being discarded (global hop limit).

Each router decrements a packet's hop limit by 1 before forwarding the packet. If decrementing the hop limit causes it to go to 0 (zero), the decrementing router drops the packet instead of forwarding it.

### Effect on the Hop Limit Included in Per Router Advertisements (RAs)

If the routing switch is enabled to send RAs on a given IP routing interface, and that interface's RA configuration does not include a hop limit entry, then the global hop limit configured by this command is inserted in the RAs sent from the routing switch on that interface.

But if the interface's RA configuration does include a local hop limit entry, then the global config hop limit is replaced by the local hop limit entry configured for inclusion in RAs sent by the routing switch on that interface.

### Options

Default: 64; Range: *1 - 255*

The `no` form of the command resets the global hop-limit to the default 64.

# ipv6-gateway-addr

The IPv6 default route (::/0) is a static route used for all traffic that has a destination network not reachable through any other IPv6 route in the routing table.

### Syntax

`ipv6 route ::/0 *ipv6-gateway-addr* distance *1 - 255*`

### Description

Used in the **global config** context to configure the default route and gateway to use for traffic sent to the default route.

### Options

**`::/0`**

Specifies the default IPv6 route.

***`ipv6-gateway-addr`***

Specifies the next-hop router for traffic sent to the default route.

**`distance *1 - 255*`**

Specifies the administrative distance to associate with a static route.

Default: 1; Range: *1 - 255*

### Example input

Configuring the IPv6 default route

If 2001:db8:c::9f:35 is the IPv6 address of your ISP router, all nonlocal traffic could be directed to the ISP by configuring the following default route:

```
Switch(config)# ipv6 route ::/0 2001:db8:c::9f:35
```

To view the default route in the routing table, use `show ipv6 route`.

# show ipv6 route

### Syntax

```
show ipv6 route
```

### Description

Displays the current IPv6 routing table content for the routing switch.

Where there are multiple routes to the same destination, only the route with the lowest administrative distance is entered in the routing table and used to forward traffic to that destination.

### Options

[ *ipv6-destination* [connected|static|ospf3]]

[ connected *ipv6-destination* ]

[ static *ipv6-destination* ]

[ ospf3 *ipv6-destination* ]

### Usage

The complete IPv6 route table is displayed by entering the CLI command `show ipv6 route` from any context level in the console CLI.

### Example output

```
Switch(config)# show ipv6 route

                        IPv6 Route Entries
Destination : ::/0
Gateway : 2001:db8:e::55:2
Type: static      Sub-Type: NA         Distance: 130 1
Metric: 1

Destination : ::1/128
Gateway : lo0
Type: connected   Sub-Type: NA         Distance: 0    Metric: 1

Destination : 2001:db8:1::127/128
Gateway : lo6
Type: connected   Sub-Type: NA         Distance: 0    Metric: 1

Destination : 2001:db8:a::/64
Gateway : fe80::22:1%vlan22
Type: ospf3       Sub-Type: InterArea 2

Distance: 110  Metric: 2

Destination : 2001:db8:b::/64
Gateway : VLAN22
Type: connected   Sub-Type: NA         Distance: 0 3  Metric: 1

Destination : 2001:db8:c::/64
Gateway : 2001:db8:e::55:2
```

```
Type: static     Sub-Type: NA         Distance: 120  Metric: 1
```

[1] The Default Route configured with a Non-Default Distance

[2] OSPFv3 Route, with "InterArea" Sub-Type and Default Distance

[3] Static Route with Non-Default Distance

# Enabling IPv6 routing

**Procedure**

1. On each VLAN, configure stateless address autoconfiguration and at least one IPv6 global unicast address:
2. `vlan n ipv6 address autoconfig`
3. `vlan n ipv6 address prefix/prefix-length eui-64`
4. These commands result in a link-local address and a global unicast address having an interface ID derived from the routing switch's MAC address.
5. Suppress automatic (default) RAs on VLAN interfaces where you need to make configuration changes or where you do not currently want these advertisements generated:
   - To globally suppress RAs on the routing switch, use this command in the **global config** context:
   - To suppress RAs on individual VLANs, use this command in the context of each VLAN where you want the advertisements suppressed:`ipv6 nd ra suppress`The `no` form of the above two `suppress` commands disables RA suppression. For more information on RAs, see **IPv6 router advertisements** on page 161.
6. Enable IPv6 routing. (This command enables RA transmission on any VLAN where RAs are not suppressed.)
7. `ipv6 unicast-routing`
8. For nondefault RA operation, configure RAs per-VLAN, including suppression of RAs on any VLANs where you do not want the routing switch to transmit RAs.
9. Configure one or more of the following routing features:
   - IPv6 static routing. See **IPv6 static routing** on page 155.
   - DHCPv6-relay. See **DHCPv6-relay** on page 151.
   - OSPFv3. See **OSPFv3 routing** on page 185.

# Configuring a router ID

When `router ospf3 enable` is used to enable OSPFv3 on a routing switch, the following message appears if an IPv4 network or loopback address is not detected:

```
Either an IPv4 loopback address or a router ID needs to be configured before
enabling OSPFv3.
Do you want to continue? [y/n]
```

**Procedure**

1. Type **N** (no) to keep OSPFv3 disabled.
2. Type **Y** to enable OSPFv3 without a router ID. However, without the ID, OSPFv3 traffic is not routed. In this case do one of the following:
   - Configure an IPv4 address on a loopback interface or VLAN.
   - Manually configure a router ID on the routing switch.

# IPv6 networks and subnets

An IPv6 network is a group of hosts and routers that share a common network prefix and exist on the same VLAN interface. Where multiple unique network prefixes exist on the VLAN, each prefix corresponds to a different subnet. For example, if a given network has a prefix of **2001:db8:1ad:27b::/64**, any global unicast address for an individual device belonging to this network has:

- The same prefix (**2001:db8:1ad:27b::/64**).
- A unique value (for the interface ID) in the remaining 64 bits of the global unicast addresses.

In the above case, if device "A" has an interface ID of **218:71ff:fedd:cf00**, its complete global unicast address is:

2001:db8:1ad:27b:218:71ff:fedd:cf00/64

2001:db8:1ad:27b: Prefix

218:71ff:fedd:cf00: Interface ID

/64: Prefix Length

Traffic between hosts on the same network is switched and employs link-local addresses that include a reserved prefix (FE80::/64) and a unique interface ID generated from the device MAC address. Continuing the example from above, device "A" uses the following link-local address for switching:

However, when a packet must be sent from one network to another, where the source and destination have different IPv6 network prefixes, the packet must be routed. For example, routing is required to send traffic between the devices at these two addresses residing in different networks:

```
2620:0:a03:e102:218:71ff:fedd:cf00/64
1
```

```
2001:0:db8:17fd:218:71ff:fedd:cf00/64
2
```

[1,2] Prefix

## VLANs and routing

On the routing switches covered by this guide, IPv6 addresses are associated with individual IP routing interfaces. Link-local addresses are used for switching traffic among devices on the same IP routing interface, and global unicast addresses are used for routing traffic between different IP routing interfaces.

## Link-local

Only one link-local IPv6 address, such as fe80::215:60ff:fe7a:adc0, can be configured on a given VLAN on a routing switch.

## Global unicast

Multiple global unicast addresses can be configured on the same VLAN interface, as long as the network prefix for each address occurs only once on the routing switch.

That is, for global unicast addressing:

- The same interface identifier can be used with multiple, unique network prefixes (and the link-local address) on any VLAN.
- Different VLANs must be configured with different network prefixes.
- Only one instance of a given network prefix can be configured on a routing switch.

To summarize these rules:

| IPv6 address type | Limit | Application |
|---|---|---|
| link-local | one per IP routing interface | Can be either unique or a duplicate of link-local addresses configured on other VLANs on the routing switch. |
| global unicast | multiple per IP routing interface | The network prefix must be unique for each global unicast address configured on a given routing switch. |

For the maximum number of IPv6 addresses configurable on the routing switch and on a given IP routing interface, see the *Aruba-OS Switch Management and Configuration Guide* for your routing switch.

**Global unicast configuration**
You can configure both of the following addresses on either the same VLAN or on different VLANs:

```
2001:db8:0:1f::1:6/64(prefix = 2001:db8:0:1f::/64)
```

```
2001:db8:0:2f::1:6/64(prefix = 2001:db8:0:2f::/64)
```

## IPv6 management interface

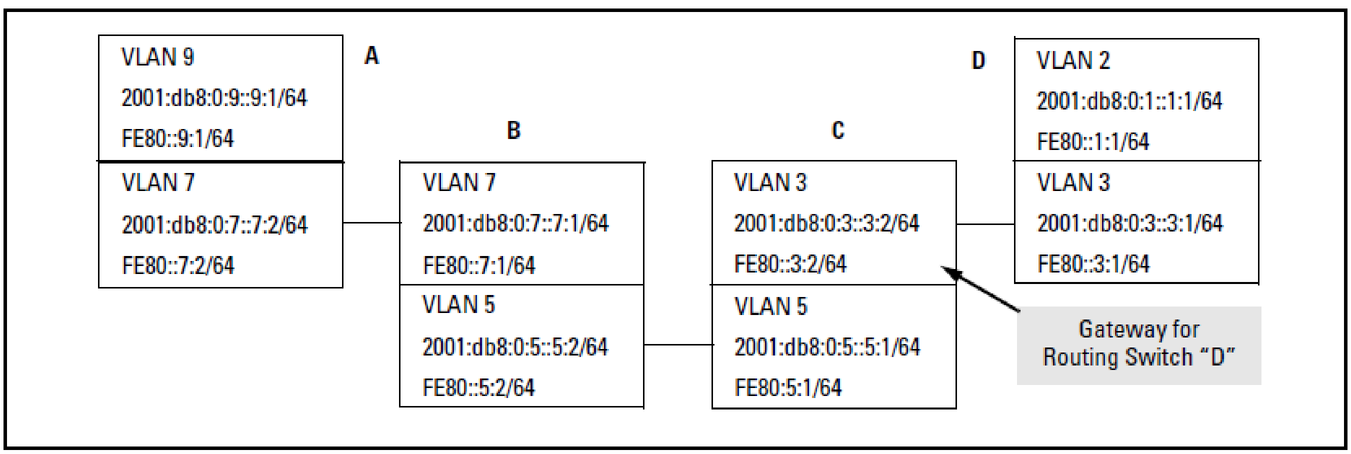
In the default configuration, there is a single VLAN (Default_VLAN; VID=1) on the routing switch. With only the default VLAN configured, a single link-local IPv6 address serves as the management access address for the entire device. If routing is enabled, the global unicast address on this VLAN also acts as the routing interface.

# IPv6 routing operation

A switch moves packets within the same local network or subnet. A router moves packets between networks or subnets. When a router receives a packet, it matches the packet's destination address to a route in its routing table. This route specifies the gateway, or next-hop, through which the router must forward the packet to enable it to move toward its destination. The gateway is specified as either the next-hop link-local address on a shared VLAN, the ID of that VLAN (vid), or a global unicast address.

For example, in the following figure, the gateway for a packet moving through router "D" to the 2001:db8:0:9::/64 network (router "A") is either VLAN 3 or the link-local address FE80::3:2/64 in VLAN 3 on router "C."

**Figure 13:** *Example of a routing domain*




A routing switch maintains a routing table containing the best routes to the destinations it has acquired. The routing table can be built from statically configured routes and dynamically configured OSPFv3 routes. Static routes must be manually configured and are best suited for small networks having few routes and where topology changes are infrequent. A dynamic routing protocol such as OSPFv3 offers scalable control for discovering and assessing reliable routes. The best route to a given destination may change over time, and dynamic routing protocols can react to such changes by replacing routes in the routing table. Dynamic routing also adapts to changes in network topology, while static routing requires manual configuration changes to support topology changes.

### Concurrent static and dynamic routing operation

Static and dynamic routing can operate concurrently in a network. Where there are both static and dynamic routes to the same destination, the routing switch selects the route with the lowest administrative distance for inclusion in the routing table. If the selected route goes down, the routing switch can replace it with a previously existing alternative route.

### Router Advertisements (RAs)

RAs are used in both static and dynamic routing environments and are transmitted per-IP routing interface from the routing switch for host configuration. RAs carry configuration settings for parameters, such as network prefix and neighbor discovery, and are also used to direct hosts to DHCPv6 servers for configuration settings.

#### Commands to enable IPv6 routing automatically

Enabling IPv6 routing automatically enables RA transmission on all IP routing interface where it has not been suppressed with one of the following commands:

**Global Config** Context:

```
Switch(config)# ipv6 nd suppress-ra
```

**VLAN Config** Context:

```
Switch(vlan-1)# ipv6 nd ra suppress
```

**Tunnel Config** Context:

```
Switch(tunnel-3)# ipv6 nd ra suppress
```

### DHCPv6-relay

When a host on a given VLAN is configured to acquire configuration settings from a DHCPv6 server, it transmits a DHCPv6 request on the VLAN. If there is no DHCPv6 server on the VLAN, you can route the host request to a server on another VLAN by enabling a DHCPv6-relay on the routing switch and configuring a helper address on the VLAN.

The settings in the `managed-config-flag` and `other-config-flag` RA options override an enabled DHCPv6 option in a host on the VLAN.

## Global IPv6 routing parameters configuration

| Feature | Default and range | Page |
|---|---|---|
| IPv6 hop-limit | 255 (*1 - 255*) | **ipv6 hop-limit** on page 146 |
| Default network route | None configured | **ipv6-gateway-addr** on page 146 |
| Router ID | Lowest-numbered address on the lowest-numbered routing interface | **System router ID** on page 152 |

## System router ID

Each routing switch uses a unique router ID to identify itself when exchanging route information with other devices. This ID is formatted as a 32-bit dotted-decimal (IPv4 format) number. For example:

```
10.100.215.1
```

An automatically assigned ID is used unless overridden by a manually configured ID.

### Automatic router ID selection

The first detected IPv4 address becomes the router ID. Prior to a reboot, this can be any IPv4 address configured on the routing switch. Following a reboot, the router ID is set to the lowest IPv4 loopback address detected.

If multiple IPv4 loopback addresses are detected at reboot, the address configured on the lowest-numbered IPv4 loopback interface (lo0 through lo7) becomes the router ID. If the lowest-numbered loopback interface has multiple IPv4 addresses, the lowest of these addresses is selected as the router ID. Once a router ID is selected, it does not automatically change unless a higher-priority address is configured on the routing switch and the routing protocol is restarted with a reboot.

# Different route types in the IPv6 routing table

**Default route (::/0)**

A static route used by all traffic that has a destination network not reachable through any other IPv6 route in the routing table.

**Directly connected routes**

Destinations on the router itself. One route is automatically entered per configured IPv6 interface. Each such route is automatically assigned an administrative distance of "0" and a metric of "1". Directly connected routes include:

**IP routing interface**

Where the routing switch is connected to a next-hop router on the same interface, a route is automatically entered for the network on which the IP routing is configured. This includes destinations for both global unicast and link-local addresses configured on the routing switch for that interface.

**Manually configured IPv6 loopback interfaces**

IPv6 loopback interfaces that are manually configured.

**Loopback route**

A static IPv6 route automatically created in the routing table for use if other routes to a destination are not available. The gateway is a loopback interface (lo0) and the destination is ::1/128.

**Statically configured routes**

On a given routing switch, one static route can be configured directly into the routing table for each destination. In the default configuration, administrative distance and route metric are both "1". See **ipv6-gateway-addr** on page 146.

**OSPFv3**

If OSPFv3 is enabled, the routing switch learns of routes from the advertisements other OSPFv3 routers transmit. If the OSPFv3 route has a lower administrative distance than any other routes from different sources to the same destination, the routing switch places the route in the IPv6 route table. See **Metric and administrative distance** on page 154.

## Routing table content

A routing protocol such as OSPFv3 develops its own database of routes. When the protocol has more than one route to a destination, it selects the route with the lowest administrative distance and inserts this route into the routing table. For each such route, the routing table maintains the following data:

| Parameter | Use |
|---|---|
| Destination (IPv6 network prefix) | Composed of the contiguous, high-order bits in a packet's destination network prefix that must match the destination network prefix in the routing table entry. For example:<br>`Address:   2001:db8:1ad:0:218:71ff:fedd:cf00/64`<br>`Prefix:    2001:db8:1ad:0/64`<br><br>`Address:   2626:17b:1:1: 218:71ff:fedd:cf00/48`<br>`Prefix:    2626:17b:1/48` |
| Gateway | The next-hop router in the path to the destination. It can be either the IPv6 address of the next directly connected router or the IP routing interface to use for forwarding the routed traffic toward its destination. If an IPv6 address is used, it can be either the link-local or global unicast address of the interface on the next-hop router. |
| Type (route type) | **Connected:**<br>A destination configured on the routing switch itself and can be a loopback interface, a global unicast address, or a link-local address.<br>**Static:**<br>A manually configured route to a destination on another router.<br>**OSPF3:**<br>A route discovered by the OSPFv3 protocol running on the routing switch. |
| Subtype | Applies to OSPFv3 routes only. |
| Distance (Administrative Distance) | Used to compare routes to the same destination, learned by different routing methods, to select the best route. The distance for connected routes is always 0. The default distance for static and dynamic routes is configurable; Default: 1. |
| Metric | Calculated by the routing switch and used to compare different routes, learned by the same routing method, to select the best overall route. |

## Destination network

Destination network prefixes identify the networks known to a routing switch. When the routing switch receives a packet for routing, it matches the packet's destination address to a network prefix in the routing table and forwards the packet to the indicated gateway for that network. The prefix in the routing table defines how many leftmost contiguous bits to use when matching a packet's destination address to a destination network prefix. For example, a route table entry of

```
2001:db8:0:1ad:0:f1:7a:0/112
```

applies to all packets with a destination address for which the first 112 bits are

```
2001:db8:0:1ad::f1:7a
```

If a packet matches more than one routing table entry, the router uses the most specific route (the route with the longest prefix), which is assumed to be the most accurate for that packet. For example, for the packet destination listed below, both route table entries apply, but the route selected will be the 72-bit entry, because it is the more specific route.

```
Packet destination address:   2001:db8:0:1d5:a15::f:101/64
72-bit entry in route table:  2001:db8:0:1d5:a00::/72
64-bit entry in route table:  2001:db8:0:1d5::/64
```

## Gateway for forwarding routed traffic

The gateway to a destination network can be either of the following:

- Global unicast or link-local address of the next-hop router in the IP routing providing a path to the packet destination.
- The IP routing interconnecting the originating router to the next-hop router.

## Metric and administrative distance

The routing table contains the single best route to each destination that the router has learned. However, a router may learn more than one route to the same destination. The router compares the metrics and administrative distances of these routes to select the best route to add to its routing table.

- Administrative distanceThe routing switch uses this parameter to compare routes learned by different routing methods. It indicates how reliable the router considers the method through which it discovered the route: a lower value indicates a more trustworthy route.Administrative distance is not a factor if you are using only static routes. However, if you are using static routing in conjunction with a routing protocol such as OSPFv3 to provide routes to an identical destination, the routing switch selects the route with the lowest administrative distance. Where the default administrative distances are used, a static route normally supersedes a dynamic route to the same destination because the former has the lowest default administrative distance and metric.

| Routing method | Administrative distance | Metric |
|---|---|---|
| Direct connection | 0 (not configurable) | 1 |
| Static route | Default: 1; range: 1 to 255 | 1 |
| OSPFv3 | Default: 110 (external, inter-area, and intra-area) range (for all three): 1 to 255 | Variable |

- Metric: The routing switch uses this parameter to compare routes to identical destinations learned by the same routing protocol. The metric is the cost of sending traffic on a given route and is based on various criteria:
  - Link conditions (bandwidth, delay, reliability)
  - Organizational policies (monetary cost, autonomous systems that a packet must traverse)Each routing protocol has its own method for computing a route's metric. For static routes, the metric defaults to "1" and is not configurable.

# ipv6 route dest

This feature enables you to create static routes (including null routes with or without ICMP notification to the sender) by adding such routes directly to the route table in the routing switch.

**Syntax**

ipv6 route *dest-ipv6-addr/prefix-length* [next-hop-gateway-addr|vlan *vid*|tunnel *tunnel-id*|blackhole|reject][distance *1 - 255*][name*string*][tag*tag*-value ]

no ipv6 route *dest-ipv6-addr/prefix-length* [next-hop-gateway-addr|vlan *vid*|tunnel *tunnel-id*|blackhole|reject][distance *1 - 255*][name*string*][tag*tag*-value ]

**Options**

***dest-ipv6-addr/prefix-length***

Network prefix for the destination on IPv6 address.

**[next-hop-gateway-addr|vlan *vid*|tunnel *tunnel-id***

The gateway for reaching the destination.

The next-hop address option (link-local or global unicast) is not required to be directly reachable on a local subnet. (If it is not directly reachable, the route is added to the routing table when a path to this address is learned.)

If the next-hop address is link-local, it must include both the address and the applicable VLAN VID.

For example: FE80::127%vlan10, where VLAN 10 is the interface where FE80::127 exists. For a tunnel, it would be FE80::127%tun3.

**blackhole**

Specifies a null route where IP traffic for the specified destination is discarded and no ICMP error notification is returned to the sender.

**reject**

Specifies a null route where IP traffic for the specified destination is discarded and an ICMP error notification is returned to the sender.

**distance *1 - 255***

Specifies the administrative distance to associate with a static route.

Default: 1; Range: *1 - 255*

The example below configures two static routes for traffic delivery and identifies two other null routes for which traffic should be discarded instead of forwarded.

**Example input**

Configuring static routes

```
Switch(config)# ipv6 route 2001:db8:0:1::/64 fe80::10.1
```

Configures static route to a specific destination network . Notice that the next-hop gateway can be either a link-local or a global unicast address.

```
Switch(config)# ipv6 route 2001:db8:0:2::/64 reject
```

Configures a null route to drop traffic for the 2001:db8:0:2::/64 network and return an ICMP notice to the sender.

```
Switch(config)# ipv6 route 2001:db8:0:5::/64 blackhole
```

Configures a null route to drop traffic for the 2001:db8:0:2::/64 network without ICMP notice to the sender.

```
Switch(config)# ipv6 route 2001:db8::/48 vlan 66 distance 120
```

Configures a static route for traffic to destinations in the 2001:db8:0::/48 network. Sets the administrative distance higher than the default distance for any dynamic routes discovered for the same destination, which gives precedence in the routing table to dynamic routes.

```
Switch(config)# ipv6 route 2001:db8:0:3::/64 tunnel 3 distance 130
```

Configures a static route for traffic to destinations in the 2001:db8:0:3::/64 network.

# show ipv6 route

### Syntax

show ipv6 route [*ipv6-addr*|connected|static|ospf3]

### Description

Lists all entries in the IPv6 routing table.

### Options

***ipv6-addr***

Lists entries for a specific IPv6 address. Can be followed by any of the other options for this command.

**connected**

Lists entries for connected routes. Can be followed by the ipv6-*addr* option to list only the connected routes having a specific link-local or global IPv6 address.

**static**

Lists entries for static routes in the routing table. Can be followed by the ipv6-*addr* option to list only the static routes matching a specific destination.

**ospf3**

Lists the entries for OSPFv3 routes in the routing table. Can be followed by the ipv6-*addr* option to list only the OSPFv3 routes matching a specific destination.

### Example output

This example lists all entries in the IPv6 routing table.

```
Switch(config)# show ipv6 route static

                              IPv6 Route Entries

 Destination : 2620:a::/64
 Gateway : 2620:b::22:1
 Type : static    Sub-Type : NA    Distance : 1   Metric : 1

 Destination : 2620:c::/64
 Gateway : 2620:e::55:2
 Type : static    Sub-Type : NA    Distance : 1   Metric : 1
```

# Static routing overview

Static routes provide tools for restricting and troubleshooting routed traffic flows and in small networks can provide the simplest and most reliable configuration for IPv6 routing.

Static routes are manually configured in the routing table. A static route entry comprises the following:

- IPv6 network prefix for the route's destination network
- Next-hop gateway, which can be one of the following:
  - Either the link-local address and VLAN ID or the VLAN link to the next-hop router
  - Global unicast address on the next-hop router
  - A "null" interface (the routing switch drops traffic forwarded to the null interface)
- Optionally, a nondefault administrative distance

NOTE

To enable routing in both directions on a static route, you must configure reciprocal static routes on the routers at both ends of the route.

On a given routing switch you can create one static route or null route to a given destination. Multiple static or null routes to the same destination are not supported.

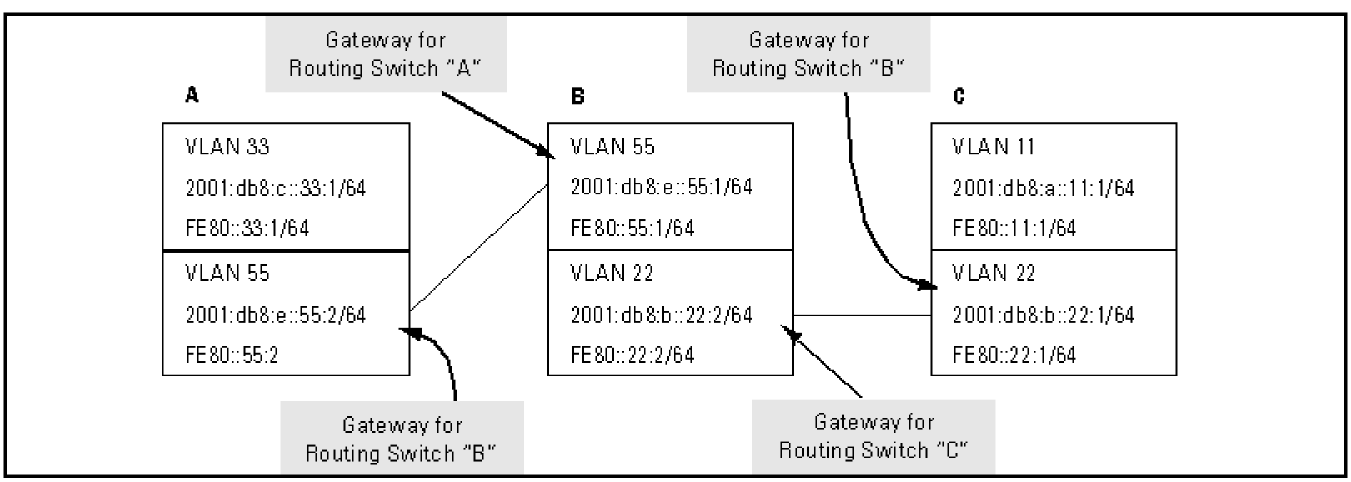
The routing switches can concurrently support a maximum of 256 IPv6 static routes and 256 IPv4 static routes.

For example, in the following figure, static routes enabling routed traffic between routers "A," "B," and "C" could be configured as follows:

**Table 17:** *Example of static route configuration in a network*

| Router "A" | Router "B" | Router "C" |
|---|---|---|
| ipv6 route2620:a::/64 2620:e::55:1 | ipv6 route2620:a::/64 2620:b:: 22:1 | ipv6 route2620:c::/64 2620:b:: 22:2 |
| ipv6 route 2620:b::/ 642620:e::55:1 | ipv6 route 2620:c::/64 2620:e:: 55:2 | ipv6 route 2620:e::/64 2620:b:: 22:2 |
| Note: Next-hop addresses can be either global unicast or link-local. | | |

**Figure 14:** *Example of a routing domain*



## Advantages of static routing

Static routing is relatively reliable and gives you tight control over traffic flow. You determine exactly which connections to use to forward traffic to each destination. In a given VLAN, you can use multiple IPv6 addresses to add multiple static routes in the VLAN. Other advantages include:

- Efficiency in a small network with few paths to manage
- Ease of configuration and maintenance
- Lower processor utilization

## Disadvantages of static routing

Network management and monitoring applications can detect failed static routes.

## Static route types

You can configure these types of static IPv6 routes:

**Standard**

The static route consists of

- Destination network prefix
- Link-local IPv6 address and VLAN ID of the (next-hop router) gateway IPv6 address

**Interface-based**

The static route consists of:

- Destination network address or host and a corresponding network prefix
- VLAN interface through which you want the routing switch to send traffic for the route

**Null (discard)**

Null routes include the following:

**Default**

When IPv6 routing is enabled, a route for the ::1/128 network is created and traffic to this network is rejected (dropped). The loopback address (lo0) is entered as the gateway. This route is for all traffic to the "loopback" network, with the single exception of traffic to the host address of the switch's loopback interface.

**Configured**

Provides a route that is used as a backup route for discarding traffic where the primary route is unavailable. A configured null route consists of:

- Destination network address or host and a corresponding network mask
- Either the `reject` keyword (traffic dropped with ICMP notification to the sender) or `blackhole` keyword (traffic dropped without any ICMP notification).

Nondefault null routes created with the `reject` or `blackhole` keywords use a gateway of zero (**0**).

**Figure 15: Example of static routes in an ECMP application** on page 159 illustrates the default and configured null route entries in the switch's routing table.

## Static routing default settings

The routing switch applies default administrative distance and metric values to ensure that static routes are preferred over dynamic routes to the same destination.

**Administrative distance**

If static routes, this is the value the routing switch uses to compare a static route to routes from other route sources to the same destination before placing a route in the routing table. The default administrative distance for static routes is 1, but can be configured to any value in the range of 1 to 255.

**Metric**

If static routes, this is the value the routing switch uses when comparing a static route to routes in the routing table from any dynamic routes to the same destination. The metric for static routes is fixed, that is, always set to "1".

## Static route states follow VLAN states

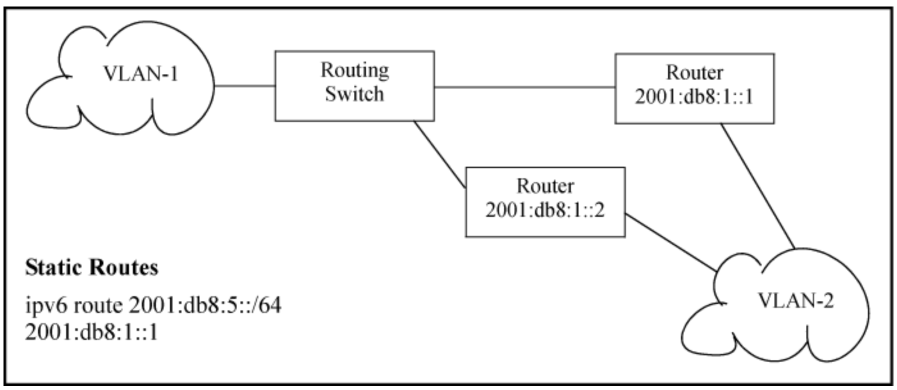
Static routes remain in the routing table only while the interface link to the next-hop router is up. If the next-hop router interface link goes down, the software removes the static route from the routing table. If the next-hop interface comes up again, the software adds the route back to the routing table.

This feature allows the routing switch to adjust to changes in network topology. The routing switch does not continue trying to use routes on unreachable paths, but instead uses routes only when their paths are reachable.

## Static routes for ECMP applications

Equal-cost multipath routing (ECMP) is a routing strategy where next-hop packet forwarding to a single destination can occur over multiple "best paths." Each path has the same cost as the other paths, but a different next-hop router. In static routing, load-balancing can be achieved through ECMP. The following figure illustrates static routes applied to an ECMP topology.

**Figure 15:** *Example of static routes in an ECMP application*




# BFD static routing

NOTE: BFD static routing is available only on switches running KB software. BFD over IPv6 static routes is not supported.

Bidirectional Forwarding Detection (BFD) provides short-duration detection of failures in the path between adjacent forwarding devices. You can now associate BFD with static routes to monitor the reachability of the next-hop gateway. When BFD is configured over a static route, it monitors the connectivity of the local router with the next hop IP. BFD must be configured on the local router interface as well as the next-hop router interface.

If a failure occurs, the corresponding BFD session is taken down and the corresponding static route entry from the Routing Information Base (RIB) is removed. If an alternate route to the destination exists, it is automatically added to the RIB. When the BFD session goes down, the session is not deleted, but the reachability of the next hop is attempted periodically by sending periodic BFD control packets. Once the next hop is reachable, BFD changes the session state to UP and installs the corresponding route in the RIB.

If the static route already exits, BFD can be configured on top of it. Nothing changes. If the static route does not exist, it can be configured along with the BFD configuration.

The `ip route bfd` command helps to enable BFD under ip static route for a particular next-hop destination and BFD source IP.

When BFD is configured over a static route, BFD starts monitoring the connectivity of the local router with the next hop ip. Static routes are added to or removed from the Routing Information Base (RIB) based on the status of BFD sessions. To successfully establish a BFD session, BFD should be configured on the local router interface as well as the next-hop router interface. BFD starts monitoring a given static route once the corresponding BFD session reaches the UP state.

Connectivity to the next hop router may be lost due to an event like an interface down or a neighbor going down; in such a scenario, BFD can detect such failures and trigger corrective measures to reduce network outages. When BFD detects a link failure (that is, BFD does not receive a control or echo packet for a specified amount of time), it takes the corresponding BFD session down and removes the corresponding static route entry from the RIB. If an alternate route to destination exists, it is automatically added to the RIB.

When the BFD session goes down, the session is not deleted. Once the next hop is reachable, BFD changes the session state to UP and installs the corresponding route in the RIB.

The BFD session is maintained in the BFD session database until it is explicitly removed by the user. Static route BFD is not supported for monitoring multi-hop connectivity.

Beginning with software release K.15.01, the routing switches support IPv6 RA configuration and transmission based on RFC 4861, "Neighbor Discovery for IP Version 6 (IPv6)" and RFC 4862, "IPv6 Stateless Address Autoconfiguration."

IPv6 RAs on a VLAN provide the ND policy the system administrator has configured for devices running in IPv6 host mode with address autoconfiguration enabled. RAs also enable hosts on a VLAN to build a list of default (reachable) routers on that VLAN or tunnel.

# Global configuration context commands

This sections explains the commands used for IPv6 router advertisement.

## ipv6 nd suppress-ra

**Syntax**

`ipv6 nd suppress-ra`

`no ipv6 nd suppress-ra`

**Description**

Global config command to suppress transmission of IPv6 Router Advertisements on all VLANs or tunnels configured on the routing switch. Overrides RAs enabled per-VLAN.

The `no` form of the command globally disables Router Advertisement suppression. Globally enabling Router Advertisements on the routing switch does not override per-VLAN or per-tunnel IPv6 Router Advertisement suppression (using the `ipv6 nd ra suppress` command in a **VLAN** or **tunnel** context).

Default: RA suppression disabled

## ipv6 unicast-routing

**Syntax**

`ipv6 unicast-routing`

`no ipv6 unicast-routing`

**Description**

Global config command to enable or disable IPv6 routing. Must be enabled for routing operation. Enabling IPv6 routing activates RA generation on VLANs or tunnels unless RAs are suppressed globally or per-VLAN or per-tunnel.

**Usage**

The `no` form of the command disables IPv6 routing and RAs on the routing switch.

Default: Disabled

# VLAN or tunnel context ND configuration

This sections explains the commands used for VLAN or tunnel context ND configuration.

## ipv6 nd ra managed-config-flag

**Syntax**

`ipv6 nd ra managed-config-flag`

`no ipv6 nd ra managed-config-flag`

**Description**

Controls the M-bit setting in RAs the router transmits on the current VLAN. Enabling the M-bit directs clients to acquire their IPv6 addressing and ND host configuration information for the current VLAN or tunnel interface from a DHCPv6 server.

**Usage**

- When the M-bit is enabled, receiving hosts ignore the `other-config-flag` (O-bit) setting described below.
- When the M-bit is disabled (the default), receiving hosts expect to receive their IPv6 addressing and ND configuration settings from the RA unless the O-bit is enabled.

The `no` form of the command turns off (disables) the setting for that command in RAs.

## ipv6 nd ra other-config-flag

**Syntax**

`ipv6 nd ra other-config-flag`

`no ipv6 nd ra other-config-flag`

**Description**

Ignored unless the M-bit (above) is disabled in RAs. Controls the O-bit in RAs the router transmits on the current VLAN or tunnel.

The `no` form of the command turns off (disables) the setting for that command in RAs.

**Usage**

Enabling the O-bit while the M-bit is disabled directs hosts on the VLAN to acquire their ND configuration settings from a DHCPv6 server and their global unicast prefixes from the RA.

NOTE

In the default configuration, both the M-bit and the O-bit are disabled, and a host receiving the RA must acquire its prefix and ND configuration from the RA itself and not from a DHCPv6 server.

Default for both settings: Disabled

# Commands to configure the range for intervals between RA transmissions on a VLAN

The interval between RA transmissions on a VLAN is a random value that changes every time an RA is sent. The interval is calculated to be a value between the current `max-interval` and `min-interval` settings described below.

## ipv6 nd ra max-interval

**Syntax**

`ipv6 nd ra max-interval` *`4 - 1800`*

`no ipv6 nd ra max-interval` *`4 - 1800`*

**Description**

This command displays the interval between RA transmissions on a VLAN.

The `no` form of the `max-interval` command returns the setting to its default, provided the default value is less than or equal to 75% of the new maximum interval you are setting.

**Specifiers**

Default: 600 seconds; Range: *4 - 1800* seconds

**Options**

**`max-interval`**

Must be equal to or less than the configured lifetime setting. Attempting to set `max-interval` to a value greater than the configured lifetime setting results in an error message.

**Usage**

Attempting to set `max-interval` to a value that is not sufficiently larger than the current `min-interval` also results in an error message.

Default: 600 seconds; Range: *4 - 1800* seconds

## ipv6 nd ra min-interval

**Syntax**

`ipv6 nd ra min-interval` *`3- 1350`*

`no ipv6 nd ra min-interval` *`3- 1350`*

**Description**

This command sets the minimum interval between RA transmissions on a VLAN.

The `no` form of the `min-interval` command returns the setting to its default, provided the default value is less than or equal to 75% of the new maximum interval you are setting.

**Options**

**`min-interval`**

Must be less than or equal to 75% of `max-interval`. Attempting to set `min-interval` to a higher value results in an error message.

**Usage**

Default: 200 seconds; Range: *3 - 1350* seconds

## ipv6 nd ra hop-limit

**Syntax**

`ipv6 nd ra hop-limit` *`0 - 255`*

`no ipv6 nd ra hop-limit` *`0 - 255`*

**Description**

**VLAN** context command to specify the hop-limit a host includes in the packets it transmits.

The `no` form of the command resets the hop-limit to zero (unspecified), which eliminates the hop-limit from the RAs originating on the VLAN or tunnel.

**Usage**

A setting of **0** means the hop-limit is unspecified in the RAs originating on the current VLAN. In this case, the hop-limit is determined by the host.

Default: 64; Range: unspecified *0 - 255*

## ipv6 nd ra lifetime

**Syntax**

`ipv6 nd ra lifetime 0 - 9000`

**Description**

**VLAN** or **tunnel** context command for configuring the lifetime in seconds for the routing switch to be used as a default router by hosts on the current VLAN. This setting must be configured to a value greater than or equal to the `max-interval` setting. A given host on a VLAN or tunnel refreshes the default router lifetime for a specific router each time the host receives an RA from that router. A specific router ceases to be a default router candidate for a given host if the default router lifetime expires before the host is updated with a new RA from the router.

**Usage**

A setting of **0** (unspecified) for default router lifetime in an RA indicates that the routing switch is not a default router on the subject VLAN or tunnel.

Default: Three times the `ra max-interval` setting. Range: unspecified *0 - 9000* seconds

# ipv6 nd ra reachable-time

**Syntax**

`ipv6 nd ra reachable-time [1000 - 3600000|unspecified]`

**Description**

**VLAN** or **tunnel** context command for all hosts on the VLAN or tunnel to configure as the reachable time duration for a given neighbor after receiving a reachability confirmation from the neighbor. This value is used to ensure a uniform reachable time among hosts on the VLAN or tunnel by replacing the individually configured settings on various hosts on the VLAN.

**Options**

***1000 - 3600000***

Reachable time in milliseconds.

**unspecified**

Configures the reachable time to zero, which disables the reachable-time setting in RAs on the current VLAN.

NOTE

If multiple routers on the same VLAN or tunnel are configured to advertise a reachable time, all such routers should use the same reachable-time setting.

# ipv6 nd ra ns-interval

**Syntax**

`ipv6 nd ra ns-interval [1000 - 4294967295|unspecified]`

**Description**

Used on VLAN or tunnel interfaces to advertise the period (retransmit timer) in milliseconds between ND solicitations sent by a host for an unresolved destination, or between DAD neighbor solicitation requests. Increasing this setting is indicated where neighbor solicitation retries or failures are occurring, or in a "slow" (WAN) network.

**Options**

***1000 - 4294967295***

An advertised setting in this range replaces the corresponding, locally configured setting in hosts on the VLAN.

**unspecified**

Sets the retransmit timer value in RAs to zero, which causes the hosts on the VLAN or tunnel to use their own locally configured NS-interval settings instead of using the value received in the RAs.

**Specifiers**

Default: unspecified (0) ; Range: *1000 - 4294967295* ms

**Usage**

The `no` form returns the setting to its default.

NOTE

This is the retransmit timer advertised as a host-specific variable. It is separate from the retransmit timer used by the routing switch for its own ND solicitations (`ipv6 nd ns-interval`).

If multiple routers on the same VLAN or tunnel are configured to advertise an `ns-interval` (retransmit time), all such routers should use the same NS-interval setting.

The default NS-interval setting for IPv6 host operation on E devices is 1000 ms. When the above command is used with the unspecified option to configure RAs, host devices configured by using the RA maintain their preconfigured NS-interval settings.

## Commands to configure global unicast prefix and lifetime for hosts on a VLAN

These commands define the content of RAs transmitted on a VLAN or tunnel.

### ipv6 nd ra prefix

**Syntax**

ipv6 nd ra prefix *ipv6-prefix/prefix-len*

no ipv6 nd ra prefix *ipv6-prefix/prefix-len*

[*valid-lifetime preferred-lifetime*|at|*valid-date preferred-date*|infinite|at|no-advertise]

[no-autoconfig]

[off-link]

**Syntax**

ipv6 nd ra prefix default

no ipv6 nd ra prefix default

[*valid-lifetime preferred-lifetime*|at|*valid-date preferred-date*|infinite|at|no-advertise]

[no-autoconfig]

[off-link]

**Options**

Options for *valid-lifetime preferred-lifetime* :

**Time in seconds:**

`[0 - 4294967295|0 - 4294967295]`

**Specific date and time**

*valid-lifetime preferred-lifetime*

`valid-lifetime-MM/DD/YY`

`valid-lifetime-HH:MM:SS`

`preferred-lifetime-MM/DD/YY`

`preferred-lifetime-HH:MM:SS`

`at valid-date preferred-date`

`valid-date - MM/DD/[YY]YY]`

`valid-date - HH:MM[:SS}`

`preferred-date - MM/DD/[YY]YY`

`preferred-date - HH:MM[:SS}`

**VLAN** or **tunnel** context command for specifying prefixes for the routing switch to include in RAs transmitted on the VLAN or tunnel. IPv6 hosts use the prefixes in RAs to autoconfigure themselves with global unicast addresses. A host's autoconfigured address is composed of the advertised prefix and the interface identifier in the host's current link-local address.

***valid-lifetime***

The total time the prefix remains available before becoming unusable. After preferred-lifetime expiration, any autoconfigured address is deprecated and used only for transactions that began before the preferred-lifetime expired. If the valid lifetime also expires, the address becomes unusable.

Default: *2,592,000 seconds - 30 days*; Range: *0 - 4294967295* seconds

***preferred-lifetime***

The span of time during which the address can be freely used as a source and destination for traffic. This setting must be less than or equal to the corresponding `valid-lifetime` setting.

Default: *604,000 second – 7 days*; Range: *0 - 4294967295* seconds

NOTE: The valid and preferred lifetimes designated in this command are fixed values. Each successive transmission of the same RA contains the same valid and preferred lifetimes.

**default**

Applied to all on-link prefixes that are not individually set by the`ipv6 ra prefix` *`ipv6-prefix/prefix-len`* command. It applies the same valid and preferred lifetimes, link state, autoconfiguration state, and advertise options to the advertisements sent for all on-link prefixes that are not individually configured with a unique lifetime. This also applies to the prefixes for any global unicast addresses configured later on the same VLAN or tunnel.

Using `default` once, and then using it again with any new values results in the new values replacing the former values in advertisements.

If `default` is used without the `no-advertise`, `no-autoconfig`, or the `off-link` keyword, the advertisement setting for the absent keyword is returned to its default setting.

NOTE: To configure a prefix as `off-link` or `no-autoconfig` , you must enter unique valid and preferred lifetimes with the `prefix` command (instead of the `default` command).

***ipv6-prefix / prefix-len***

Specifies the prefixes to advertise on the subject VLAN or tunnel. A separate instance of the command must be used for each prefix to advertise.

**infinite**

Specifies that the prefix lifetime will not expire. This option sets the valid and preferred lifetimes to infinity. (All bits set to 1; ffffffff.)

**no-advertise**

Specifies no advertisement for the prefix.

For example, if the routing-switches VLAN or tunnel interface is configured with any prefixes that you do not want advertised on the VLAN, use this command to specify the prefixes to withhold from advertisements on the subject VLAN or tunnel.

Default: Advertising enabled.

**no-autoconfig**

Disables host autoconfiguration by turning off the A-bit in RAs. This requires hosts to acquire prefixes through manual or DHCPv6 assignments. Depending on the host implementation, a host that was previously configured by an RA to use autoconfiguration will not be affected by a later RA that includes `no-autoconfig` (unless the host disconnects and reconnects to the network). To re-enable host autoconfiguration (turn on the A-bit in RAs) for a given RA, use `ipv6 nd ra prefix` again, without invoking `no-autoconfig`.

Default: A-bit turned on— host autoconfig turned on.

**off-link**

Sets the (L-bit) prefix information in an RA to indicate that the advertised prefix is not on the subject VLAN. A host that was previously configured using an RA without `off-link` will not be affected by a later RA that includes `off-link` (unless the host disconnects and reconnects to the network).

Can be used in instances where the prefix is being deprecated, and you do not want any newly brought up hosts to use the prefix.

Default: L-bit turned off.

The `no` form of the command deletes the specified prefix from RAs.

### Using the default command to configure prefix advertisement content

The table lists the global unicast addresses configured on a VLAN, with original and updated settings configured using the `default` command.

**Table 18:** *Example using the default command to configure prefix advertisement content*

| Address or prefix | Interface | Original lifetime & autoconfig | Updated lifetime & autoconfig | Advertise on VLAN 100? |
|---|---|---|---|---|
| 2001:db8:0:f::f1/64 | VLAN 100 | 15 days<br>14 days<br>Auto: Yes | 30 days<br>25 days<br>Auto: No | Yes |
| 2001:db8:0:b::b1/64 | VLAN 100 | | | |
| 2001:db8:0:c::c1/64 | VLAN 100 | | | |

*Table Continued*

| Address or prefix | Interface | Original lifetime & autoconfig | Updated lifetime & autoconfig | Advertise on VLAN 100? |
|---|---|---|---|---|
| 2001:db8:0:d::d1/64 | VLAN 100 | | | |
| 2001:db8:0:a::/64 | Off-Link | 12/31/2010at 00:00:01<br>12/20/2010at 00:00:01<br>Auto: Yes | not updated | |

**Example output**

Using the default command to configure and update prefix advertisements

```
Switch(config)# vlan 100
Switch(vlan-100)# ipv6 address 2001:db8:0:f::f1/64
Switch(vlan-100)# ipv6 address 2001:db8:0:b::b1/64 [1]
Switch(vlan-100)# ipv6 address 2001:db8:0:c::c1/64
Switch(vlan-100)# ipv6 nd ra prefix default 1296000 1209600 [2]
Switch(vlan-100)# show ipv6 nd ra prefix vlan 100

 IPv6 Neighbor Discovery Prefix Information

 VLAN Name : VLAN100 [3]

   IPv6 Prefix        : Default
   Valid Lifetime     : 15 days
   Preferred Lifetime : 14 days
   On-link Flag       : On
   Autonomous Flag    : On
   Advertise Flag     : On

Switch(vlan-100)# ipv6 address 2001:db8:0:d::d1/64 [4]
Switch(vlan-100)# ipv6 nd ra prefix 2001:db8:0:d::/64 infinite no-autoconfig
Switch(vlan-100)# ipv6 nd ra prefix 2001:db8:0:a::/64 at 12/31/2010 00:00:01
12/20/2010 00:00:01
                    off-link [5]
Switch(vlan-100)# show ipv6 nd ra prefix vlan 100

 IPv6 Neighbor Discovery Prefix Information

 VLAN Name : VLAN100

   IPv6 Prefix        : Default [6]
   Valid Lifetime     : 15 days
   Preferred Lifetime : 14 days
   On-link Flag       : On
   Autonomous Flag    : On
   Advertise Flag     : On

   IPv6 Prefix        : 2001:db8:0:a::/64 [7]
   Valid Lifetime     : 12/31/2010 00:00:01
   Preferred Lifetime : 12/20/2010 00:00:01
   On-link Flag       : Off
   Autonomous Flag    : On
   Advertise Flag     : On

   IPv6 Prefix        : 2001:db8:0:d::/64 [8]
   Valid Lifetime     : Infinite
   Preferred Lifetime : Infinite
   On-link Flag       : On
   Autonomous Flag    : Off
   Advertise Flag     : On
```

```
Switch(vlan-100)# ipv6 nd ra prefix default 2592000 2160000 no-autoconfig 9
Switch(vlan-100)# show ipv6 nd ra prefix vlan 100

 IPv6 Neighbor Discovery Prefix Information

 VLAN Name : VLAN100

   IPv6 Prefix        : Default 10
   Valid Lifetime     : 30 days
   Preferred Lifetime : 25 days
   On-link Flag       : On
   Autonomous Flag    : Off
   Advertise Flag     : On

   IPv6 Prefix        : 2001:db8:0:a::/64 11
   Valid Lifetime     : 12/31/2010 00:00:01
   Preferred Lifetime : 12/20/2010 00:00:01
   On-link Flag       : Off
   Autonomous Flag    : On
   Advertise Flag     : On

   IPv6 Prefix        : 2001:db8:0:d::/64 12
   Valid Lifetime     : Infinite
   Preferred Lifetime : Infinite
   On-link Flag       : On
   Autonomous Flag    : Off
   Advertise Flag     : On
```

[1] Global unicast addresses configured on VLAN 100

[2] To enable advertising prefixes of global unicast addresses configured on the VLAN: The default command sets default lifetime, prefix link status (on or off-link), autoconfiguration (Autonomous Flag) status (on or off), and advertisement setting (on or off).

[3] Show command displays default prefix mode settings for global unicast addresses configured on VLAN 100.

[4] New global unicast address configured on the VLAN. Followed by command to assign unique lifetime and autoconfig setting in the advertisements for this prefix. Link flag and Advertise flag omitted from the command and therefore set to “On” by default.

[5] Off-link prefix designated with unique lifetime. Autoconfig (Autonomous) flag and Advertise flag omitted from the command and therefore set to “On”.

[6] Show command displays default advertisement settings for prefixes of global unicast addresses configured on VLAN 100.

[7] Show command displays unique advertisement settings for 2001:db8:0:a::/64 also configured on VLAN 100.

[8] Show command displays unique advertisement settings for 2001:db8:0:d::/64 identified as an off-link prefix.

[9] For prefixes configured on the VLAN and not specifically addressed by a prefix command, default changes the default lifetime and the autoconfig setting in advertisements for these prefixes. On-Link flag and Advertise flag omitted from the command and therefore set to “On” by default.

[10] Show command displays changes in default prefix mode settings for global unicast addresses configured on VLAN 100.

[11] No change for the on-link prefix specifically configured by a prefix command, and the off-link prefix that is also configured for advertisement on the VLAN.

[12] No change for the on-link prefix specifically configured by a prefix command, and the off-link prefix that is also configured for advertisement on the VLAN.

## valid-lifetime preferred-lifetime

**Syntax**

`[`*`valid-lifetime preferred-lifetime`*`|at|`*`valid-date preferred-date`*`|infinite|at|no-advertise]`

**Description**

**VLAN** or **tunnel** context command for specifying prefixes for the routing switch to include in RAs transmitted on the VLAN or tunnel. IPv6 hosts use the prefixes in RAs to autoconfigure themselves with global unicast addresses. A host's autoconfigured address is composed of the advertised prefix and the interface identifier in the host's current link-local address.

**Options**

***`valid-lifetime`***

The total time the prefix remains available before becoming unusable. After preferred-lifetime expiration, any autoconfigured address is deprecated and used only for transactions that began before the preferred-lifetime expired. If the valid lifetime also expires, the address becomes unusable.

Default: *2,592,000 seconds - 30 days*; Range: *0 - 4294967295* seconds

***`preferred-lifetime`***

The span of time during which the address can be freely used as a source and destination for traffic. This setting must be less than or equal to the corresponding `valid-lifetime` setting.

Default: *604,000 second – 7 days*; Range: *0 - 4294967295* seconds

**`default`**

Applied to all on-link prefixes that are not individually set by the`ipv6 ra prefix` *`ipv6-prefix/prefix-len`* command. It applies the same valid and preferred lifetimes, link state, autoconfiguration state, and advertise options to the advertisements sent for all on-link prefixes that are not individually configured with a unique lifetime. This also applies to the prefixes for any global unicast addresses configured later on the same VLAN or tunnel.

Using `default` once, and then using it again with any new values results in the new values replacing the former values in advertisements.

If `default` is used without the `no-advertise`, `no-autoconfig`, or the `off-link` keyword, the advertisement setting for the absent keyword is returned to its default setting.

NOTE

To configure a prefix as `off-link` or `no-autoconfig` , you must enter unique valid and preferred lifetimes with the `prefix` command (instead of the `default` command).

**`infinite`**

Specifies that the prefix lifetime will not expire. This option sets the valid and preferred lifetimes to infinity. (All bits set to 1; ffffffff.)

**`no-advertise`**

Specifies no advertisement for the prefix.

For example, if the routing-switches VLAN or tunnel interface is configured with any prefixes that you do not want advertised on the VLAN, use this command to specify the prefixes to withhold from advertisements on the subject VLAN or tunnel.

Default: Advertising enabled.

## no-autoconfig

**Syntax**

`[no-autoconfig]`

**Description**

Disables host autoconfiguration by turning off the A-bit in RAs. This requires hosts to acquire prefixes through manual or DHCPv6 assignments. Depending on the host implementation, a host that was previously configured by an RA to use autoconfiguration will not be affected by a later RA that includes `no-autoconfig` (unless the host disconnects and reconnects to the network).

**Usage**

To re-enable host autoconfiguration (turn on the A-bit in RAs) for a given RA, use `ipv6 nd ra prefix` again, without invoking `no-autoconfig`.

Default: A-bit turned on— host autoconfig turned on.

### off-link

**Syntax**

`[off-link]`

**Description**

Sets the (L-bit) prefix information in an RA to indicate that the advertised prefix is not on the subject VLAN. A host that was previously configured using an RA without `off-link` will not be affected by a later RA that includes `off-link` (unless the host disconnects and reconnects to the network).

**Usage**

Can be used in instances where the prefix is being deprecated, and you do not want any newly brought up hosts to use the prefix.

Default: L-bit turned off.

## ipv6 nd ra suppress

**Syntax**

`ipv6 nd ra suppress`

`no ipv6 nd ra suppress`

**Description**

**VLAN** or **tunnel** context command to turn off (disable) transmission of RAs from the routing switch on the VLAN.

**Usage**

The `no` form of the command turns on (enables) router advertisement transmission from the routing switch on the current VLAN or tunnel.

Default: Suppression disable, that is, router advertisement enabled on the VLAN or tunnel.

## IPv6 Router Advertisements restrictions

The RA Guard feature restricts the ports (or trunks) that can accept IPv6 Router Advertisements (RAs). Additionally, ICMPv6 router redirects are blocked on the configured ports.

Only physical ports and trunk ports are supported. Dynamic ports, dynamic trunks, and mesh ports are not supported.

NOTE

IPv6 RAs are ICMPv6 type 134 messages and may be sent to either the “all nodes” multicast address (FF02:0:0:0:0:0:0:1) or to the address of the device itself as a result of an IPv6 router solicitation. IPv6 router redirect messages are ICMPv6 type 137 messages. They are sent to the source address of the packet that triggered the redirect.

## ipv6 ra-guard ports

### Syntax

```
ipv6 ra-guard ports <port-list> log
```

```
no ipv6 ra-guard ports <port-list> log
```

### Description

Enables or disables RA Guard on the specified ports, which blocks IPv6 router advertisements and router redirects.

The no form of the command disables RA Guard.

### Options

log Enables debug logging of RA and redirects packets to debug output.

### Usage

```
Switch(config)# ipv6 ra-guard ports 6 log
```

## Router advertisement operations

- When a logical trunk port is enabled, all members of the trunk are enabled for RA Guard. Likewise, when a logical trunk port is disabled (no ipv6 ra-guard ports *<trunk-port>* ), all members of the trunk are disabled for RA.
- When ports are configured for RA Guard, hardware resources are allocated. If there are not enough hardware resources, this message displays:Commit failed
- When debug logging is enabled (ipv6 ra-guard ports <port-list> log), the RA and redirect packets are sent to the processor, which can be CPU-intensive. This message displays:The log option uses a lot of CPU and should be used only for short periods of time.
- The debug security ra-guardcommand is used to filter and display RA Guard debug log messages.

To display configuration and statistical information about RA Guard, enter the show ipv6 ra-guard command.

**Figure 16:** *Output showing configuration and Statistics for RA Guard*

```
Switch(config)# show ipv6 ra-guard

IPv6 RA Guard Information

 Port  Block RAs Blocked Redirs Blocked Log
 ----- ----- ----------- -------------- ---
 1     No    0           0              No
 2     No    0           0              No
 3     No    0           0              No
 4     No    0           0              No
 5     No    0           0              No
 6     Yes   123         450            Yes
 7     No    0           0              No
 8     No    0           0              No
```

When RA Guard is enabled, there will be one or two lines displayed in the running config file.

**Figure 17:** *Running Config File Showing Line for RA-Guard*

```
Switch(config)# show running-config

Running configuration:

; J8693A Configuration Editor; Created on release #K.15.07.0000x
; Ver #02:01.0f:0c

hostname "HP Switch"
module 1 type J86yyA
module 2 type J86xxA
module 3 type J8694A
no stack auto-join
vlan 1
   name "DEFAULT_VLAN"
   untagged 1-4,7-48,A1-A4
   ipv6 address fe80::2 link-local
   ip address dhcp-bootp
   ipv6 enable
   no untagged 5-6
   exit
vlan 2
   name "VLAN2"
   untagged 5-6
   ip address 10.10.10.1 255.255.255.0
   exit
power-over-ethernet pre-std-detect
sflow 3 destination 3fff::3
ipv6 unicast-routing
ipv6 ra-guard ports 6 log
...
```

RA Guard is enabled on port 6; logging is enabled.

# show ipv6 nd ra

### Syntax

`show ipv6 nd ra`

`show ipv6 nd ra prefixvlan` *`vid`*

### Description

Without the optional keywords, this command displays the global and per-VLAN, and per tunnel router advertisement neighbor discovery configuration on a specific routing switch. This indicates the per-VLAN or per-tunnel content of RAs transmitted from the routing switch.

### Options

**`prefix`**

Displays the prefixes, valid lifetime, and onlink/auto values advertised by the routing switch on all VLANs or tunnels configured for RA operation.

**`prefix vlan`** ***`vid`***

Displays values for each prefix configured using `ipv6 nd ra prefix` on the specified VLAN or tunnel.

**`IPv6 Prefix`**

Displays values for specific prefixes configured for RAs on a VLAN or tunnel by the `ipv6 nd ra prefix` command, plus `Default` (to apply to any global unicast prefixes on the same VLAN(s) or tunnel(s) that have not been configured by `ipv6 nd ra prefix`).

**`Valid Lifetime`**

The valid lifetime configured for the indicated prefix.

**`Preferred Lifetime`**

The preferred lifetime configured for the indicated prefix.

**`On-link Flag`**

Indicates whether the prefix is advertised as on-link.

Default: On; On-link enabled.

**`Autonomous Flag`**

Indicates whether address autoconfiguration is turned on.

Default: On; Autoconfiguration enabled.

**`Advertise Flag`**

Indicates whether advertisement for the subject prefix is turned on.

Default: On.

**Example output**

General Output Listing the RA Configuration on a Routing Switch

```
Switch(config)# show ipv6 nd ra

IPv6 Router Advertisement Configuration

Global RA Suppress    : No
Global Hop Limit      : 10
IPv6 Unicast Routing : Enabled

Interface Supp Interval Lifetime Mngd Other RCH Time NS Intrvl Hop
ID        RA   Min/Max  (sec)    Flag Flag  (ms)     (ms)     Limit
--------- ---- -------- -------- ---- ----- -------- -------- -----
vlan-1    Yes  200/600  1800     No   No    0        0        10
vlan-22   No   200/600  1800     No   No    0        0        10
tunnel-3  Yes  200/600  1000     No   No    0        0        4
```

Output where specific prefixes have been configured for RAs

```
Switch(tunnel-3)# ipv6 nd ra prefix default 1296000 1209600

Switch(config)# show ipv6 nd ra prefix

  IPv6 Neighbor Discovery Prefix Information

VLAN Name : VLAN22

  IPv6 Prefix                   Valid Lifetime      Onlink/Auto
  ----------------------------- ------------------- -----------
  Default                       Infinite            On/On
  2001:db8:0:a::/64             1h:20m:44s          Off/On

Tunnel Name : Tunnel3

  IPv6 Prefix                   Valid Lifetime      Onlink/Auto
  ----------------------------- ------------------- -----------
  Default                       15 days             On/On
```

Detailed prefix configuration data for a specific VLAN

```
Switch(config)# show ipv6 nd ra prefix vlan 30

 IPv6 Neighbor Discovery Prefix Information
```

```
VLAN Name : VLAN30

  IPv6 Prefix         : Default
  Valid Lifetime      : Infinite
  Preferred Lifetime  : Infinite
  On-link Flag        : On
  Autonomous Flag     : On
  Advertise Flag      : On

  IPv6 Prefix         : 2001:db8:f:1b::/64
  Valid Lifetime      : 11/31/2010 00:00:01
  Preferred Lifetime  : 11/01/2010 00:00:01
  On-link Flag        : Off
  Autonomous Flag     : On
  Advertise Flag      : On

  IPv6 Prefix         : 2001:db8:f:1d::/64
  Valid Lifetime      : 11/31/2010 00:00:01
  Preferred Lifetime  : 11/01/2010 00:00:01
  On-link Flag        : On
  Autonomous Flag     : On
  Advertise Flag      : On
```

# Router advertisement general operation

An IPv6 routing switch configured as a member of a given VLAN transmits RAs for use by hosts on the VLAN or tunnel. It also transmits unscheduled RAs in response to router solicitations received from IPv6 hosts on the VLAN. The values a host receives in an RA are applied to settings that have not already been configured on the host by the system operator. (Values in an RA can also replace host settings that were learned from a previous RA.)

When IPv6 unicast routing is enabled, RAs are transmitted by default on VLANs or tunnels enabled for IPv6 and configured with an IPv6 link-local address, unless RA transmission has been explicitly suppressed. RA configuration includes:

| Advertisement Value | Default | Page |
|---|---|---|
| managed flag (M-bit) | Not set | **VLAN or tunnel context ND configuration** on page 161 |
| other-config-flag (O-bit) | Not set | **VLAN or tunnel context ND configuration** on page 161 |

*Table Continued*

| Advertisement Value | Default | Page |
|---|---|---|
| prefix | The prefix of any global unicast IPv6 address configured on the VLAN interface[1] | **Commands to configure global unicast prefix and lifetime for hosts on a VLAN** on page 165 |
| length | N/A; based on existing configuration | — |
| valid lifetime | 2,592,000 seconds (30 days) | — |
| preferred lifetime | 604,800 seconds (7 days) | — |
| autoconfig (A-bit) | Set (host autoconfig enabled) | — |
| on-link (L-bit) | Set (use prefix on subject VLAN) | — |
| maximum | 600 seconds | **Commands to configure the range for intervals between RA transmissions on a VLAN** on page 162 |
| minimum | 200 seconds | **Commands to configure the range for intervals between RA transmissions on a VLAN** on page 162 |
| current hop limit | 64 | **ipv6 nd ra hop-limit** on page 163 |
| default lifetime | 1800 seconds (3 x max. transmission interval) | **ipv6 nd ra lifetime** on page 164 |
| reachable time | Unspecified (0) | **ipv6 nd ra reachable-time** on page 164 |
| retransmission timer | Unspecified (0) | **ipv6 nd ra ns-interval** on page 164 |

[1] Default operation excludes prefixes of stateless autoconfigured addresses.

- Enabling IPv6 unicast routing on a routing switch initiates transmission of RAs on active, IPv6-enabled VLANs unless RA transmission has been suppressed.
- RAs are not routed.
- A host response to an RA depends on how the host implements IPv6. Generally, settings in an RA received by a host replaces settings received from an earlier RA. Settings configured directly on a host by an operator may override values received in an RA for the same settings.
- When a host receives a default "unspecified" value in an RA, the host applies either its own current setting for that value, or the defaults specified in RFC 4861 or other applicable RFCs, depending on how IPv6 is implemented in the host.
- The M-bit and O-bit flags enable RAs to be configured either to act as the sole source of host addressing and related settings, or to direct the host to use a DHCPv6 server for some or all such settings.

## Setting IPv6 router advertisement policies

- Is there a role for a DHCPv6 server in host configuration on a given VLAN, and what host services and policy will be configured? Affects M-bit and O-bit options.
- What is the ND policy that should be advertised? Includes hop-limit for host-generated traffic, the default router period, neighbor reachable time, and retransmit time for neighbor solicitations.
- What prefixes should be advertised, and what prefixes should be suppressed? Prefixes configured on the routing switch VLAN interface will be included in RAs on that VLAN unless denied.
- What should be the maximum and minimum intervals (in seconds) for transmitting RAs?
- Are there any VLANs or tunnels on the routing switch where RAs should be suppressed?
- Will multiple routing devices be used to send RAs on a VLAN?
  - The first RA received by a host determines the default router for that host. Other routers included in subsequent RAs received by the host become backup default routers for that host.
  - What, if any, differences are acceptable in RAs from different routing devices?

# Configuring IPv6 router advertisement

When IPv6 unicast routing is enabled on the routing switch, RAs are transmitted on all IPv6-enabled VLANs or tunnels unless explicitly suppressed globally or per-VLAN. The following steps provide a general outline of the steps for configuring the routing switch for nondefault RA operation on all IPv6-enabled VLANs or tunnels:

**Procedure**

1. Enable IPv6 routing on your network.
2. Enable IPv6 unicast routing. (This must be enabled to allow configuration of other routing protocols).
   `Switch(config)# ipv6 unicast-routing`.
3. Configure the desired per-VLAN or per-tunnel RA operation:
   a. Use the M-bit and O-bit settings to specify the source for IPv6 host configuration; see page **VLAN or tunnel context ND configuration** on page 161:
      I. M-bit setting:
         - Get configuration from RAs (default).
         - Get configuration from DHCPv6.
      II. O-bit setting (applies only if M-bit setting is left in default state):
         - Use RA source for global unicast prefixes (default).
         - Do not use the RA for nonprefix configuration.
   b. Configure global unicast prefix assignments.

I. Specify any prefixes not configured on the routing switch VLAN or tunnel interface that should be transmitted in RAs to IPv6 hosts on the VLAN.

II. Deny any prefixes configured on the routing switch VLAN or tunnel interface that should not be transmitted in RAs to IPv6 hosts on the VLAN. Default: Global unicast prefixes configured on the routing switch VLAN interface are included in RAs.

**c.** Configure the maximum and minimum interval for transmitting router advertisements on the VLAN; see page **ipv6 nd ra prefix** on page 165.

NOTE

The routing switch also transmits RAs when it receives router solicitations from a host. Autoconfiguration must be enabled on the host before it will generate router solicitations on the VLAN or tunnel.

**d.** Configure the ND policy for hosts on the VLAN or tunnel to use:

I. Hop-limit; Default; 64, see page **ipv6 nd ra hop-limit** on page 163.

II. Default router lifetime; Default: 1800 seconds, see page **ipv6 nd ra prefix** on page 165.

III. Reachable time duration to advertise for confirmed neighbors; Default: unspecified (0); see page **ipv6 nd ra ns-interval** on page 164.

IV. Retransmit time to advertise for neighbor solicitations; Default: unspecified (0); see page **ipv6 nd ra ns-interval** on page 164.

**e.** Configure per-VLAN RA suppression for any VLAN or tunnel on which you do not want the routing switch to transmit RAs. See page **show ipv6 nd ra** on page 173.

```
Switch(vlan-1)# ipv6 nd ra suppress
Switch(tunnel-3)# ipv6 nd ra suppress
```

# About configuring router advertisements on multiple switches with a common VLAN

Multiple routing switches transmitting RAs on the same VLAN or tunnel can provide redundancy. Typically, a host identifies the first router from which it receives an RA as the default router. The host uses any RAs received later from other routers to identify backup default routers.

While advertised prefixes can be different, the per-VLAN or per-tunnel RA policy should be the same for all routers transmitting RAs on a given VLAN. This includes the following parameters:

- Managed-config-flag (M-bit)
- Other-config-flag (O-bit)
- Default router lifetime
- Hop-limit
- Reachable-time for neighbors
- Retransmit time for neighbor solicitations

For introductory information on DHCPv6-relay, see **About configuring DHCPv6–Relay** on page 182.

# Commands to enable and configure DHCPv6–relay

DHCPv6-relay is disabled by default. To enable and configure it, use the commands in this section.

## dhcpv6–relay

**Syntax**

`dhcpv6-relay`

`no dhcpv6-relay`

**Description**

Used in the **global config** context to enable DHCPv6-relay globally on the routing switch.

The `no` form disables DHCPv6-relay operation on the routing switch.

Default: Disabled

**Usage**

To use DHCPv6-relay on a given VLAN, at least one IPv6 helper-address must be configured on the VLAN, and IPv6 routing (`ipv6 unicast-routing`) must be enabled.

## ipv6 helper-address

**Syntax**

`ipv6 helper-address unicast` *`ipv6-unicast-helper-addr`*

`no ipv6 helper-address unicast` *`ipv6-unicast-helper-addr`*

`ipv6 helper-address multicast [all-dhcp-servers|`*`ipv6-unicast-helper-addr`*`][ egress vlan` *`vid`*`]`

`no ipv6 helper-address multicast [all-dhcp-servers|`*`ipv6-unicast-helper-addr`*`] [ egress vlan` *`vid`*`]`

**Description**

Used in the **VLAN** context to enable DHCPv6-relay operation on the VLAN, and to specify either a unicast or multicast DHCPv6-relay helper address for forwarding DHCPv6 service requests from hosts on the subject VLAN.

**Options**

***ipv6-unicast-helper-addr***

Specifies the global unicast address of a remote DHCPv6 server configured to support hosts on the indicated VLAN.

**all-dhcp-servers**

Specifies that the routing switch forward host requests for DHCPv6 service to multicast address FF05::1:3 via the VLAN specified by `egress vlan` *`vid`*

**egress vlan *vid***

Specifies the VLAN on which DHCPv6 service requests forwarded to a multicast destination will be relayed. The egress VLAN must be a different VLAN than the one on which the multicast helper address is configured.

A service request relayed on the egress VLAN to a downstream router remains in that VLAN unless the downstream router is configured on that VLAN with a unicast helper address for a server on another VLAN.

Using the `no`form of the command removes the specified helper address. Removing all helper addresses from a given VLAN disables DHCPv6-relay on that VLAN.

NOTE: DHCPv6-relay operation must be enabled with `dhcpv6-relay` at the global config level.

## show ipv6 helper-address

**Syntax**

`show ipv6 helper-address [ vlan vid ]`

**Description**

Displays the DHCPv6-relay configuration on all VLANs configured on the routing switch or on the VLAN you specify.

**Example output**

Display of Unicast Helper Address configured on VLAN 10 in **Figure 18: Routing a unicast DHCPv6 request across a multiple-hop topology** on page 183.

```
Switch(config)# show ipv6 helper-address

 VLAN: 10

 IPv6 Helper Address                      Egress Vlan
 ---------------------------------------- -----------
 2001:db8:0:12::11                        1
```

[1] Egress VLAN not used with unicast Helper addresses

**Example output**

Display of Multicast Helper Address configured on VLAN 10:

```
Switch(config)# show ipv6 helper-address

 VLAN: 14

 IPv6 Helper Address                      Egress Vlan
 ---------------------------------------- -----------
 FF05::1:3                                1
```

[1] Egress VLAN required for multicast Helper addresses

## show run for DHCPv6

**Syntax**

`show run`

**Description**

Use this option to verify whether DHCPv6-relay is enabled on the routing switch. The output includes per-VLAN listings of any configured helper addresses.

**Usage**

```
Switch(config)# show run

Running configuration:
 .
 .
 .
ipv6 hop-limit 25 1
ipv6 unicast-routing 2
interface loopback 1
   ip address 1.1.1.1
   exit
snmp-server community "public" unrestricted
vlan 10
   untagged 20-22
   ipv6 address fe80::1 link-local
   ipv6 address 2001:db8:0:10::1
   ipv6 helper-address unicast 2001:db8:0:12::11 3
   exit
vlan 14
   ipv6 address fe80::1 link-local
   ipv6 address 2001:db8:0:14::1
   exit

dhcpv6-relay 4
```

[1] Non-Default Hop-Limit configured

[2] IPv6 Unicast Routing Enabled

[3] DHCPv6 Helper Address configured Per-VLAN

[4] DHCPv6-Relay Globally Enabled

Use the `show dhcpv6-relay` command to display statistical information about DHCPv6 relay.

**Example output**

DHCPv6 relay information

```
Switch(config)# show dhcpv6-relay

 DHCPV6 Relay Agent : Enabled

 Client Requests Server Responses

 VLAN Name Received Dropped Failed Received Dropped Failed
 --------- -------- ------- ------ -------- ------- ------
 DEFAULT   120      0       0      120      0       0
 VLAN20    30       1       0      29       2       1
```

## DHCPv6-relay multicast operations

- A DHCPv6-relay message sent to any multicast address carries a hop limit of 32. The hop limit for DHCPv6 requests sent to a unicast address is determined by the `ipv6 hop-limit` *`1 - 255`* command at the global configuration level; Default: 64.
- Multicast addresses reserved for DHCPv6 include the following:
  - FF02::1:2—All_DHCP_Relay_Agents_and_Servers
  - FF05::1:3—All_DHCP_Servers
- DHCPv6 client and relay functions are mutually exclusive on a VLAN. Attempting to configure one of these functions on a VLAN while the other is already configured results in one of the following messages:

    - `Cannot configure an IPv6 helper-address as DHCPv6 client is enabled on the VLAN`
    - `Cannot enable DHCPv6 client as an IPv6 helper-address is configured on the VLAN.`
- The routing switch supports concurrent, independent operation of DHCPv4 and DHCPv6.
- Operating limits:

| DHCPv6-relay feature | Maximum |
|---|---|
| Unique helper addresses supported on the routing switch | 32 |
| Unique helper addresses per VLAN interface | 32[1] |

*1* If the same helper address is used on multiple VLANs, it is counted as one address toward these maximums.

# About configuring DHCPv6–Relay

Beginning with software release K.15.01, the routing switches include operation as DHCPv6 relay agents between DHCPv6 servers and clients.

Dynamic Host Configuration Protocol (DHCPv6) is used for configuring clients with IPv6 address and other configuration parameters without user intervention on the client. Where a DHCPv6 server and a client exist on the same local network, the client's requests for service are received directly by the DHCPv6 server, and a relay agent is not needed. However, if a client and the DHCPv6 server available to support it are in different networks, or subnets, a DHCPv6 relay agent is needed to forward client service requests to the server and to relay server responses back to the client. (The DHCPv6 relay agent is transparent to the client.)

Three main elements comprise DHCPv6-relay operation:

- DHCPv6 clients per-VLAN or subnet
- A routing switch configured with the following:
    - Either IPv6 static routing or the OSPFv3 routing protocol (or both)
    - DHCPv6 relay agent enabled to forward DHCPv6 client/server traffic between a host or another relay agent and a remote DHCPv6 server
- One or more remote DHCPv6 servers reachable from the routing switch

## DHCPv6–relay request forwarding

With DHCPv6-relay enabled and a reachable unicast helper address configured on a given VLAN, a client request for DHCPv6 service will be routed to the designated server. If a multicast helper address is configured on the VLAN, the client request will be sent from the routing switch on the VLAN designated in the configuration for that helper address.

## DHCPv6-relay helper addresses

A unicast helper address enables routing of a client service request to the IPv6 address of a specific, remote DHCPv6 server.

The multicast forwarding options route DHCPv6 requests on a VLAN interface to either

- The "All_DHCP_Servers" (FF05::1:3) multicast address
- A user-selected multicast address

The routing switch supports up to 32 unique helper addresses and counts multiple instances of the same helper address on different VLANs as one address. Where multiple helper addresses are configured on the same VLAN, the routing switch forwards client service requests to all such addresses, and selects the server from which it receives the first response.

## Enabling DHCPv6–relay operation

For the DHCPv6 relay agent to function on the routing switch, you must complete the following steps:

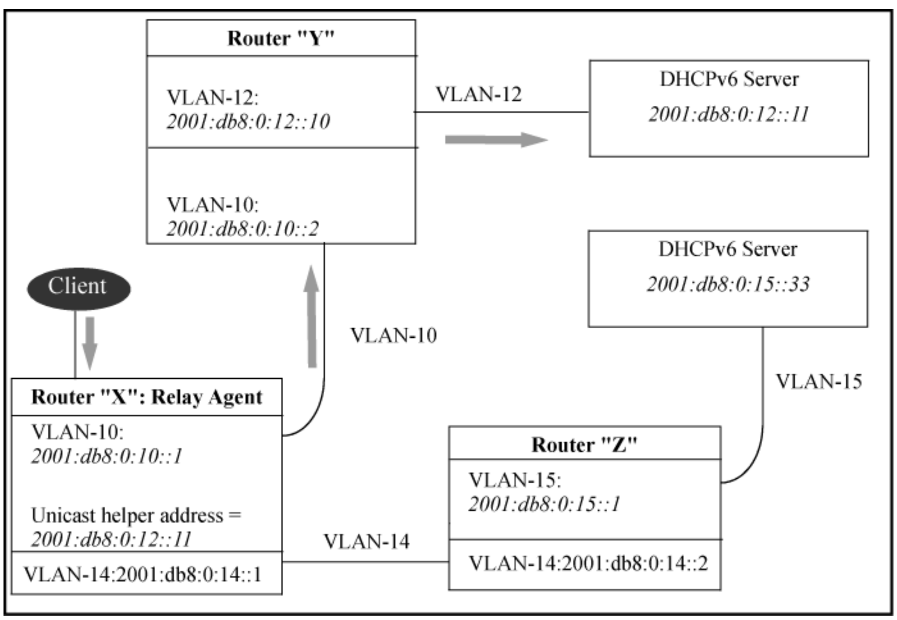
**Procedure**

1. Ensure that there is a route configured between a DHCPv6 server and the routing switch and that the server is configured to support host requests forwarded from the routing switch.
2. For each VLAN on which you want the routing switch to provide DHCPv6-relay services, determine the helper addresses the relay agent should have for forwarding client DHCPv6 requests to reachable DHCPv6 servers. You can configure one or more helper addresses and can use either or both of the following types:
3. In each **VLAN** context where DHCPv6-relay service is needed, use the `ipv6 helper-address` *`unicast / multicast`* command to configure one or more DHCPv6 helper addresses.
4. In the **global config** context, use the `dhcpv6-relay` command to globally enable DHCPv6 relay on the routing switch.
5. If IPv6 routing is not already enabled on the routing switch, use the `ipv6 unicast-routing` command in the **global config** context to enable IPv6 routing.
6. On each VLAN where you have configured a helper address, ensure that the target DHCPv6 server is reachable.

# Multiple-hop forwarding of DHCPv6 service requests

If a routing switch receives a unicast DHCPv6 service request forwarded by a relay agent, the request is routed to the specified server. For example, in the following figure, router "X" is a relay agent configured to forward DHCPv6 requests received from VLAN 10 to unicast helper address 2001:db8:0:12::11. Router "Y" receives the request from router "X" on VLAN 10 and routes it to the DHCPv6 server at 2001:db8:0:12::11 on VLAN 12. In this case, router "Y" acts as an IPv6 router and not as a DHCPv6 relay.

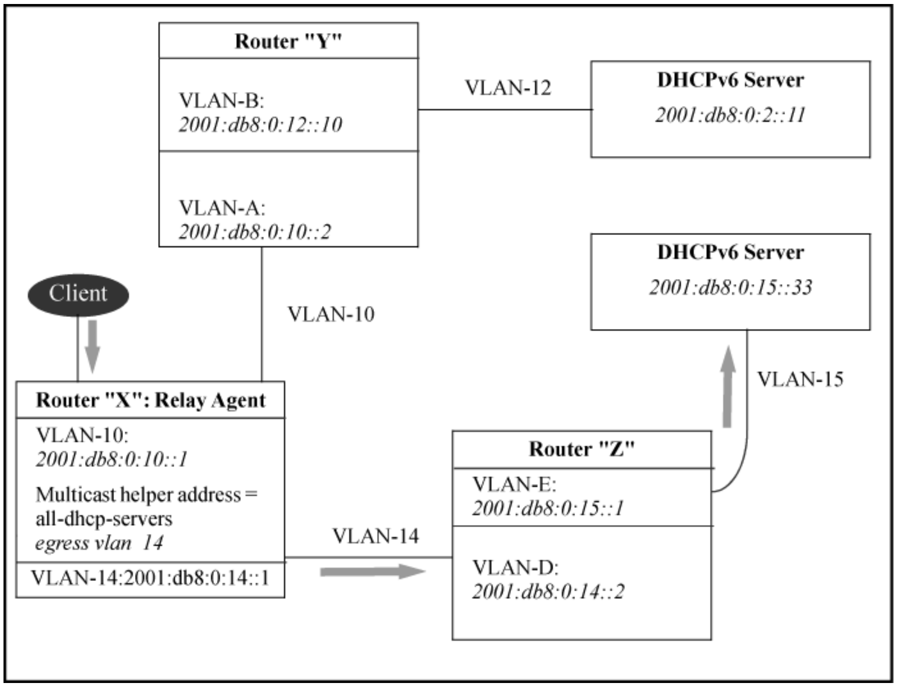
**Figure 18:** *Routing a unicast DHCPv6 request across a multiple-hop topology*




Router "X" is a relay agent configured to forward DHCPv6 service requests received from VLAN 10 to the "all-DHCPv6-servers" multicast helper address (FF05::1:3) through VLAN 14. Router "Z" receives the request from router "X" on VLAN 14. Because router "Z" is configured with unicast helper address 2001:db8:0:15::33 on VLAN

14, the service request is relayed to the DHCPv6 server at 2001:db8:0:15::33 on VLAN 15. (In this example, both router "X" and router "Z" act as DHCPv6 relay agents.)

**Figure 19:** *Routing a multicast DHCPv6 request across a multiple-hop topology*




A multihop relay scenario shown in the figure requires the following:

- All relays in the path except the relay closest to the server have a multicast helper address.
- The last relay in the path to the server has a unicast helper address.

Thus, if router "Z" was not configured with a helper address as shown above, the relayed service request would be restricted to VLAN 14 and would not reach the server at 2001:db8:0:15::33 on VLAN 15.

OSPFv3 is the IPv6 implementation of open shortest path first protocol. (OSPFv2 is the IPv4 implementation of this protocol.) Beginning with software version K.15.01, the switches can be configured to run OSPFv3 either alone or simultaneously with OSPFv2. (OSPFv3 and OSPFv2 run as independent protocols on the routing switch and do not have any interaction when run simultaneously.)

This section describes OSPFv3 terms, basic features, and general operation. Both VLANS and tunnels can be assigned to areas and may be collectively referred to as an IP routing interface. For specific configuration information, turn to the topics referenced in the following command index.

NOTE

In this chapter, "OSPF" refers to OSPFv3 for IPv6 operation unless otherwise stated.

# Overview of OSPFv3

| Factor | Detail |
|---|---|
| Minimum software version | K.15.01 |
| Application | OSPFv3 applications only; runs independent of the OSPFv2 protocol used for IPv4 OSPFv2 applications. |
| Concurrent IPv4/IPv6 operation | Concurrent OSPFv2 and OSPFv3 operation supported on all VLAN interfaces configured on the routing switch. |

Beginning with software version K.15.01, the routing switches support concurrent operation of both OSPFv2 (for IPv4) and OSPFv3 (for IPv6). The two versions of the OSPFv3 protocol operate independent of each other. (For information on OSPFv2 for IPv4 operation, see the latest *Aruba-OS Switch Multicast and Routing Guide* for your routing switch.)

OSPFv3 is a link-state routing protocol applied to IPv6 routers grouped into OSPFv3 areas identified by the IPv6 routing configuration on each routing switch. Each OSPFv3 area includes one or more networks. OSPFv3 routers use hello packets and LSAs to maintain OSPFv3 operation across networks within an area and between areas within an OSPFv3 domain.

## Hello packets

OSPFv3 uses hello packets to initiate and preserve relationships between neighboring routers on the same VLAN interface. OSPFv3 automatically transmits hello packets that are configurable for transmission interval (to indicate a link that has become unavailable).

## Link-state advertisements (LSAs)

OSPFv3 uses LSAs transmitted by each router to update neighboring routers regarding its interfaces and the routes available through those interfaces. Each routing switch in an area also maintains an LSDB that describes the area topology. (All routers in a given OSPFv3 area have identical LSDBs, and each router uses the LSDB to build its own shortest-path tree.) The routing switches used to connect areas to each other flood inter-area-prefix-LSAs, inter-area-router-LSAs, and AS-external-LSAs to backbone area to update it regarding available routes.

Through this means, each OSPFv3 router determines the shortest path between itself and a desired destination router in the same OSPFv3 domain (AS). Routed traffic in an OSPFv3 AS is classified as one of the following:

- Intra-area traffic
- Inter-area traffic
- External traffic

The routing switches support the LSAs listed in the following table. For more information, see RFCs and 3101 (for Type-7-LSA).

**Table 19:** *OSPFv3 LSA types*

| LS type | Description | Use | Flood scope |
|---|---|---|---|
| 0x2001 | Router-LSA | Describes the state of each active interface on a router for a given area. (Excludes loopback interfaces and interfaces that have not achieved full adjacency.) | Area |
| 0x2002 | Network-LSA | Describes the OSPFv3 routers in a given network. | Area |
| 0x2003 | Inter-area-prefix-LSA | Describes the route to a prefix in another OSPFv3 area of the same AS. Propagated through backbone area to other areas. | Area |
| 0x2004 | Inter-area-router-LSA | Describes the route to an ASBR in another OSPFv3 normal area (including the backbone area) of the same AS. (Excludes prefixes for link-local addresses.) Propagated through backbone area to other areas. (Excludes any ASBR in the same area as the router sending the LSA.) | Area |
| 0x4005 | AS-external-LSA | Describes the route to a destination prefix in another AS (external route). (Excludes prefixes for link-local addresses.) Originated by ASBR in normal or backbone areas of an AS and propagates through backbone area to other normal areas.Does not flood over virtual links and is not summarized in virtual links. For injection into an NSSA, an NSSA ABR generates a type-7-default-LSA advertising the default route (::/0). | AS |
| 0x2007 | NSSA-LSA | Describes the route to a destination in another AS (external route). Originated by ASBR in NSSA. ABR translates type-7 LSAs to AS-external-LSAs for injection into the backbone area. | NSSA |
| 0x0008 | Link-LSA | For other routers on the same VLAN interface, describes the router's link-local address and any other IPv6 prefixes reachable on the VLAN. Link LSAs are not flooded over virtual links. | Link-local |
| 0x2009 | Intra-area-prefix-LSA | Generated on transit links within an area by the DR operating on those links. Also, every OSPFv3 router generates this LSA to refer to stub and loopback prefixes on the router. | Area |

# OSPFv3 RFC compliance

NOTE: This feature is available only on switches running KB software.

The OSPFv3 features covered comply with the following:

- RFC 5340
- RFC 3101 option
- IP sec for OSPFv3 is partially compliant with RFC 4552.

This feature supports authentication for OSPFv3 routing traffic on Aruba devices via IP sec. The authentication support for OSPFv3 is provided in compliance with RFC 4552. Authentication is supported using AH protocol. Authentication and confidentiality support using ESP is not supported in this release.

# area <AREA-ID> security

### Syntax

```
area <AREA-ID> security authentication ipsec spi <SPI> [ {md5|sha1} key {hex |
ascii} [encrypted] <KEY> ]
```

```
no area <AREA-ID> security authentication ipsec spi <SPI> [ {md5|sha1} key {hex |
ascii} [encrypted] <KEY> ]
```

### Description

Configure OSPFv3 security parameters. By default, security is not configured.

### Parameters

**Authentication**

Configure AH authentication.

**Ipsec**

Use IPSec security.

**spi <SPI>**

Enter a value between 256 and 4294967296 to be used as Security Parameter Index.

**md5**

MD5 authentication; the key should be 16 ASCII characters or 32 hexadecimal digits.

**sha1**

SHA1 authentication; the key should be 20 ASCII characters or 40 hexadecimal digits.

**ascii**

Specify the key in characters.

**encrypted**

Specify that the key is encrypted.

**key <KEY>**

The ASCII or hexadecimal or encrypted value of the authentication key.

### Examples

```
Switch(config# router ospf3

Switch(ospf3)# area backbone security authentication ipsec spi 400 md5 key ascii
"abcdefghijklmnop"

Switch(ospf3)# area 0.0.0.1 security authentication ipsec spi 401 md5 key hex
"31323334353637383930313233343536"

Switch(ospf3)# area 0.0.0.2 security authentication ipsec spi 402 sha1 key hex
"0987654321abcde0987654321abcde0987654321"

Switch(ospf3)# area 0.0.0.3 security authentication ipsec spi 403 sha1 key ascii
"0987654321!@#$%^&*()"

Switch(ospf3)# area 0.0.0.5 security authentication ipsec spi 403 sha1 key ascii
encrypted "R/8Npx7MjsR1kPbPyyh3PIEVasWO8j9W1T32dg8/gEAENgVnW+28dCnRdNne2eIW"
```

NOTE

OSPFv3 IPSec configurations are displayed in `show running-config` only when "include-credentials" is enabled.

## area <AREA-ID> security disable

### Syntax

```
no area <AREA-ID> security disable
```

### Description

Disable OSPFv3 IPsec security for an OSPFv3 area.

### Examples

```
Switch(config)# router ospf3

Switch(ospf3)# area 0.0.0.2 security disable
```

## ipv6 ospf3 security

### Syntax

```
no ipv6 ospf3 security authentication ipsec spi <SPI> [ {md5|sha1} key {hex |
ascii} [encrypted] <KEY> ]
```

### Description

Configure OSPFv3 IPsec security parameters for a VLAN or Tunnel interface.

### Parameters

**authentication**

Configure AH authentication.

**ipsec**

Use IPsec security.

**spi <SPI>**

Enter a value between 256 and 4294967296 to be used as the IPsec security parameter index.

**md5**

MD5 authentication; the key should be 16 ASCII characters or 32 hexadecimal digits.

**sha1**

SHA1 authentication; the key should be 20 ASCII characters or 40 hexadecimal digits.

**ascii**

Specify the key in characters.

**hex**

Specify the key in hexadecimal.

**encrypted**

Specify that the key is encrypted.

**key <KEY>**

The ASCII or hexadecimal or encrypted value of the authentication key.

### Examples

```
Switch(config)# vlan 2
Switch(vlan-2)# name "VLAN2"
Switch(vlan-2)# tagged A1
Switch(vlan-2)# ipv6 address 2001::1/64
Switch(vlan-2)# ipv6 ospf3 area backbone
Switch(vlan-2)# ipv6 ospf3 security authentication ipsec spi 200000 sha1 key ascii
"abcdefghijklmnopqrst"
Switch(vlan-2)# exit
```

NOTE

An OSPFv3 IPsec configuration on a VLAN or tunnel interface overrides the IPsec configurations made in the area to which it belongs.

## ipv6 ospf3 security disable

### Syntax

```
no ipv6 ospf3 security disable
```

### Description

Disable OSPFv3 IPsec security for a VLAN or tunnel interface.

### Parameter

**disable**

Disable IPSec security.

### Examples

```
Switch(vlan-22)# ipv6 ospf3 security disable
Switch(vlan-22)# no ipv6 ospf3 security disable
```

# show ipv6 ospf3 security

### Syntax

```
show ipv6 ospf3 security {area <area-id> |
interface {vlan <vlan-id> | tunnel <tunnel-id>} }
```

### Description

Shows security information for OSPFv3.

If **area <area-id>** is specified, area related security information is displayed.

If **interface vlan <vlan-id>** is specified, VLAN interface related security information is displayed.

If **interface tunnel <tunnel-id>** is specified, tunnel interface related security information is displayed.

### Options

**area <area-id>**

Displays area related security information for the given area.

**interface vlan <vlan-id>**

Displays VLAN related security information for the given VLAN.

**interface tunnel <tunnel-id>**

Displays tunnel related security information for the given tunnel.

### Examples

```
Switch(ospf3)# show ipv6 ospf3 security area 16

Key Information

  SPI         : 44444445           Active        : Yes
  SA          : Inbound, Outbound   Security Mode : 1
  Auth. Mode  : SHA1
  Auth. Key   : 31323334353637383930313233343536373839 30
```

```
Switch(vlan-5)# show ipv6 ospf3 security interface vlan 5

 IPSec Secured Packets Sent: 16
 IPSec Secured Packets Received: 73
 Received Packets Dropped: 0

 Number of Interfaces Following the Area Security Configuration: 5

 Key Information

  SPI         : 200000             Active        : Yes
  SA          : Inbound, Outbound   Security Mode : 1
  Auth. Mode  : SHA1
  Auth. Key   : abcdefghijklmnopqrst
```

```
Switch# show ipv6 ospf3 security interface tunnel 1

 IPSec Secured Packets Sent: 15
 IPSec Secured Packets Received: 26
 Received Packets Dropped: 0
```

```
Number of Interfaces Following the Area Security Configuration: 6

Key Information

 SPI         : 12345               Active        : Yes
 SA          : Inbound, Outbound   Security Mode : 1
 Auth. Mode  : MD5
 Auth. Key   : plmoknijbuhvygvt
```

## show ipv6 ospfv3 ipsec summary

### Syntax

```
show ipv6 ospfv3 ipsec summary
```

### Description

Show summarized IPsec information about all OSPFv3 areas and interfaces.

### Example

```
Switch(config)# show ipv6 ospf3 ipsec summary

Area Summary:

Area ID: 0.0.0.0

 SA  VLAN-Intfc SPI        Local IP
 --- ---------- ---------- ----------------------------------------
 Remote IP                                Proto Direction
 ---------------------------------------- ----- ---------
 2   any        100000     ::
 any                                      AH    Outbound
 3   any        100000     fe80::42a8:f0ff:fe9e:900
 any                                      AH    Inbound
 6   any        100000     ::
 any                                      AH    Outbound
 7   any        100000     fe80::42a8:f0ff:fe9e:900
 any                                      AH    Inbound
 8   any        100000     ::
 any                                      AH    Outbound
 9   any        100000     fe80::42a8:f0ff:fe9e:900
 any                                      AH    Inbound
Area ID: 0.0.0.2


 SA  VLAN-Intfc SPI        Local IP
 --- ---------- ---------- ----------------------------------------
 Remote IP                                Proto Direction
 ---------------------------------------- ----- ---------
Area ID: 0.0.0.3


 SA  VLAN-Intfc SPI        Local IP
 --- ---------- ---------- ----------------------------------------
 Remote IP                                Proto Direction
 ---------------------------------------- ----- ---------
```

```
Area ID: 0.0.0.4


 SA  VLAN-Intfc SPI        Local IP
 --- ---------- ---------- ---------------------------------------
 Remote IP                               Proto Direction
 --------------------------------------- ----- ---------
 10  any        402        ::
 any                                     AH    Outbound
 11  any        402        fe80::42a8:f0ff:fe9e:900
 any                                     AH    Inbound
Area ID: 0.0.0.5


 SA  VLAN-Intfc SPI        Local IP
 --- ---------- ---------- ---------------------------------------
 Remote IP                               Proto Direction
 --------------------------------------- ----- ---------
 12  any        403        ::
 any                                     AH    Outbound
 13  any        403        fe80::42a8:f0ff:fe9e:900
 any                                     AH    Inbound

Interface Summary:

 SA  VLAN-Intfc SPI        Local IP
 --- ---------- ---------- ---------------------------------------
 Remote IP                               Proto Direction
 --------------------------------------- ----- ---------
 0   vlan-2     200000     ::
 any                                     AH    Outbound
 1   vlan-2     200000     fe80::42a8:f0ff:fe9e:900
 any                                     AH    Inbound
 4   vlan-22    122000     ::
 any                                     AH    Outbound
 5   vlan-22    122000     fe80::42a8:f0ff:fe9e:900
 any                                     AH    Inbound
 34  tunnel-1   12345      ::
 any                                     AH    Outbound
 35  tunnel-1   12345      fe80::42a8:f0ff:fe9e:900
 any
```

# show crypto ipsec

### Syntax

```
show crypto ipsec {sa | summary} [aruba-vpn | ospfv3]
 [spi <SPI> | ip-address <IP-ADDR>]
```

### Description

Show IPsec SA statistics or summarized information.

**Parameters**

**sa**

Shows IPsec SA statistics.

**summary**

Shows summarized IPsec information.

**Options**

**aruba-vpn**

Displays aruba-vpn related IPsec SA statistics.

**ospfv3**

Displays OSPFv3 related IPsec SA statistics

**spi <SPI>**

Displays IPaec SA statistics that match the given SPI.

**ip-address <IP-ADDR>**

Displays IPsec SA statistics that match the given IP address.

**Examples**

```
Switch(config)# show crypto ipsec summary

 Crypto IPSec Summary:

 SA  VLAN-Intfc SPI        Local IP
 --- ---------- ---------- ---------------------------------------
 Remote IP                               Proto Appln      Direction
 --------------------------------------- ----- ---------- ---------
 0   vlan-2     200000     ::
 ff02::5                                 AH    OSPFv3     Outbound
 1   vlan-2     200000     fe80::42a8:f0ff:fe9e:900
 ff02::5                                 AH    OSPFv3     Inbound
 2   vlan-12    100000     ::
 ff02::5                                 AH    OSPFv3     Outbound
 3   vlan-12    100000     fe80::42a8:f0ff:fe9e:900
 ff02::5                                 AH    OSPFv3     Inbound
 4   vlan-22    122000     ::
 ff02::5                                 AH    OSPFv3     Outbound
 5   vlan-22    122000     fe80::42a8:f0ff:fe9e:900
 ff02::5                                 AH    OSPFv3     Inbound
 6   vlan-201   100000     ::
 ::                                      AH    OSPFv3     Outbound
 7   vlan-201   100000     fe80::42a8:f0ff:fe9e:900
 ff02::5                                 AH    OSPFv3     Inbound
 8   vlan-202   100000     ::
 ::                                      AH    OSPFv3     Outbound
 9   vlan-202   100000     fe80::42a8:f0ff:fe9e:900
 ff02::5                                 AH    OSPFv3     Inbound
 10  vlan-204   402        ::
 ff02::5                                 AH    OSPFv3     Outbound
 11  vlan-204   402        fe80::42a8:f0ff:fe9e:900
 ff02::5                                 AH    OSPFv3     Inbound
 12  vlan-205   403        ::
 ff02::5                                 AH    OSPFv3     Outbound
```

```
 13  vlan-205   403        fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 14  vlan-206   404        ::
 ff02::5                                  AH    OSPFv3     Outbound
 15  vlan-206   404        fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 16  vlan-207   405        ::
 ff02::5                                  AH    OSPFv3     Outbound
 17  vlan-207   405        fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 18  vlan-208   406        ::
 ff02::5                                  AH    OSPFv3     Outbound
 19  vlan-208   406        fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 20  vlan-209   408        ::
 ff02::5                                  AH    OSPFv3     Outbound
 21  vlan-209   408        fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 22  vlan-210   409        ::
 ff02::5                                  AH    OSPFv3     Outbound
 23  vlan-210   409        fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 24  vlan-211   410        ::
 ff02::5                                  AH    OSPFv3     Outbound
 25  vlan-211   410        fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 26  vlan-212   10000      ::
 ff02::5                                  AH    OSPFv3     Outbound
 27  vlan-212   10000      fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 28  vlan-213   10001      ::
 ff02::5                                  AH    OSPFv3     Outbound
 29  vlan-213   10001      fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 30  vlan-214   10002      ::
 ff02::5                                  AH    OSPFv3     Outbound
 31  vlan-214   10002      fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 32  vlan-215   44444445   ::
 ff02::5                                  AH    OSPFv3     Outbound
 33  vlan-215   44444445   fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH    OSPFv3     Inbound
 34  tunnel-1   12345      ::
 ff02::5                                  AH    OSPFv3     Outbound
 35  tunnel-1   12345      fe80::42a8:f0ff:fe9e:900
 ff02::5                                  AH
Inbound

Switch(config)# show crypto ipsec sa

  Crypto IPSec Status

  VLAN Interface          : vlan-2
  SA-ID                   : 0
  Source Address          : ::
  Destination Address     : ff02::5
  Source Port             : 0              Destination Port   : 0
```

```
SPI                     : 200000
Encapsulation Protocol  : AH
Encryption              : NONE          Hash                : SHA1
PFS                     : 0             PFS Group           : 0
Mode                    : Transport
Key Life                : 0             Remaining Key Life  : 0
Key Size                : 0             Remaining Key Size  : 0
Application             : OSPFv3

VLAN Interface          : vlan-2
SA-ID                   : 1
Source Address          : fe80::42a8:f0ff:fe9e:900
Destination Address     : ff02::5
Source Port             : 0             Destination Port    : 0
SPI                     : 200000
Encapsulation Protocol  : AH
Encryption              : NONE          Hash                : SHA1
PFS                     : 0             PFS Group           : 0
Mode                    : Transport
Key Life                : 0             Remaining Key Life  : 0
Key Size                : 0             Remaining Key Size  : 0
Application             : OSPFv3

VLAN Interface          : vlan-12
SA-ID                   : 2
Source Address          : ::
Destination Address     : ff02::5
Source Port             : 0             Destination Port    : 0
SPI                     : 100000
Encapsulation Protocol  : AH
Encryption              : NONE          Hash                : MD5
PFS                     : 0             PFS Group           : 0
Mode                    : Transport
Key Life                : 0             Remaining Key Life  : 0
Key Size                : 0             Remaining Key Size  : 0
Application             : OSPFv3

VLAN Interface          : vlan-12
SA-ID                   : 3
Source Address          : fe80::42a8:f0ff:fe9e:900
Destination Address     : ff02::5
Source Port             : 0             Destination Port    : 0
SPI                     : 100000
Encapsulation Protocol  : AH
Encryption              : NONE          Hash                : MD5
PFS                     : 0             PFS Group           : 0
Mode                    : Transport
Key Life                : 0             Remaining Key Life  : 0
Key Size                : 0             Remaining Key Size  : 0
Application             : OSPFv3
```

# OSPFv3 area types

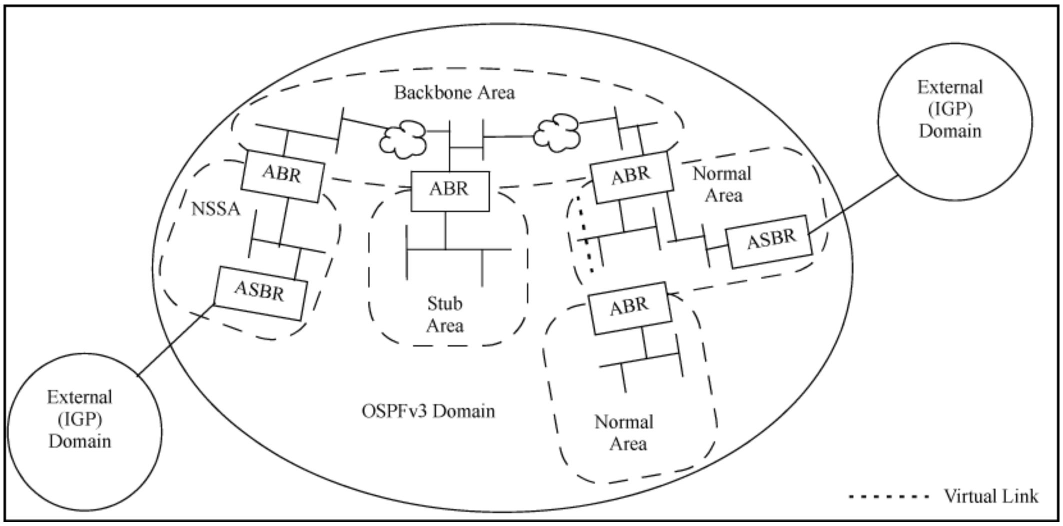
OSPFv3 is built upon a hierarchy of network areas. All areas for a given OSPFv3 domain reside in the same AS. An AS is defined as a number of contiguous networks, all of which share the same interior gateway routing protocol.

An AS can be divided into multiple areas, including the backbone (area 0). Because each area represents a collection of contiguous networks and hosts, the topology of a given area is not known by the internal routers in any other area. Areas define the boundaries to which router-LSAs and network-LSAs are broadcast. This limits the amount of LSA flooding that occurs within the AS and also helps to control the size of the link-state databases (LSDBs) maintained in OSPFv3 routers. An area is represented in OSPFv3 by either a 32-bit dotted-decimal address or a number. Area types include:

- Backbone
- Normal
- Not-so-stubby (NSSA)
- Stub

All areas in an AS must connect with the backbone through one or more ABRs. If a normal area is not directly connected to the backbone area, it must be configured with a virtual link to an ABR that is directly connected to the backbone. The stub and NSSA area types do not allow virtual link connections to the backbone area.

**Figure 20:** *Example of an AS with multiple areas and external routes*




## Normal area

This area type allows inter-area-prefix-LSAs and AS-external-LSAs to and from the backbone area. (As noted earlier, the backbone area is a special type of normal area.) A normal area connects to the AS backbone area through one or more ABRs (physically or through a virtual link). ASBRs are allowed in normal areas.

## Backbone area

Every AS must have one (and only one) backbone area (identified as area 0 or 0.0.0.0). The ABRs of all other areas in the same AS connect to the backbone area, either physically through an ABR or through a configured, virtual link. The backbone is a special type of normal area and serves as a transit area for carrying the inter-area-prefix-LSAs, AS-external-LSAs, and routed traffic between non-backbone areas, as well as the router-LSAs and network-LSAs and routed traffic internal to the area. ASBRs are allowed in backbone areas.

## Stub area

This area connects to the AS backbone through one or more ABRs. It does not allow an internal ASBR and does not allow AS-external-LSAs. A stub area supports these actions:

- Advertise the area's inter-area routes to the backbone area.
- Advertise inter-area routes from other areas.
- Use the inter-area-prefix-LSA default route to advertise routes to an ASBR and to other areas.

You can configure the stub area ABR to do the following:

- Suppress advertising some or all of the area's summarized internal routes into the backbone area.
- Suppress LSA traffic from other areas in the AS by replacing inter-area-prefix-LSAs and the default external route from the backbone area with the default route (::/0).

Virtual links are not allowed for stub areas.

## Not-so-stubby-area (NSSA)

This area type connects to the backbone area through one or more ABRs. NSSAs are used where an ASBR exists in an area where you want to:

- Block injection of external routes from other areas of the AS.
- Advertise type-7-LSA external routes (learned from the ASBR) to the backbone area as AS-external-LSAs.

NSSAs also support the following:

- Advertise inter-area-prefix-LSAs from the backbone area into the NSSA. (If `no-summary` is enabled, the NSSA ABR suppresses these LSAs from the backbone and, instead, injects the inter-area-prefix-LSA default route into the NSSA.)
- Advertise NSSA inter-area-prefix-LSAs to the backbone area.

In the above operation, the ASBR in the NSSA injects external routes as type-7-LSAs. (AS-external-LSAs are not allowed in an NSSA.) The ABR connecting the NSSA to the backbone converts the type-7-LSAs to AS-external-LSAs and injects them into the backbone area for propagation to networks in the backbone and to any normal areas configured in the AS. The ABR also injects inter-area-prefix-LSAs from the backbone area into the NSSA.

The default route (::/0) is always injected into the NSSA as either a type-7-LSA or an inter-area-LSA, depending on the `no-summary` configuration (default: disabled). That is, if inter-area-prefix-LSAs are allowed in the NSSA (the default operation), a type-7-LSA default route (::/0) is injected into the NSSA. But if inter-area-prefix-LSAs are blocked (by enabling `no-summary`), the inter-area-prefix-LSA default route is injected into the NSSA instead of the type-7-LSA default route.

You can also configure the NSSA ABR to suppress advertising some or all of the area's summarized internal or external routes into the backbone area. See **router ospf3 area** on page 199.

Virtual links are not allowed for NSSAs.

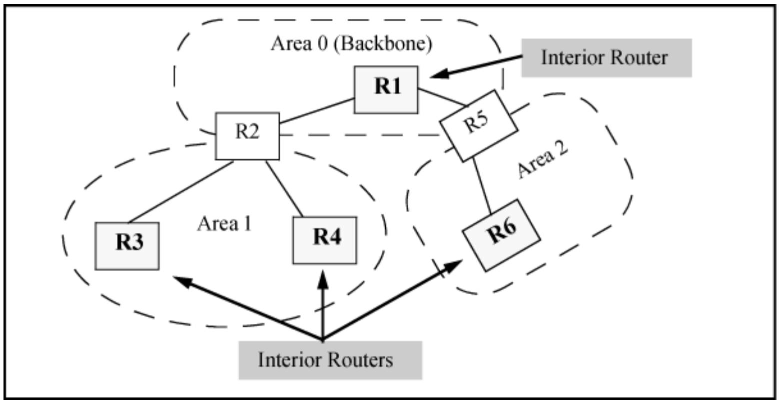
# OSPFv3 router types

This section explains the various types of OSPFv3 routers.

## Interior routers

This type of OSPFv3 router belongs to only one area. Interior routers flood router-LSAs to all routers in the same area and maintain identical LSDBs. In the following figure, routers R1, R3, R4, and R6 are all interior routers because they link to other routers in the same area.

**Figure 21:** *Example of interior routers*

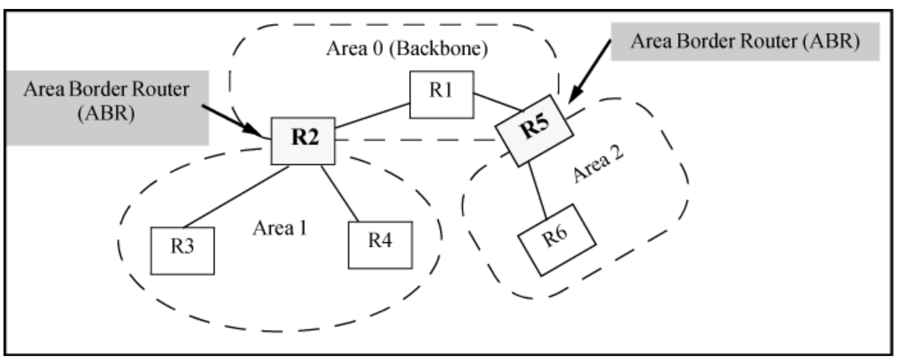



## Area border routers (ABRs)

This type of OSPFv3 router has membership in multiple areas. ABRs are used to connect the various areas in an AS to the backbone area for that AS. Multiple ABRs can be used to connect a given area to the backbone, and a given ABR can belong to multiple areas other than the backbone. An ABR maintains a separate LSDB for each area to which it belongs. (All routers within the same area have identical LSDBs.) The ABR is responsible for flooding inter-area-prefix-LSAs and inter-area router LSAs between its border areas.

You can reduce this LSA flooding by configuring area ranges. An area range enables you to assign an aggregate address to a range of IPv6 addresses. This aggregate address is advertised instead of all the individual addresses it represents. You can assign up to eight ranges in an OSPFv3 area. In the following figure, routers R2 and R5 are ABRs because they both have membership in more than one area.

**Figure 22:** *Example of deploying ABRs to connect areas to the backbone*




### Configuring an ABR to use a virtual link to the backbone

All OSPFv3 ABRs (area border routers) must have either a direct, physical or indirect, virtual link to the OSPFv3 backbone area (0.0.0.0 or 0). If an ABR does not have a physical link to the area backbone, it can use a virtual link to provide a logical connection to another ABR having a direct physical connection to the area backbone. Both ABRs must belong to the same area, and this area becomes a transit area for traffic to and from the indirectly connected ABR.

NOTE

A backbone area can be purely virtual with no physical backbone links. Also note that virtual links can be linked in a series. If so, one end may not be physically connected to the backbone.

Because both ABRs in a virtual link connection are in the same OSPFv3 area, they use the same transit area ID. This setting is configured using `area` *`area-id`* `virtual-link` *router-id* in the **router ospf3** context and should match the area ID value configured on both ABRs in the virtual link.

The ABRs in a virtual link connection also identify each other with a neighbor router setting:

- On the ABR having the direct connection to the backbone area, the neighbor router is the router ID (in decimal or 32-bit dotted decimal format) of the router interface needing a logical connection to the backbone.
- On the opposite ABR (the one needing a logical connection to the backbone), the neighbor router is the router ID (in decimal or 32-bit dotted decimal format) of the ABR that is directly connected to the backbone.

NOTE

By default, the router ID is the lowest numbered IPv4 address or (user-configured) IPv4 loopback interface configured on the device.

When you establish an area virtual link, you must configure it on both of the ABRs (both ends of the virtual link).

## Adjusting virtual link performance by changing the interface settings

The following OSPFv3 interface parameters are automatically set to their default values for virtual links. No change to the defaults is usually required unless needed for specific network conditions. This is a subset of the parameters described under **Adjusting performance by changing the VLAN interface settings** on page 228. (The `cost` and `priority` settings are not configurable for a virtual link, and the commands for reconfiguring the settings are accessed in the **router OSPFv3** context instead of the **VLAN** context.)

NOTE

The parameter settings described in this section for virtual links must be the same on the ABRs at both ends of a given link.

| Parameter | Default | Page |
|---|---|---|
| `dead-interval` | 40 seconds | **ipv6 ospf3 dead-interval** on page 229 |
| `hello-interval` | 10 seconds | **ipv6 ospf3 hello-interval** on page 229 |
| `retransmit-interval` | 5 seconds | **ipv6 ospf3 retransmit-interval** on page 230 |
| `transit-delay` | 1 second | **ipv6 ospf3 transit-delay** on page 230 |

## Configuring ranges on an ABR to reduce advertising to the backbone

## router ospf3 area

### Syntax

```
router ospf3 area [ospf3-area-id|backbone] range ipv6-addr/prefix [type | summary
[ cost 1 - 16777215 ]|inter-area|nssa][no-advertisement]
```

`router ospf3 area [`*`ospf3-area-id`*`|backbone] range `*`ipv6-addr/prefix`*` [type | summary [ cost `*`1 - 16777215`*` ]|inter-area|nssa]`

**Description**

Use this command on a routing switch intended to operate as an ABR for the specified area to do either of the following:

- Simultaneously create the area and corresponding range setting for routes to summarize or block.
- For an existing area, specify a range setting for routes to summarize or block (prevent).

**Options**

***ospf3-area-id***

Same area ID as in **area [ospf3–area-id|backbone][ normal ]** on page 219, except you cannot use a backbone area number (**0** or **0.0.0.0**) for a stub area or NSSA.

**range *ipv6-addr/prefix***

Defines the range of RAs to either summarize for injection into the backbone area or to prevent from being injected into the backbone area.

The *`ipv6-addr`* value specifies the IPv6 address portion of the range, and *`prefix`* specifies the leftmost significant bits in the address.

The ABR for the specified area compares the IPv6 address of each outbound RA with the address and significant bits in the mask to determine which routes to select for either summarizing or blocking.

For example, `2001:db8:0:f::/64` defines a range including any address that has 2001:db8:0:f in the leftmost 64 bits.

**[type | summary [ cost *1 - 16777215* ]|inter-area|nssa][ no-advertise ]**

Configures the type of route summaries to advertise or block.

**[summary [ cost *1 - 16777215* ]]**

Specifies internal routes in the configured range of route advertisements. If `no-advertise` (above) is used in the command, then the ABR prevents the selected internal routes from being summarized in a type-3 LSA and advertised to the backbone. If `no-advertise` is not used in the command, then the selected routes are summarized to the backbone in a type-3 LSA.

**cost *1 - 16777215***

User configured cost for an area summary range. If cost is specified, then the range will advertise the specified cost instead of the calculated cost.

**inter-area**

Specifies internal routes in the configured range of RAs. If `no-advertise` (below) is used in the command, the ABR prevents the selected internal routes from being summarized in an inter-area-prefix-LSA and advertised to the backbone.

**nssa**

Specifies external routes (type-7-LSAs) in the configured range of RAs.

The `no` form of the command removes the specified range from the configuration.

**no-advertise**

Use this keyword only if you want to configure the ABR to prevent advertisement to the backbone of a specified range of routes. (This has the effect of "hiding" the specified range from the backbone area.)

If you do not use this option, the ABR advertises the specified range of routes according to the `type [ inter-area | nssa ]` selection described above.

**Assigning a Cost**

The `cost` parameter provides a way to define a fixed, user-assigned cost of an LSA type 3 summarized prefix.

```
Switch(vlan-100)#ipv6 ospf3 cost 20
```

CLI to verify the set value:

```
Switch(vlan-1)#show ipv6 ospf3 interface vlan 100

OSPFv3 configuration and statistics for VLAN 100

Interface     Status     Area ID     State     Cost     Pri     Passive
------------------------------------------------------------------
vlan-100      Enabled    0.0.0.0     DROTHER   20       1       No

Switch(vlan-1)#
```

The `no` form removes the specified range from the configuration.

**ABR allowing or blocking advertisement of a range of internal routes available in an area**

```
Example of defining a range of internal routes to advertise to the backbone
```

The following command defines a range of internal routes in area 30 to summarize for injection into the backbone area. (In this example, area 30 can be a normal or stub area, or an NSSA.)

```
Switch(ospf3)# area 30 range 2001:db8:1a/48 type inter-area
```

For the same range of routes, you can use either of the following commands to block injection of a range of inter-area routes (inter-area-prefix-LSAs) from area 30 into the backbone.

```
Switch(ospf3)# area 30 range 2001:db8:1a/48 type inter-area no-advertise
```

**Examples of allowing or blocking a range of external routes available through an ASBR in an NSSA follow.**

```
This example applies only to external routes that can be advertised from an NSSA
to the backbone.
```

```
Switch(ospf3)# area 7 range 2001:db8:5f:1::/64 type nssa
```

```
Switch(ospf3)# area 7 range 2001:db8:7a:15::/64 type nssa no-advertise
```

Defines the range of external routes in the Area 7 NSSA to block from advertising to the backbone.

## Autonomous system boundary router (ASBR)

This type of OSPFv3 router runs one or more Interior Gateway protocols and serves as a gateway to other autonomous systems operating with interior gateway protocols. The ASBR imports and translates different protocol routes into OSPFv3 through redistribution. ASBRs can be used in backbone areas, normal areas, and NSSAs, but not in stub areas.

### Designated routers

In an OSPFv3 network having two or more routers, one router is elected to serve as the designated router (DR) and another router to act as the backup designated router (BDR). All other routers in the area forward their routing information to the DR and BDR, and the DR forwards this information to all routers in the network. This minimizes the amount of repetitive information that is forwarded on the network by eliminating the need for each individual router in the area to forward its routing information to all other routers in the network. If the area includes multiple networks, each network elects its own DR and BDR.

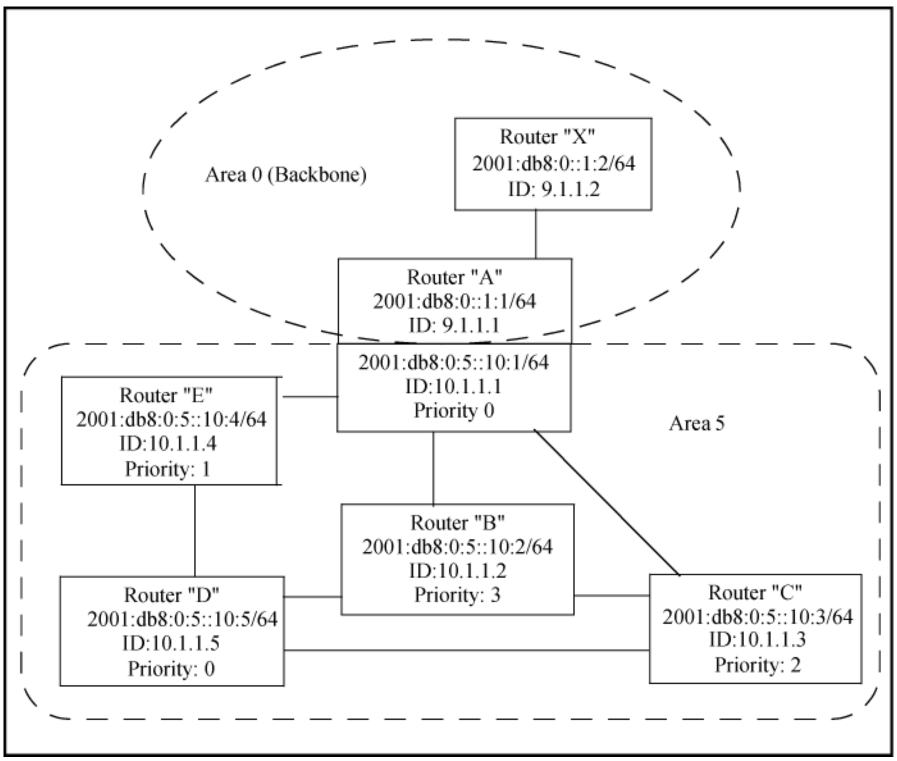
In an OSPFv3 network with no DR and no BDR, the neighboring router with the highest priority is elected as the DR, and the router with the next highest priority is elected as the BDR. If the DR goes off-line, the BDR automatically becomes the DR, and the router with the next highest priority then becomes the new BDR. If multiple HPE routing switches on the same OSPFv3 network are declaring themselves as DRs, both priority and router ID are used to select the designated router and backup designated routers.

Priority is configurable by using the `vlan vid ipv6 ospfv3 priority 0 - 255` command at the interface level. You can use this parameter to help bias one router as the DR. If two neighbors share the same priority, the router with the highest router ID is designated as the DR. The router with the next highest router ID is designated as the BDR.

For example, in the following figure, the DR and BDR for the 2001:db8:0:5::/64 network in area 5 are determined as follows:

- Router A Priority: 0 Cannot become a DR or BDR.
- Router B Priority: 3 DR for the 2001:db8:0:5::/64 network.
- Router C Priority: 2 BDR for the 2001:db8:0:5::/64 network.
- Router D Priority: 0 Cannot become a DR or BDR.
- Router E Priority: 1 Becomes the new BDR if router B becomes unavailable and router C becomes the new DR.

**Figure 23:** *Example of designated routers in an OSPFv3 area*




To verify the router priority on an interface, use the `show ipv6 ospf3 interface` command and check the **Pri** field.

NOTE

Once a DR is elected, the DR and BDR status do not change if a higher-priority router joins the network, unless the DR or BDR goes down.

By default, the router ID is typically the lowest-numbered IPv4 loopback address or the lowest-numbered (user-configured) loopback interface configured on the device.

If multiple networks exist in the same OSPFv3 area, the recommended approach is to ensure that each network uses a different router as its DR. Otherwise, if a router is a DR for more than one network, latency in the router could increase because of the increased traffic load resulting from multiple DR assignments.

When only one router on an OSPFv3 network claims the DR role despite neighboring routers with higher priorities or router IDs, this router remains the DR. This is also true for BDRs.

The DR and BDR election process is performed when one of the following events occurs:

- An interface is in a waiting state and the wait time expires
- An interface is in a waiting state and a hello packet is received that addresses the BDR
- A change in the neighbor state occurs, such as:
    - A neighbor state transitions from 2 or higher
    - Communication to a neighbor is lost
    - A neighbor declares itself to be the DR or BDR for the first time

# OSPFv3 activation and dynamic configuration

All configuration commands affecting OSPFv3 (except reconfiguring the router ID) are dynamically implemented and can be used without restarting OSPFv3 routing.

## Configuration procedures for OSPFv3

To begin using OSPFv3 on the routing switch:

**Procedure**

1. Enable IPv6 on at least one VLAN interface.
2. In the **global config** context, use `ipv6 unicast-routing` to enable routing
3. Execute `router ospf3 enable` to enable OSPFv3 routing.
4. Use `area` in the **ospf3** context to assign the areas to which the routing switch will be attached.
5. Assign VLAN interfaces to the configured areas by moving to each **VLAN** context and using the command `ipv6 ospf3 area` *`ospf-area-id`* assigns all interfaces in the VLAN to the same area.
6. Optional: Assign loopback interfaces to OSPFv3 areas by using the `interface loopback` *`0 - 7`* `ipv6 ospf3 area` command.
7. Optional: On each routing switch used as an ASBR in your OSPFv3 domain:
    a. Configure route-maps to permit route prefixes you want redistributed in your OSPFv3 domain and to deny all others.
    b. Configure redistribution to enable importing the static and connected routes you want to make available in the domain.
8. Optional: Configure ranges on ABRs to reduce inter-area RA.
9. Optional: Use administrative distance to influence route choices.
10. Optional: Enforce strict LSA operation for graceful restart helper mode.
11. Optional: Adjust performance by changing the IP routing interface settings, if needed. Includes `cost`, `dead-interval`, `hello-interval`, and `priority` commands.
12. Configure virtual links for any areas not directly connected to the backbone.

## Configuration rules

- If the switch is to operate as an ASBR, you must enable redistribution; see **Configuration procedures for OSPFv3** on page 203. When you do so, ASBR capability is automatically enabled. For this reason, you should first configure route policy and redistribution filters on the ASBR. Otherwise, all possible external routes will be allowed to flood the domain.

- Each IP routing interface on which you want OSPFv3 to run must be assigned to one of the defined areas. When a VLAN interface is assigned to an area, the IPv6 addresses configured on that VLAN are automatically included in the assignment.

# OSPFv3 global and interface settings

When first enabling OSPF, you may want to consider configuring ranges and restricting redistribution (if an ASBR is used) to avoid unwanted advertisements of external routes. You may also want to enable OSPFv3 traps to enhance troubleshooting. However, it is generally recommended that the remaining parameters with non-null default settings be left as-is until you have the opportunity to assess OSPFv3 operation and determine whether any adjustments to default settings are warranted.

For information on when to use the global and per-interface commands used with OSPFv3, see **ip router-id** on page 217. For detailed information on each command, see the page listed for each command.

Before enabling OSPFv3, ensure that `ipv6 unicast-routing` is enabled. Also, either begin each command with `router ospf3`, or execute `router ospf3` at the global CONFIG level and then execute the individual commands in that context. For example:

**Syntax**

```
Switch(config)# router ospf3
```

```
Switch(ospf3)# enable
```

Use the appropriate **interface** context to set interface level OSPFv3 parameters for the desired interface. To access this context level, use `vlan` *`vid`* or `interface tunnel` *`tunnel-id`* either to move to the **interface** context level or to specify that context from the **global config** level.

**Enabling OSPF**

```
Switch(config)# vlan 20
Switch(vlan-20)# ipv6 ospf3 cost 15
Switch(config)# vlan 20 ipv6 ospf3 cost 15
Switch(config)# interface tunnel 3
Switch(tunnel-3)# ipv6 ospf3 cost 15
Switch(config)# interface tunnel 3 ipv6 ospf3 cost 15
```

# Configuring a virtual link

For more information, see **Configuring an ABR to use a virtual link to the backbone** on page 198

## area virtual-link router-id

**Syntax**

`area [`*`area-id`*` ] virtual-link [`*`router-id`*`]`

`no area [`*`area-id`*` ] virtual-link [`*`router-id`*`]`

**Description**

In the **ospf3** context, used on a pair of ABRs at opposite ends of a virtual link in the same area to configure the virtual link connection.

The `no` form removes the virtual link.

**Options**

***area-id***

This must be the same for both ABRs in the link and is the area number of the virtual link transit area in either decimal or 32-bit dotted decimal format.

If `area-id` is not already configured on the routing switch, this command creates it.

***router-id***

On an ABR directly connected to the backbone area, this value must be the router ID of an ABR (in the same area) needing a virtual link to the backbone area as a substitute for a direct physical connection.

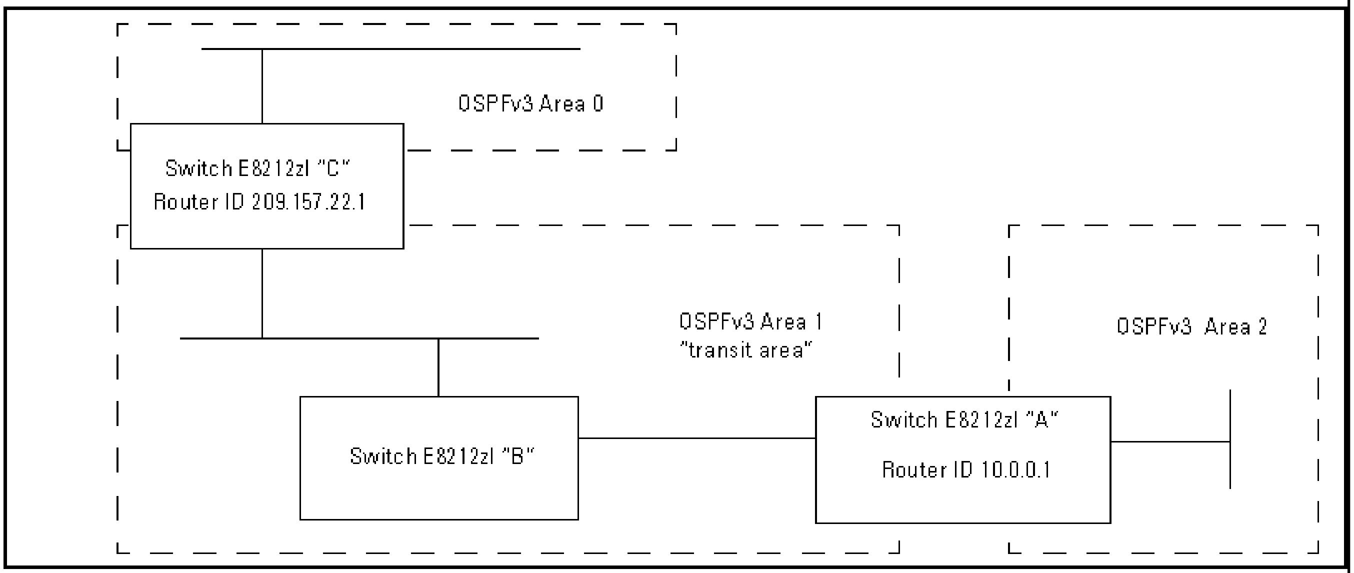
On the ABR that needs the virtual link to the backbone area, this value must be the router ID of the ABR (in the same area) having a direct physical connection to the backbone area.

**Configuring a virtual link**

The following figure shows an OSPFv3 ABR, routing switch "A" that lacks a direct connection to the backbone area (area 0). To provide backbone access to routing switch "A," you can add a virtual link between routing switch "A" and routing switch "C," using area 1 as a transit area.

To configure the virtual link, define it on the routers that are at each end of the link. No configuration for the virtual link is required on the other routers on the path through the transit area (such as routing switch "B" in this example).

**Figure 24:** *Defining OSPFv3 virtual links within a network*




To configure the virtual link on routing switch "A," enter the following command specifying the area 1 interface on routing switch "C":

```
Switch(ospf3)# area 1 virtual-link 209.157.22.1
```

To configure the virtual link on routing switch "C," enter the following command specifying the area 1 interface on routing switch "A."

```
Switch(ospf3)# area 1 virtual-link 10.0.0.1
```

## Adjusting a dead interval on a virtual link

This section describes the command to be used for adjusting a dead interval on a virtual link.

## area virtual-link dead-interval

### Syntax

`area [area-id ] virtual-link [router-id] dead-interval [1 - 65535]`

`no area [area-id ] virtual-link [router-id] dead-interval [1 - 65535]`

### Description

In the **ospf3** context, this command is used on both ABRs in a virtual link to change the number of seconds that a neighbor router waits for a hello packet from the specified interface before declaring the interface "down." This should be some multiple of the Hello interval.

The `dead-interval` setting must be the same on both ABRs on a given virtual link.

The `no` version restores the default value.

### Options

***area-id***

Specifies the OSPFv3 area in which both ABRs in a given virtual link operate. In this use, the area ID is sometimes termed "transit area ID."

This value must be the same for both ABRs in the virtual link. If the area does not exist, this command creates it.

***router-id***

For an ABR in a given virtual link, this is the router ID (in decimal or 32-bit dotted decimal format) used to create the link on that ABR.

Default: 40 seconds

# adjust hello interval

### Syntax

`area [area-id] virtual link [router-id] hellointerval [1 - 65535]`

### Description

In the **ospf3** context, used on both ABRs in a virtual link to indicate the length of time between the transmission of hello packets between the ABRs on opposite ends of the virtual link. The hello-interval setting must be the same on both ABRs on a given virtual link.

### Options

***area-id***

Specifies the OSPFv3 area in which both ABRs in a given virtual link operate. In this use, the area ID is sometimes termed "transit area ID". This value must be the same for both ABRs in the virtual link. If the area does not exist, this command creates it.

***router-id***

For an ABR in a given virtual link, this is the router ID (in decimal or 32-bit dotted decimal format) used to create the link on that ABR.

The `no` version restores the default value.

Default: 10 seconds

### Example input

To change the hello-interval on the virtual link configured for the network in **Figure 24: Defining OSPFv3 virtual links within a network** on page 205 to 60 seconds:

- On routing switch “A” (router ID 10.0.0.1) you would use the following command to reconfigure the current hello-interval to 60 seconds: `Switch(ospf3)# area 1 virtual-link 209.157.22.1 hellointerval 60`
- On routing switch “C” (router ID 209.157.22.1) you would use the following command to reconfigure the current hello-interval to 60 seconds`Switch(ospf3)# area 1 virtual-link 10.0.0.1 hello-interval 60`

## adjust retransmit interval

**Syntax**

`area [`*`area-id`*`] virtual link [`*`router-id`*`] retransmit-interval [`*`1 - 1800`*`]`

**Description**

In the **ospf3** context, used on both ABRs in a virtual link to change the number of seconds between LSA retransmissions on the virtual link.

The `retransmit-interval` setting must be the same on both ABRs on a given virtual link. This value is also used when retransmitting database description and link-state request packets.

**Options**

***`area-id`***

Specifies the OSPFv3 area in which both ABRs in a given virtual link operate. In this use, the area ID is sometimes termed "transit area ID."

This value must be the same for both ABRs in the virtual link. If the area does not exist, this command creates it.

***`router-id`***

For an ABR in a given virtual link, this is the router ID (in decimal or 32-bit dotted decimal format) used to create the link on that ABR.

This value is the router ID of the opposite router in the virtual link.

***`1 - 1800`***

Specifies the retransmit interval.

The `no` version of the command restores the default value.

Default: 5 seconds

**Example input**

```
switch(config)# router ospf3 area 1 virtual-link 1.1.1.1 retransmit-interval
```

## adjust transit delay

**Syntax**

`area [`*`area-id`*`] virtual-link [`*`router-id`*`] transit-delay [`*`1- 1800`*`]`

**Description**

In the **ospf3** context, used on both ABRs in a virtual link to change the estimated number of seconds it takes to transmit a link state update packet over a virtual link. The `transit-delay` setting must be the same on both ABRs on a given virtual link.

**Options**

***`area-id`***

Specifies the OSPFv3 area in which both ABRs in a given virtual link operate. In this use, the area ID is sometimes termed "transit area ID."

This value must be the same for both ABRs in the virtual link. If the area does not exist, this command creates it.

***router-id***

For an ABR in a given virtual link, this is the router ID (in decimal or 32-bit dotted decimal format) used to create the link on that ABR. This value is the router ID of the opposite router in the virtual link.

The `no` version of the command restores the default value.

## ipv6 ospf3 passive

### Syntax

`ipv6 ospf3 passive`

`no ipv6 ospf3 passive`

### Description

**VLAN** context command for enabling or disabling passive OSPFv3 operation on the VLAN.

The `no` option returns the VLAN interface to active OSPFv3 operation.

Default: OSPFv3 active

### Example output

This example shows how to configure an OSPF3v3 interface as passive. Enter this command in the **VLAN** context:

```
Switch(vlan-1)# ipv6 ospf3 passive
```

To display the OSPFv3 passive information, enter the command shown in below:

**`show ipv6 ospf3 interface` command with passive configured on an interface**

```
Switch(config)# show ipv6 ospf3 interface

 OSPFv3 configuration and statistics for interfaces

  Interface  Status   Area ID   State   Cost  Pri   Passive
  ---------- -------- --------- ------- ----- ----- -------
  vlan-55    Enabled  0.0.0.1   BDR     1     1     No
  vlan-75    Enabled  0.0.0.3   BDR     1     1     Yes
```

**`show ipv6 ospf3 interface` command for a specific VLAN with passive configured on an interface**

You can display the OSPFv3 passive information for a particular VLAN: suppose that a routing switch has OSPFv3 configured on VLAN 75. The following example shows a detailed output for VLAN 75 alone.

```
Switch(config)# show ipv6 ospf3 interface  VLAN 75 detail

 OSPFv3 configuration and statistics for VLAN 75

  Interface      : vlan-75  Status                    : Enabled
  Area ID        : 0.0.0.3  State                     : WAIT
  Priority       : 1        Cost                      : 1
  Type           : BCAST    Passive                   : Yes
  Hello Interval : 10       Dead Interval             : 40
  Transit Delay  : 1        Retransmit Interval       : 5
  Events         : 0        Designated Router         : 15.1.1.2
  Neighbors      : 1        Backup Designated Router : 15.1.4.4
```

## show ipv6 ospf3 virtual-link

**Syntax**

show ipv6 ospf3 virtual-link[ *rtr-id* ][ area *area-id* ]

**Description**

Displays OSPFv3 information learned about all virtual links detected by the routing switch.

**Options**

***rtr-id***

Displays virtual link information for a specific virtual-neighbor router detected by the routing switch.

**area *area-id***

Displays information learned from a virtual neighbor detected in a specific area.

**Example output**

This example displays output for all virtual links detected on the routing switch:

```
Switch# show ipv6 ospf3 virtual-link

 OSPFv3 Virtual Interface Status

 Transit AreaID  Neighbor Router Interface State
 --------------- --------------- ---------------
 0.0.0.1         1.0.0.4         P2P
```

**Display output for a specific virtual link**

```
Switch# show ipv6 ospf3 virtual-link 1.0.0.4
 Transit AreaID  : 0.0.0.1
 Neighbor Router : 1.0.0.4

 Interface State : P2P                    Transit Delay   : 1
 Events          : 1                      Rtr Interval    : 5
                                          Hello Interval  : 10
                                          Dead Interval   : 40
```

## show ipv6 ospf3 virtual-neighbor

**Syntax**

show ipv6 ospf3 virtual-neighbor[ *rtr-id* ][ area *area-id*]

**Description**

Displays OSPFv3 information learned about all virtual neighbor routers detected by the routing switch.

**Options**

**rtr-id**

Displays information for a specific virtual-neighbor router detected by the routing switch.

**area *area-id***

Displays information learned from a virtual neighbor detected in a specific area.

**Example output**

This example displays output for all virtual neighbors detected on the routing switch:

```
Switch# show ipv6 ospf3 virtual-neighbor

 OSPFv3 Virtual Interface Neighbor Information

 Router ID     State    IPv6 Addr                 Events
 ------------- -------- ------------------------- ---------
 1.0.0.4       FULL     2620:e::55:2              7
```

**Display output for a specific virtual neighbor**

```
Switch# show ipv6 ospf3 virtual-neighbor 1.0.0.4
 Router ID : 1.0.0.4
 State     : FULL

 IPv6 Addr : 2620:e::55:2
 RtQLen    : 5
 Events    : 7
```

## OSPFv3 passive

OSPFv3 sends LSAs to all other routers on the same VLAN interface. With OSPFv3 configured as passive on a VLAN interface, the routing switch is identified as a route in the OSPFv3 domain, but does not form an adjacency to any other router and does not send or receive OSPFv3 traffic on the subject VLAN interface. (A VLAN configured as passive operates similar to a VLAN connected to a stub network and does advertise the interface as a stub link into OSPFv3.)

Up to 128 active interfaces and a combined total of 512 active and passive interfaces are supported on the routing switch.

## router ospf3 redistribute

This step enables ASBR operation on a routing switch and must be executed on each routing switch connected to external routes you want to redistribute in your OSPFv3 domain.

**Syntax**

`router ospf3 redistribute [connected|static|ripng] route-map` *`map-name`*

`no router ospf3 redistribute [connected|static|ripng] route-map` *`map-name`*

**Description**

Executed on an ASBR to permit or deny redistribution of static and/or connected routes to the ASBR’s domain, as specified in the named route-map.

The `no` form removes the redistribution configuration for the specified route-map.

**Options**

**`static`**

Redistribute static routes into OSPFv3.

**`connected`**

Redistribute connected routes into OSPFv3.

**`ripng`**

Redistribute RIPng routes into OSPFv3.

**Example input**

Enabling route redistribution

To implement redistribution for the connected and static routes configured in the route-map named "mymap," you would execute the following commands on the applicable ASBR:

```
Switch(config)# router ospf3 redistribute connected
Switch(config)# router ospf3 redistribute static
Switch(config)# router ospf3 redistribute ripng
```

## Command to modify the default metric for redistribution

**Syntax**

`router ospf3 default-metric [`*`0 - 16777215`*`]`

`no router ospf3 default-metric`

**Description**

Globally assigns the cost metric to apply to all external routes redistributed by the ASBR. By using different cost metrics for different ASBRs, you can prioritize the ASBRs in your AS. Default: 10

**Example input**

Modifying the default metric for redistribution. To assign a default metric of 4 to all routes imported into an OSPFv3 domain through an ASBR, enter the following command in the ASBR:

```
Switch(config)# router ospf3 default-metric 4
```

## Command to modify the redistribution metric type

The redistribution metric type is used by default for all routes imported into OSPFv3. Type 1 metrics are the same "units" as internal OSPFv3 metrics and can be compared directly. Type 2 metrics are not directly comparable and are treated as larger than the largest internal OSPFv3 metric.

**Syntax**

`router ospf3 default-metric-type [type1|type2]`

**Description**

Globally reconfigures the redistribution metric type on an ASBR.

**Options**

**`type1`**

Specifies the OSPFv3 metric plus the external metric for an external route.

**`type2`**

Specifies the external metric for an external route.

Default: type2

**Example input**

Modifying the redistribution metric type. To change from the default setting on an ASBR to type 1, enter the following command:

```
Switch(config)# router ospf3 default-metric-type type1
```

## Redistributing/Assigning Loopback IPv6 address to OSPFv3

Enter the `redistribute connected` command as described in **router ospf3 redistribute**.

In the following configuration, loopback interface 6 is configured with IPv6 address 2001:db8:1::127 and is assigned to OSPFv3 area 0.0.0.1, and thus is advertised as an OSPFv3 IntraArea route, regardless of whether route redistribution is enabled.

In the same configuration, loopback interface 2 is configured with IPv6 address 2001:db8:2:133 but is not assigned to an OSPFv3 area. As a result, it will be advertised to neighbors as an External route, and only if it is a "permitted" route in a route-map invoked by the `redistribute` command.

**Assigning loopback IPv6 addresses to OSPFv3 areas**

```
Switch(config)# interface loopback 6
Switch(lo-6)# ipv6 address 2001:db8:1:127
Switch(lo-6)# ipv6 ospf3 area 1
Switch(lo-6)# interface loopback 2
Switch(lo-2)# ipv6 address 2001:db8:2:133 1
Switch(lo-2)# exit
```

[1]Assigns an IPv6 address to loopback interface 2, but does not assign the interface to an OSPFv3 area

## Verifying the OSPFv3 redistribution of loopback interfaces

Enter the following command from a neighboring router to display the IPv6 route table entries for detected OSPFv3 routes.

### show ipv6 route ospf3

**Syntax**

```
show ipv6 route ospf3
```

**Example output**

Verifying OSPFv3 redistribution of loopback interfaces on a neighboring router

```
Switch(config)# show ipv6 route ospf3

                               IPv6 Route Entries

 T(Type):
  S: Static  C: Connected  O: OSPFv3  R: RIPng

 ST(Sub-type):
  O : OSPF Intra  E1:External1  N1:NSSA Ext1
  OI: OSPF Inter  E2:External2  N2:NSSA Ext2

 Destination/
  Gateway                                           T     ST     Distance
Metric
-------------------------------------------------------------------------------------
--
 121::2/128
 fe80::3ea8:2aff:fe3f:8200%vlan1001                 O     O      110          1
 2000::/64
 fe80::328d:99ff:fe4c:543f%vlan202                  O     O      110          2
 2000::/64
 fe80::328d:99ff:fe56:8300%vlan202                  O     O      110          2
 2001::/64
 fe80::328d:99ff:fe4c:543f%vlan202                  O     E2     110          10
 2003::/64
 fe80::328d:99ff:fe4c:543f%vlan202                  O     E2     110          10
 2004::/64
 fe80::328d:99ff:fe4c:543f%vlan202                  O     E2     110          10
 2005::/64
 fe80::328d:99ff:fe4c:543f%vlan202                  O     E2     110          10
```

```
2540::/64
fe80::328d:99ff:fe4c:543f%vlan202             O    O     110        2
2541::/64
fe80::328d:99ff:fe4c:543f%vlan202             O    E2    110        10
2542::/64
fe80::328d:99ff:fe4c:543f%vlan202             O    O     110        2
2544::/64
fe80::328d:99ff:fe4c:543f%vlan202             O    E2    110        10
3001::/64
fe80::328d:99ff:fe4c:543f%vlan202             O    E2    110        10
5001::/64
fe80::328d:99ff:fe4c:543f%vlan202             O    E2    110        10
5101::/64
```

## Reducing AS-external-LSAs and inter-area-prefix-LSAs

An OSPFv3 ASBR uses AS-external-LSAs to originate advertisements of a route to another routing domain. These advertisements are:

- Flooded in the area in which the ASBR operates.
- Injected into the backbone area and then propagated to any other OSPFv3 areas (except stub and NSSA areas) within the local OSPFv3 AS. If the AS includes an NSSA, there are two additional options:
  - If the NSSA includes an ASBR, you can suppress advertising some or all of its summarized external routes into the backbone area.
  - Replace all inter-area-prefix-LSAs and all external routes from the backbone area with the default route (::/0).

### Algorithm for AS-external-LSA reduction

The AS-external-LSA reduction feature behavior changes under the following conditions:

- There is one ASBR advertising (originating) a route to the external destination, but one of the following happens:
  - A second ASBR comes online.
  - A second ASBR that is already online begins advertising an equivalent route to the same destination.

  In either case above, the routing switch with the higher router ID floods the AS-external-LSAs and the other routing-switch flushes its equivalent AS-external-LSAs.
- One of the ASBRs starts advertising a route that is no longer equivalent to the route the other ASBR is advertising. In this case, the ASBRs each flood AS-external-LSAs. Since the LSAs either no longer have the same cost or no longer have the same next-hop router, the LSAs are no longer equivalent, and the LSA reduction feature no longer applies.
- The ASBR with the higher router ID becomes unavailable or is reconfigured so that it is no longer an ASBR. In this case, the other ASBR floods the AS-external-LSAs.

### About replacing inter-area-prefix-LSAs and type-7-external-LSA default routes with an AS-external-LSA default route

By default, a routing switch operating as an ABR for a stub area or NSSA injects non-default, inter-area routes (inter-area-prefix-LSAs) into the stub areas and NSSAs. For NSSAs, the routing switch also injects a type-7-LSA default external route. You can further reduce LSA traffic into these areas by using `no-summary`. This command option configures the routing switch to:

- Replace injection of inter-area-prefix-LSAs into a stub area or NSSA with an inter-area-prefix-LSA default summary route (::/0).
- Replace injection of all external routes into an NSSA with an inter-area-prefix-LSA default route.

You can enable this behavior when you first configure the stub area or NSSA, or at a later time.

The no-summary command does not affect intra-area advertisements, meaning the switch still accepts summary LSAs from OSPFv3 neighbors within its area and floods them to other neighbors. The switch can form adjacencies with other routers regardless of whether summarization is enabled or disabled for areas on each switch.

When you use no-summary, the change takes effect immediately. If you apply the option to a previously configured area, the switch flushes all of the summary LSAs it has generated (as an ABR) from the area.

NOTE

This feature applies only when the routing-switch is configured as an ABR for a stub area or NSSA. To completely prevent summary LSAs from injection into the area, use no-summary to disable the summary LSAs on each OSPFv3 router that is an ABR for the area.

To implement the above operation for a stub area or NSSA, enter a command such as the following:

```
Switch(ospf3)# area 40 stub metric-cost 3 no-summary
```

### Enforcing strict LSA operation for graceful restart helper mode

OSPFv3 operation on the routing switches includes helper mode operation for graceful restart of OSPFv3 on a neighboring router upon receipt of a "grace LSA" from the neighbor. In the default configuration, helper mode operation in this case includes terminating graceful restart support ("strict LSA" operation) on the routing switch if it detects a topology change requiring updated LSAs during the restart period of the neighboring router. Terminating this support forces the helper routing switch to re-establish its LSAs and OSPFv3 functions on the network segment affected by the OSPFv3 restart on the neighboring router. (For more information on OSPFv3 graceful restart, see RFC 3623.)

In the default OSPFv3 configuration, the default helper mode operation terminates graceful restart if topology changes affect the network segment.

NOTE

Configure router-id or IPv4 loopback address for OSPFv3 Non-Stop Forwarding to work on the switch.

## route-map name

Use the route-map command to enter the **route-map** context and configure one or more route-maps.

**Syntax**

route-map *name* [permit|deny ][ seq *1 - 4294967295* ]

no route-map *name* [permit|deny ][ seq *1 - 4294967295* ]

**Description**

Used in the **OSPFv3** context (router-ospf3) of a routing switch operating as an ASBR. This command enters the **route-map** context and enables configuration of one or more route-maps for permitting or denying external connected or static routes.

The no form of the command removes the named route-map from the switch configuration.

For details on configuring route-maps, including several commands used in the **route-map** context, see "Route Maps" in the "IP Routing Features" chapter of the latest *Multicast and Routing Guide* for your routing switch.

**Example input**

To permit the content of a route-map named "mymap" with a sequence number of 100 on a routing switch operating as an ASBR, enter the following command in the **global config** context:

```
Switch(config)# route-map mymap permit seq 100
Switch(route-map-mymap-10)_
```

## router ospf3 trap

**Syntax**

`router ospf3 trap <tab>`

**Description**

This command displays the router OSPF3 traps.

**Options**

**all-traps**

Enable all the OSPFv3 traps.

**`disable`**

Disable OSPFv3 traps.

**`enable`**

Enable OSPFv3 traps.

**`interface-state-change`**

Send a trap when the state of a non-virtual interface changes.

**`virtual-interface-state-change`**

Send a trap when the state of a virtual interface changes.

**`neighbor-state-change`**

Send a trap when the state of a non-virtual neighbor changes.

**`virtual-neighbor-state-change`**

Send a trap when the state of a virtual neighbor changes.

**`interface-config-error`**

Send a trap when a configuration conflict occurs for a non-virtual interface.

**`virtual-interface-config-error`**

Send a trap when a configuration conflict occurs for a virtual interface.

**`interface-receive-bad-packet`**

Send a trap when an invalid packet is received on a non-virtual interface.

**`virtual-interface-receive-bad-packet`**

Send a trap when an invalid packet is received on a virtual interface.

**`restart-status-change`**

Send a trap when the graceful restart state changes for the device.

**`nbrrestart-helper-status-change`**

Send a trap when the graceful restart helper state changes for the neighbor.

**`virtual-nbrrestart-helper-status-change`**

Send a trap when the graceful restart helper state changes for the virtual neighbor.

**`interface-retransmit-packet`**

[Deprecated] A packet has been retransmitted on a non-virtual interface.

**virtual-interface-retransmit-packet**

[Deprecated] A packet has been retransmitted on a virtual interface.

**originate-lsa**

[Deprecated] A new LSA has been sent.

**max-age-lsa**

[Deprecated] An LSA expired as its maximum age has been reached.

**unknown-lsa**

[Deprecated] An LSA with unknown code has been received.

**lsdb-overflow**

[Deprecated] Send a trap when number of LSAs in router's lsdb has exceeded the limit.

**lsdb-approach-overflow**

[Deprecated] Send a trap when number of LSAs in router's lsdb has exceeded ninety percent of the limit.

**asbr-status-change**

[Deprecated] Send a trap when ASBR status changes for the device.

**abr-status-change**

[Deprecated] Send a trap when ABR status changes for the device.

## Troubleshooting: Logging neighbor adjacency change events

**Cause**

In the default configuration, the routing switch generates event log messages to indicate neighbor adjacency changes during initialization and normal operation. This enables OSPFv3 misconfiguration troubleshooting, while producing a lower volume of event log messages than is seen with the debug troubleshooting option. Both a standard (default) mode and an optional detail mode are provided. Using the optional `debug destination` command, the logging output can be directed to a syslog server or a terminal. For more information on `debug`, see the latest *Aruba-OS Switch Management and Configuration Guide* for your routing switch.

### logging neighbor-adjacency

**Syntax**

`logging neighbor-adjacency [ detail ]`

**Description**

Used in the **ospf3** context to enable logging of standard or detailed adjacency changes. In the default configuration, logs OSPFv3 neighbor changes into or out of the full adjacency state.

**Options**

**detail**

Generates event log messages for all OSPFv3 neighbor adjacency state changes.

**no logging neighbor-adjacency**

Disables logging neighbor adjacency on the routing switch.

**no logging neighbor-adjacency detail**

Cancels detailed neighbor adjacency change logging and returns the routing switch to logging only neighbor changes into our out of full adjacency.

Default: Standard full adjacency changes logging enabled.

The neighbor-adjacency event log messages are described in the latest *Event Log Reference Guide* for your routing switch. In the event log output, neighbors are identified by router ID.

**Example output**

This example shows the neighbor-adjacency change logging using `show log -r OSPF3` command.

```
Switch(ospf3)# show log -r OSPF3:
 Keys: W=Warning     I=Information
       M=Major       D=Debug E=Error
---- Reverse event Log listing: Events Since Boot ----
e 05/01/10 15:21:09 02809 OSPF3: ADJCHG: Neighbor 15.255.155.1 on interface
            vlan-22 moved to Down state, Inactivity Timer
e 04/27/10 14:36:48 02809 OSPF3: ADJCHG: Neighbor 10.10.10.45 on interface
            vlan-11 moved to Full state, Loading Done
```

```
Switch(ospf3)# show log -r OSPF3:
 Keys: W=Warning     I=Information
       M=Major       D=Debug E=Error
---- Reverse event Log listing: Events Since Boot ----
e 05/01/10 15:21:09 02809 OSPF3: ADJCHG: Neighbor 15.255.155.1 on interface
            vlan-22 moved to Down state, Inactivity Timer
e 04/27/10 14:36:48 02809 OSPF3: ADJCHG: Neighbor 10.10.10.45 on interface
            vlan-11 moved to Full state, Loading Done
```

# Commands to activate OSPFv3

After either an IPv4 address or a router ID has been configured on the routing switch, OSPFv3 activates when enabled with the following two commands:

```
Switch(config): ipv6 unicast-routing
```

```
Switch(config): router ospf3 enable
```

NOTE

The `router ospf3 enable` command enables OSPFv3 without a system reset.

# Commands to configure OSPFv3 on the routing switch

This section describes the commands to configure OSPFv3 on the routing switch.

The maximum number of active OSPF interface limits is 8.

## ip router-id

**Syntax**

`ip router-id <ip-addr>`

**Options**

**`ip router-id <ip-addr>`**

Executed at the global configuration level to assign a router ID to the routing switch.

Default: Disabled

**Example output**

```
Switch(config)# ip router-id 0.0.0.1
```

CLI to verify the set value:

```
OSPF General Status
 OSPF protocol                        :enabled
 Router ID                            :0.0.0.1
 RFC 1583 compatability               :compatible

 Intra-area distance                  :110
 Inter-area distance                  :110
 AS-external distance                 :110

 Default import metric                :10
 Default import metric type           :external type 2

 Area Border                          :yes
 AS Border                            :no
 External LSA Count                   :512
 External LSA Checksum Sum            :16790016
 Originate New LSA Count              :19
 Receive New LSA Count                :529

 Graceful Restart Interval            :120
```

## ipv6 unicast-routing

**Syntax**

`ipv6 unicast-routing`

`no ipv6 unicast-routing`

**Description**

Executed at the global configuration level to enable IPv6 routing on the routing switch.

**Usage**

Default: Disabled

The `no` form disables IPv6 routing. (Global OSPFv3 routing must be disabled before you disable IPv6 routing.)

**Example input**

Enabling IPv6 Routing

```
Switch(config)# ipv6 unicast-routing
```

## router ospf3 [enable|disable]

**Syntax**

`router ospf3 [enable|disable]`

**Description**

The `router ospf3` command executed alone puts the routing switch into **ospf3** context. The keyword options enable or disable OSPFv3 on the routing switch.

**Usage**

This command allows you to configure OSPFv3 before activating it on the routing switch. Global `IPv6 unicast-routing` must be enabled before executing this command.

NOTE

If you disable OSPFv3, the switch retains all the configuration information for the disabled protocol in flash memory. If you subsequently restart OSPF, the existing configuration will be applied.

**Example input**

Enable global OSPFv3 routing

```
Switch(config)# router ospf3 enable
Switch(ospf3)#
```

## About assigning the routing switch to OSPFv3 areas

After you globally enable OSPFv3 on the routing switch, use this command to create one or more OSPFv3 areas within your autonomous system (AS).

A routing switch can belong to one area or to multiple areas. Participation in a given area requires configuring one or more VLANs and assigning each to the desired area.

- If you want the VLANs configured on the routing switch to all reside in the same area, you need to configure only that one area. (In this case, the routing switch would operate as an internal router for the area.)
- If you want to put different VLANs on the routing switch into different areas, you need to re-execute this command for each area. (In this case, the routing switch operates as an ABR for each of the configured areas.)

NOTE

Each ABR must be either directly connected to the backbone area (0) or be configured with a virtual link to the backbone area through another ABR that is directly connected to the backbone area.

### area [ospf3–area-id|backbone][ normal ]

**Syntax**

`area [`*`ospf3-area-id`*`|backbone][ normal ]`

**Description**

After using `router ospf3` to globally enable OSPFv3 and enter the global **OSPF3** context, execute this command to assign the routing switch to a backbone or other normal area.

**Options**

***ospf-area-id***

Specifies a normal area to which you are assigning the routing switch. You can assign the routing switch to one or more areas, depending on the area in which you want each configured VLAN or subnet to reside.

You can enter area IDs in either whole number or dotted decimal format. (The routing switch automatically converts whole numbers to the dotted decimal format.)

For example, if you enter an area-ID of **1**, it appears in the switch's configuration as **0.0.0.1**, and an area-ID of 256 appears in the switch configuration as **0.0.1.0**. Entering an area ID of **0** or **0.0.0.0** automatically joins the routing switch to the backbone area.

The maximum area ID value is 255.255.255.254 (4,294,967,294).

**backbone**

Assigns the routing switch to the backbone area and automatically assigns an area ID of **0.0.0.0** and an area type of **normal**Using **0** or **0.0.0.0** with the above `ospf3-area-id` option achieves the same result.

The backbone area is automatically configured as a "normal" area type.

**normal**

Applied by default if not specified in an area command. Required to convert an existing NSSA or stub area to a normal area.

**Usage**

The `no` form of the command removes the routing switch from the specified area.

Default: No areas. Range: *1 - 16* areas (of all types)

**Example input**

Configuring an OSPFv3 backbone or normal area

To configure a backbone and a normal area with an ID of "1" (0.0.0.1) on a routing switch:

```
Switch(ospf3)# area backbone
Switch(ospf3)# area 1
```

To convert an existing NSSA or stub area to a normal area, you would include the `normal` keyword. For example, if area 10 was configured as an NSSA area you wanted to convert to a normal area, you would use the following command:

```
Switch(ospf3)# area 10 normal
```

## area ospf3–area-id stub/area ospf3–area id nssa

**Syntax**

`area` *`ospf3-area-id`* `stub [ metric-cost` *`0 - 16777215`* `][ no-summary ]`

`area` *`ospf3-area-id`* `nssa [metric-cost` *`0 - 16777215`* `][no-summary]`

**Options**

***ospf3-area-id***

Same area ID as in **area [ospf3–area-id|backbone][ normal ]** on page 219, except you cannot assign a backbone area number (**0** or **0.0.0.0**) to a stub or NSSA area.

**stub/nssa**

Designates the area identified by *ospf3-area-id* as a stub area or NSSA.

**metric-cost *0 - 16777215***

If the routing switch is used as an ABR for the designated area, assigns the cost of the default route (to the backbone) that is injected into the area.

NOTE: If the routing switch is not an ABR for a stub area or NSSA, the above cost setting is still allowed, but is not used.

In the default configuration, a routing switch acting as an ABR for a stub area or NSSA injects type-7-LSA default routes into the area. If `no-summary` is configured on the ABR, it injects inter-area-prefix-LSA routes into the area.

**metric-type [type1|type2]**

Used in NSSA ABRs only. Specifies the type of external cost metric to include in type-7-LSAs advertised for redistribution of external routes in the NSSA.

The `metric-type` command specifies whether to include the redistribution cost in the cost metric calculation for a type-7-LSA default route injected into the area.

`type1` : Calculate external route cost for a type-7-LSA default route as the sum of (1) the external route cost assigned by the ASBR plus (2) the internal cost from the router with traffic for the external route to the ASBR advertising the route.

`type2` : Use the external route cost assigned by the ASBR advertising the route.

Default: Enabled with metric-type type2.

NOTE: Different routers in the NSSA can be configured with different metric-type values.

**`no-summary`**

Where the routing switch is an ABR for a stub area or an NSSA, this option reduces the amount of LSA traffic entering the area from the backbone by replacing the injection of inter-area-prefix-LSA routes and type-7-LSA default external routes with injection of an inter-area-prefix-LSA default route.

Default: Disabled

Using `no area` *`ospf3-area-id`* `nssa no-summary` resets the routing switch to the state where injection of inter-area-prefix-LSA routes and the type-7-LSA default external routes is enabled with metric-type set to `type2`.

**Example input**

Creating stub area and NSSA assignments

The following examples of configuring a stub area and an NSSA on a routing switch use an (arbitrary) cost of "15."

```
Switch(ospf3)# area 2 stub metric-cost 15
```

Assigns a stub area with a cost of 15.

```
Switch(ospf3)# area 3 nssa metric-cost 15
```

Assigns an NSSA with a cost of 15 and, by default, uses a Network-LSA default cost metric for Type-7-LSA (external) routes received from the backbone.

```
Switch(ospf3)# area 4 nssa metric-cost 15 no-summary
```

Assigns an NSSA with a cost of 15, blocks injection of Inter-Area-Prefix- LSA routes, and starts injection of Inter-Area-Prefix-LSA default routes from the backbone.

```
Switch(ospf3)# area 5 nssa metric-cost 15 metric-type type1
```

Sets the cost metric type for Type-7- LSA default routes injected into the NSSA.

## Command to enable OSPFv3 on an interface and assigning one or more VLANs to each area

After you define an OSPFv3 area, you can assign one or more VLANs to it. When a VLAN is assigned to an area, all currently configured IPv6 addresses in the VLAN are automatically included in the assignment.

NOTE: All static VLANs configured on a routing switch configured for OSPFv3 must be assigned to one of the defined areas in the AS.

### vlan vid ipv6 ospf3

**Syntax**

`vlan` *`vid`* `ipv6 ospf3 area [` *`ospf3-area-id`* `]`

```
vlan vid ipv6 ospf3
interface tunnel [tunnel-id ] ipv6 ospf3 area [ospf3-area-id]
interface tunnel [tunnel-id ] ipv6 ospf3 [area [ospf3-area-id]]
```

#### Description

Executed in a specific **VLAN** context to assign the VLAN to the specified area. If `area` is not specified, the command defaults to the backbone area. Requires that the area is already configured on the routing switch. This command assigns all configured networks in the VLAN to the specified OSPFv3 area.

#### Options

**`vlan vid`**

Defines the **VLAN** context for executing the area assignment.

**`interface tunnel tunnel-id`**

Defines the **tunnel** context for executing the area assignment.

**`area ospf3-area-id`**

Identifies the OSPFv3 area to which the VLAN should be assigned.

NOTE: If you add a new IPv6 address to a VLAN after assigning the VLAN to an OSPFv3 area, the new network automatically joins the area.

Before adding a new VLAN to an area, you must enable IPv6 on the VLAN. Otherwise the following CLI message appears:

```
IPV6 should be enabled before configuring OSPFv3.
```

The `no` form deletes the OSPFv3 configuration from the specified VLAN.

#### Example input

To assign VLAN 8 on a routing switch to area 3 and include all IP addresses configured in the VLAN, enter the following commands:

```
Switch(ospf3)# vlan 8
Switch(vlan-8)# ipv6 ospf3 area 3
```

## Command to assign IPv6 loopback addresses to an area

After you define the OSPFv3 areas to which the switch belongs, you have the option to assign user-defined IPv6 loopback addresses to an area. An IPv6 loopback interface is configured with an IPv6 address that is unique in an AS and is always reachable as long as at least one of the IPv6 interfaces on the routing switch is operational.

### interface loopback

#### Syntax

```
interface loopback [0 - 7] ipv6 ospf3 area [ospf3-area-id|backbone]
no interface loopback [0 - 7 ] ipv6 ospf3
```

#### Description

Executed in a specific **loopback** context to assign an IPv6 loopback interface to the specified area. Requires that the loopback interface is already configured with an IPv6 address on the routing switch.

#### Options

**loopback interface *0 - 7***

Defines the **loopback** context for executing the area assignment

**ipv6 ospf3 area *ospf3-area-id***

Identifies the OSPFv3 area to which the loopback interface is assigned.

NOTE

The area must already exist, and the loopback interface must already be configured with a minimum of one IPv6 address.

An IPv6 loopback interface can be assigned to only one area at any time.

When an IPv6 loopback interface is assigned to a given area, the `no` form removes the interface from that area.

**Example input**

Assigning IPv6 loopback addresses to an area

```
Switch(config)# interface loopback 3
Switch(lo-3)# ipv6 ospf3 area 12
```

# route-map name

Use the `route-map` command to enter the **route-map** context and configure one or more route-maps.

**Syntax**

route-map *name* [permit|deny ][ seq *1 - 4294967295* ]

no route-map *name* [permit|deny ][ seq *1 - 4294967295* ]

**Description**

Used in the **OSPFv3** context (`router-ospf3`) of a routing switch operating as an ASBR. This command enters the **route-map** context and enables configuration of one or more route-maps for permitting or denying external connected or static routes.

The `no` form of the command removes the named route-map from the switch configuration.

For details on configuring route-maps, including several commands used in the **route-map** context, see "Route Maps" in the "IP Routing Features" chapter of the latest *Multicast and Routing Guide* for your routing switch.

**Example input**

Configuring route-maps

To permit the content of a route-map named "mymap" with a sequence number of 100 on a routing switch operating as an ASBR, enter the following command in the **global config** context:

```
Switch(config)# route-map mymap permit seq 100
Switch(route-map-mymap-10)_
```

# router ospf3 redistribute

This step enables ASBR operation on a routing switch and must be executed on each routing switch connected to external routes you want to redistribute in your OSPFv3 domain.

**Syntax**

router ospf3 redistribute [connected|static|ripng] route-map *map-name*

no router ospf3 redistribute [connected|static|ripng] route-map *map-name*

**Description**

Executed on an ASBR to permit or deny redistribution of static and/or connected routes to the ASBR's domain, as specified in the named route-map.

The `no` form removes the redistribution configuration for the specified route-map.

**Options**

**`static`**

Redistribute static routes into OSPFv3.

**`connected`**

Redistribute connected routes into OSPFv3.

**`ripng`**

Redistribute RIPng routes into OSPFv3.

**Example input**

Enabling route redistribution

To implement redistribution for the connected and static routes configured in the route-map named "mymap," you would execute the following commands on the applicable ASBR:

```
Switch(config)# router ospf3 redistribute connected
Switch(config)# router ospf3 redistribute static
Switch(config)# router ospf3 redistribute ripng
```

## Command to modify the default metric for redistribution

**Syntax**

`router ospf3 default-metric [0 - 16777215]`

`no router ospf3 default-metric`

**Description**

Globally assigns the cost metric to apply to all external routes redistributed by the ASBR. By using different cost metrics for different ASBRs, you can prioritize the ASBRs in your AS. Default: 10

**Example input**

Modifying the default metric for redistribution. To assign a default metric of 4 to all routes imported into an OSPFv3 domain through an ASBR, enter the following command in the ASBR:

```
Switch(config)# router ospf3 default-metric 4
```

## Command to modify the redistribution metric type

The redistribution metric type is used by default for all routes imported into OSPFv3. Type 1 metrics are the same "units" as internal OSPFv3 metrics and can be compared directly. Type 2 metrics are not directly comparable and are treated as larger than the largest internal OSPFv3 metric.

**Syntax**

`router ospf3 default-metric-type [type1|type2]`

**Description**

Globally reconfigures the redistribution metric type on an ASBR.

**Options**

**type1**

Specifies the OSPFv3 metric plus the external metric for an external route.

**type2**

Specifies the external metric for an external route.

Default: type2

**Example input**

Modifying the redistribution metric type. To change from the default setting on an ASBR to type 1, enter the following command:

```
Switch(config)# router ospf3 default-metric-type type1
```

## Redistributing/Assigning Loopback IPv6 address to OSPFv3

Enter the `redistribute connected` command as described in **router ospf3 redistribute** on page 223.

**Enabling redistribution of loopback IPv6 addresses**

In the following configuration, loopback interface 6 is configured with IPv6 address 2001:db8:1::127 and is assigned to OSPFv3 area 0.0.0.1, and thus is advertised as an OSPFv3 IntraArea route, regardless of whether route redistribution is enabled.

In the same configuration, loopback interface 2 is configured with IPv6 address 2001:db8:2:133 but is not assigned to an OSPFv3 area. As a result, it will be advertised to neighbors as an External route, and only if it is a "permitted" route in a route-map invoked by the `redistribute` command.

**Assigning loopback IPv6 addresses to OSPFv3 areas**

```
Switch(config)# interface loopback 6
Switch(lo-6)# ipv6 address 2001:db8:1:127
Switch(lo-6)# ipv6 ospf3 area 1
Switch(lo-6)# interface loopback 2
Switch(lo-2)# ipv6 address 2001:db8:2:133 1
Switch(lo-2)# exit
```

[1] Assigns an IPv6 address to loopback interface 2, but does not assign the interface to an OSPFv3 area

## Verifying the OSPFv3 redistribution of loopback interfaces

Enter the following command from a neighboring router to display the IPv6 route table entries for detected OSPFv3 routes.

### show ipv6 route ospf3

**Syntax**

```
show ipv6 route ospf3
```

**Example output**

Verifying OSPFv3 redistribution of loopback interfaces on a neighboring router

```
Switch(config)# show ipv6 route ospf3

                              IPv6 Route Entries

 T(Type):
  S: Static   C: Connected   O: OSPFv3   R: RIPng
```

```
 ST(Sub-type):
  O : OSPF Intra  E1:External1  N1:NSSA Ext1
  OI: OSPF Inter  E2:External2  N2:NSSA Ext2

 Destination/
  Gateway                                                   T     ST     Distance
Metric
-------------------------------------------------------------------------------------
--
 121::2/128
 fe80::3ea8:2aff:fe3f:8200%vlan1001                         O     O      110          1
 2000::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     O      110          2
 2000::/64
 fe80::328d:99ff:fe56:8300%vlan202                          O     O      110          2
 2001::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     E2     110          10
 2003::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     E2     110          10
 2004::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     E2     110          10
 2005::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     E2     110          10
 2540::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     O      110          2
 2541::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     E2     110          10
 2542::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     O      110          2
 2544::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     E2     110          10
 3001::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     E2     110          10
 5001::/64
 fe80::328d:99ff:fe4c:543f%vlan202                          O     E2     110          10
 5101::/64
```

## OSPFv3 redistribution of loopback addresses

When you assign an IPv6 address to a loopback interface on a routing switch, the address is listed as connected in the route table on that routing switch and is advertised to neighbors as described in the table below:

**Table 20:** *Route redistribution of loopback addresses*

| Loopback address assignment | Route redistribution enabled | Route redistribution disabled |
|---|---|---|
| Loopback **not assigned** to an OSPFv3 area | The loopback address is advertised to neighbors as an OSPFv3 **External** route. | The loopback address is not advertised to neighbors. |
| Loopback **is assigned** to an OSPFv3 area | The loopback address is advertised to neighbors in the same area as an OSPFv3 Intra-Area route. For all other areas it is advertised as an OSPFv3 Inter-Area route. | |

## Configuring for external route redistribution in an OSPFv3 domain

Configuring route redistribution for OSPFv3 establishes the routing switch as an ASBR (residing in a backbone, normal, or NSSA) for importing and translating different protocol routes from other IGP domains into an OSPFv3 domain. The switches support redistribution for static routes and directly connected routes. When you configure redistribution for OSPF, you can specify that static or connected routes external to the OSPFv3 domain are imported as OSPFv3 routes. The steps for configuring external route redistribution to support ASBR operation include the following:

**Procedure**

1. Optional: Configure route-maps to permit and/or deny route prefixes for redistribution in your OSPFv3 domain.
2. Enable route redistribution.
3. Optional: Modify the default metric for redistribution.
4. Optional: Modify the redistribution metric type.
5. Optional: Change the administrative distance setting.

NOTE

In the default configuration, redistribution is permitted for all routes from supported sources.

Enable redistribution after you have configured route-maps defining the route policies you want to apply to route redistribution in the OSPFv3 domain. Otherwise, your AS may become overloaded with routes that you did not intend to redistribute.

# Influencing route choices by changing the administrative distance default

## distance [external|inter-area|intra-area]

**Syntax**

`distance [external|inter-area|intra-area][ 1 - 255 ]`

**Description**

Used in the **OSPFv3** configuration context (`router ospf3`) to globally reconfigure the administrative distance priority for the specified route type. 1 is the highest priority; 255 is the lowest priority.

**Options**

**`external`** ***`1 - 255`***

Changes the administrative distance for routes between the OSPFv3 domain and other EGP domains.

**`inter-area`** ***`1 - 255`***

Changes the administrative distance for routes between areas within the same OSPFv3 domain.

**`intra-area`** ***`1 - 255`***

Changes the administrative distance for routes within OSPFv3 areas.

Default: 110; Range: *1 - 255*

**Example output**

```
switch(config)#router ospf3 distance external 150
switch(config)#router ospf3 distance inter-area 140
switch(config)#router ospf3 distance intra-area 130
```

CLI to verify the set value

```
Switch(ospf3)#show run router ospf3

Running configuration:

router ospf3
  area 0.0.0.1 virtual-link 100.1.1.2
  area 0.0.0.4
  area backbone
  distance intra-area 130
  distance inter-area 140
  distance external 150
  enable
  exit
```

# restart strict-lsa

### Syntax

`restart strict-lsa`

`no restart strict-lsa`

### Description

Used in the **OSPFv3** context to enable or disable strict LSA operation in a network segment for a neighboring router that is attempting a graceful restart.

When enabled, this operation halts helper mode support if a change in LSAs (topology change) is detected during the restart period of the neighbor.

Default: Strict LSA operation enabled

The `no` form disables strict LSA operation.

### Example input

```
Switch(config)#router ospf3 restart strict-lsa
```

## Enforcing strict LSA operation for graceful restart helper mode

OSPFv3 operation on the routing switches includes helper mode operation for graceful restart of OSPFv3 on a neighboring router upon receipt of a "grace LSA" from the neighbor. In the default configuration, helper mode operation in this case includes terminating graceful restart support ("strict LSA" operation) on the routing switch if it detects a topology change requiring updated LSAs during the restart period of the neighboring router. Terminating this support forces the helper routing switch to re-establish its LSAs and OSPFv3 functions on the network segment affected by the OSPFv3 restart on the neighboring router. (For more information on OSPFv3 graceful restart, see RFC 3623.)

In the default OSPFv3 configuration, the default helper mode operation terminates graceful restart if topology changes affect the network segment.

NOTE

Configure router-id or IPv4 loopback address for OSPFv3 Non-Stop Forwarding to work on the switch.

# Adjusting performance by changing the VLAN interface settings

## ipv6 ospf3 cost

### Syntax

```
ipv6 ospf3 cost [1 - 65535]
```

### Description

Used in the **VLAN** context to indicate the overhead required to send a packet across an interface. You can modify the cost to differentiate between 100 Mbps and 1000 Mbps (1 Gbps) links. This command assigns the specified cost to all networks configured on the VLAN.

Default: 1

### Example input

```
Switch(vlan-100)#ipv6 ospf3 cost 20
```

## ipv6 ospf3 dead-interval

### Syntax

```
ipv6 ospf3 dead-interval [1 - 65535]
```

### Description

Used in the **VLAN** context to indicate the number of seconds that a neighbor router waits for a hello packet from the specified interface before declaring the interface "down".

This command assigns the specified dead interval to all networks configured on the VLAN.

Default: 40 seconds

### Example output

```
Switch(vlan-100)#ipv6 ospf3 dead-interval 200
```

CLI to verify the set value

```
Switch# show run vlan 100

Running configuration:

vlan 100
  name “VLAN 100”
  tagged 1/1
  ipv6 enable
  ipv6 address 2002::3/64
  ipv6 ospf3 area backbone
  ipv6 ospf3 dead-interval 50
  exit

Switch#
```

## ipv6 ospf3 hello-interval

### Syntax

```
ipv6 ospf3 hello-interval [1 - 65535]
```

### Description

Used in the **VLAN** context to indicate the length of time between the transmission of hello packets from the routing switch to adjacent neighbors on that VLAN.

This command assigns the specified Hello interval to all networks configured on the VLAN.

Default: 10 seconds

## ipv6 ospf3 priority

**Syntax**

`ipv6 ospf3 priority [1 - 255]`

**Description**

Used in the **VLAN** context to enable changing the priority of an OSPFv3 router. The priority is used when selecting the designated router (DR) and backup designated routers (BDRs).

The value can be from 0 to 255 (with 255 as the highest priority). If you set the priority to 0, the routing switch does not participate in DR and BDR election.

This command assigns the specified priority to all networks configured on the VLAN.

Default: 1

## ipv6 ospf3 retransmit-interval

**Syntax**

`ipv6 ospf3 retransmit-interval [1 - 1800]`

**Description**

Used in the **VLAN** context to enable changing the retransmission interval for link-state advertisements (LSAs) on an interface.

Default: 5 seconds

## ipv6 ospf3 transit-delay

**Syntax**

`ipv6 ospf3 transit-delay [1 - 1800]`

**Description**

Used in the **VLAN** context to enable changing the time it takes to transmit link state update packets on this interface.

This command reconfigures the estimated number of seconds it waits to transmit a link state update packet to all networks configured on the VLAN.

Default: 1 second

# Viewing a summary of OSPFv3 configuration information

**Syntax**

`show ipv6 ospf3 [ general ]`

**Description**

Displays the summary of OSPFv3 information, such as the areas configured, address ranges defined, interface information, timers, and virtual links.

**Options**

**`general`**

Displays the OSPFv3 general status information and other generic information.

**Example output**

This example shows the output for `show ipv6 ospf3` command.

```
Switch# show ipv6 ospf3

 OSPFv3 Configuration Information

  OSPFv3 Protocol : Enabled
  Router ID : 10.0.8.35

 Currently defined areas:

                 Stub         Stub        Stub
Area ID   Type   Default Cost Summary LSA Metric Type   SPF Runs
--------- ------ ------------ ----------- ------------- --------
backbone  Normal 1            don't send  Ospfv3 Metric
10.3.16.0 Normal 1            don't send  Ospfv3 Metric
10.3.32.0 Normal 1            don't send  Ospfv3 Metric

 Currently defined address ranges:

                             Prefix
  AreaID          LSAType    Advt   IPv6Addr
  --------------- ---------- ------ --------------
 OSPFv3 interface configuration:

                             Admin
  Interface   Area ID        Status   Type  Cost  Pri
  ---------- --------------- -------- ----- ----- ---
  vlan-55    0.0.0.1         Enabled  BCAST 1     1
  vlan-75    0.0.0.1         Enabled  BCAST 1     1
  tunnel-3   0.0.0.0         Enabled  P2P   1     1

 OSPFv3 configured interface timers:

             Transit Retransmit Hello    Dead
  Interface  Delay   Interval   Interval Interval
  ---------- ------- ---------- -------- ----------
  vlan-55    1       5          10       40
  VLAN-75    1       5          10       40
  tunnel-3   1       5          10       40

 OSPFv3 configured virtual interfaces:

                                  Xmit  Rxmt  Hello Dead
  Transit AreaID  Neighbor Router Delay Intvl Intvl Interval
  --------------- --------------- ----- ----- ----- ----------
  0.0.0.1         15.255.155.1    1     5     10    40
```

## show ipv6 route ospf3

### Syntax

`show ipv6 route ospf3 [` *`dest-ipv6-addr`* `]`

### Description

Displays OSPFv3 route entries in the routing table.

### Options

**dest-ipv6-addr**

Displays the OSPFv3 routing table entry for a specific destination.

**Example output**

This example shows output for all OSPFv3 routes in the routing table:

```
Switch(config)# show ipv6 route ospf3

                              IPv6 Route Entries

 T(Type):
  S: Static  C: Connected  O: OSPFv3  R: RIPng

 ST(Sub-type):
  O : OSPF Intra  E1:External1  N1:NSSA Ext1
  OI: OSPF Inter  E2:External2  N2:NSSA Ext2

 Destination/
  Gateway                                         T     ST     Distance
Metric
-------------------------------------------------------------------------------------
--
 121::2/128
 fe80::3ea8:2aff:fe3f:8200%vlan1001               O     O      110          1
 2000::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     O      110          2
 2000::/64
 fe80::328d:99ff:fe56:8300%vlan202                O     O      110          2
 2001::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     E2     110          10
 2003::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     E2     110          10
 2004::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     E2     110          10
 2005::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     E2     110          10
 2540::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     O      110          2
 2541::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     E2     110          10
 2542::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     O      110          2
 2544::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     E2     110          10
 3001::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     E2     110          10
 5001::/64
 fe80::328d:99ff:fe4c:543f%vlan202                O     E2     110          10
 5101::/64
```

## show ipv6 ospf3 area

**Syntax**

show ipv6 ospf3 area [*area-id*|backbone][ detail ]

**Description**

Displays summary information on all configured areas.

**Options**

Displays summary information on all configured areas.

**[*area-id*|backbone]**

Displays summary information for the specified area.

**detail**

Displays area summary information in a modified format.

**ospf-area-id**

Shows information for the specified area. If no area is specified, information for all the OSPFv3 areas configured is displayed.

### Example output

This example shows the `show ipv6 ospf3 area` command output:

```
Switch# show ipv6 ospf3 area

 OSPFv3 Area Information

  Area ID   Type     Cost  SPF Runs ABR  ASBR  LSA  Checksum
  --------- -------- ----- --------- ---- ----- ---- ----------
  0.0.0.0   Normal   10    780       1    0     6    0x0005050f
  0.0.0.1   Normal   10    780       2    1     9    0x0004d3f1
  0.0.0.3   Normal   10    780       1    0     11   0x0007f854
```

## show ipv6 ospf3 interface

### Syntax

show ipv6 ospf3 interface [vlan *vid*|tunnel *tunnel-id*|loopback *lo-id*][ detail ]

### Description

Displays basic OSPFv3 information related to the VLANs configured on the routing switch.

### Options

***vlan-id***

Displays information for a specific VLAN.

**tunnel *tunnel-id***

Displays information for a specific tunnel.

**loopback *lo-id***

Displays information for a loopback interface.

**detail**

Displays additional, VLAN-specific OSPFv3 information.

### Example output

This example shows the `show ipv6 ospf3 interface` command output:

```
Switch# show ipv6 ospf3 interface

 OSPFv3 configuration and statistics for interfaces

 Interface     Status   Area ID      State   Cost   Pri   Passive
 ------------ -------- ------------ ------- ------ ----- -------
 vlan-55      Enabled  0.0.0.1      BDR     1      1     No
```

```
 vlan-75      Enabled  0.0.0.3      BDR     1       1      No
 tunnel-3     Enabled  0.0.0.0      Down    1       1      No
```

This example shows the `show ipv6 ospf3 interface tunnel` command output:

```
Switch(config)# show ipv6 ospf3 interface tunnel 3

 OSPFv3 configuration and statistics for Tunnel 3

 Interface     Status    Area ID      State  Cost   Pri   Passive
 ------------- --------- ------------ ------ ------ ----- -------
 tunnel-3      Enabled   1.2.3.4      DOWN   1      1     No
```

This example shows the `show ipv6 ospf3 interface detail` command output:

```
Switch(config)# show ipv6 ospf3 interface detail

 OSPFv3 configuration and statistics for VLAN 22

  Interface        : vlan-22   Status                     : Enabled
  Area ID          : 1.2.3.4   State                      : DOWN
  Priority         : 1         Cost                       : 1
  Type             : BCAST     Passive                    : No
  Hello Interval   : 10        Dead Interval              : 890
  Transit Delay    : 1         Retransmit Interval        : 5
  Events           : 0         Designated Router          : 0.0.0.0
  Neighbors        : 0         Backup Designated Router   : 0.0.0.0

 OSPFv3 configuration and statistics for Tunnel 3

  Interface        : tunnel-3  Status                     : Enabled
  Area ID          : 0.0.0.0   State                      : DOWN
  Priority         : 1         Cost                       : 1
  Type             : P2P       Passive                    : No
  Hello Interval   : 10        Dead Interval              : 50
  Transit Delay    : 1         Retransmit Interval        : 5
  Events           : 0         Designated Router          : 0.0.0.0
  Neighbors        : 0         Backup Designated Router   : 0.0.0.0

 OSPFv3 configuration and statistics for Loopback 1

  Interface        : lo-1      Status                     : Enabled
  Area ID          : 1.2.3.4   State                      : DOWN
  Priority         : n/a       Cost                       : 1
  Type             : BCAST     Passive                    : n/a
  Hello Interval   : 10        Dead Interval              : 50
  Transit Delay    : n/a       Retransmit Interval        : n/a
  Events           : 0         Designated Router          : n/a
  Neighbors        : n/a       Backup Designated Router   : n/a
```

**Detail Option for a VLAN**

```
Switch# show ipv6 ospf3 interface vlan 55 detail

 OSPFv3 configuration and statistics for VLAN 55

  Interface        : vlan-55    Status                     : Enabled
  Area ID          : 0.0.0.1    State                      : BDR
```

```
  Priority        : 1         Cost                    : 1
  Type            : BCAST     Passive                 : No
  Hello Interval  : 10        Dead Interval           : 40
  Transit Delay   : 1         Retransmit Interval     : 5
  Events          : 0         Designated Router       : 10.0.0.1
  Neighbors       : 1         Backup Designated Router : 10.0.1.4
```

## show ipv6 ospf3 neighbor

**Syntax**

`show ipv6 ospf3 neighbor [ `*`router-id`*` ][ detail ]`

**Description**

Displays OSPFv3 information learned for neighbor routers.

**Options**

***`router-id`***

Displays information for a specific neighbor router.

**`detail`**

Displays additional, neighbor-specific OSPFv3 information.

**Example output**

This example shows the `show ipv6 ospf3 neighbor` command output:

```
Switch(ospf3)# show ipv6 ospf3 neighbor

 OSPFv3 Neighbor Information

  Interface Router ID       Pri   State      Rxmt QLen  Events
  --------- --------------- ----- ---------- ---------- ------
  Vlan-55   15.1.0.1        1     FULL       0          0
  Vlan-75   15.1.1.2        1     FULL       0          0
  tunnel-3   4.3.2.1        1     Full       0          0
```

**`show ipv6 ospf3 neighbor` detail output**

```
Switch(ospf3)# show ipv6 ospf3 neighbor 15.1.0.1 detail

 OSPFv3 Neighbor Information for neighbor 15.1.0.1

  IPv6 Address   : fe80::55:1
  Router ID   : 15.1.0.1     State                    : FULL
  Interface   : vlan-55      Designated Router        : 15.1.0.1
  Area        : 0.0.0.1      Backup Designated Router : 15.1.0.4
  Priority    : 1            Retransmit Queue Length  : 0
  Options     : 19           Neighbor Uptime          : 1days
  Events      : 0            Dead Timer Expires       : 33 sec

 OSPFv3 Neighbor Information for neighbor 15.1.0.1

  IPv6 Address   : fe80::55:1
  Router ID   : 15.1.0.1     State                    : FULL
  Interface   : tunnel-3     Designated Router        : 0.0.0.0
  Area        : 0.0.0.1      Backup Designated Router : 0.0.0.0
```

```
  Priority    : 1            Retransmit Queue Length  : 0
  Options     : 19           Neighbor Uptime          : 1days
  Events      : 0            Dead Timer Expires       : 33 sec
```

## show/clear ipv6 ospf3 statistics

### Syntax

`show ipv6 ospf3 statistics [vlan` *`vid`*`|tunnel` *`tunnel-id`*

`clear ipv6 ospf3 statistics [vlan` *`vid`*`|tunnel` *`tunnel-id`*

### Description

Displays the statistics on OSPFv3 packets sent and received on the VLAN interfaces on an OSPFv3-enabled routing switch, including the number of errors that occurred during packet transmission.

### Options

**`vlan`** ***`vid`***

Displays the statistics for the specified VLAN.

**`clear`**

Resets the OSPFv3 traffic counters to zero.

**`vlan`** ***`vid`***

Resets only those counters in the specific VLAN.

**tunnel** ***tunnel-id***

Using the `tunnel` option resets only those counters in the specific tunnel.

**Displaying OSPFv3 traffic statistics for all VLANs configured for OSPFv3 operation**

```
Switch# show ipv6 ospf3 statistics

 OSPFv3 statistics for Interfaces

  Interface    Total Tx        Total Rx        Total Errors
  ------------ --------------- --------------- ------------
  Vlan-22      9022            9018            3
  Vlan-55      28041           28480           15
  tunnel-3     5432            5430            2
```

**Displaying OSPFv3 statistics for a single VLAN**

```
Switch# show ipv6 ospf3 statistics vlan 55

 OSPFv3 statistics for VLAN 55

  Tx Hello Packets   : 26005     Rx Hello Packets   : 26005
  Tx DD Packets      : 3         Rx DD Packets      : 3
  Tx LSR Packets     : 1         Rx LSR Packets     : 1
  Tx LSU Packets     : 1436      Rx LSU Packets     : 1046
  Tx LSA Packets     : 615       Rx LSA Packets     : 1444

 OSPFv3 Errors       : 0
```

#### Displaying OSPFv3 Statistics for a Tunnel

```
Switch# show ipv6 ospf3 statistics tunnel 3

 OSPFv3 statistics for Tunnel 3

  Tx Hello Packets    : 26000     Rx Hello Packets    : 26000
  Tx DD Packets       : 3         Rx DD Packets       : 3
  Tx LSR Packets      : 1         Rx LSR Packets      : 1
  Tx LSU Packets      : 1436      Rx LSU Packets      : 1046
  Tx LSA Packets      : 615       Rx LSA Packets      : 1444

  OSPFv3 Errors : 0
```

## show ipv6 ospf3 link-state as-scope

The commands in this section enable display of the routing-switch's OSPFv3 link-state AS-scope database for the entire AS, with options for narrowing the scope of the output and increasing the range of settings included in the output.

#### Syntax

`show ipv6 ospf3 link-state as-scope [ lsid` *`lsid-#`* `][ router-id` *`rtr-id-#`* `][detail| advertise]`

#### Description

Displays link-state as-scope database for all links configured on the routing switch. The range of displayed data can be reduced by using one or more subset options:

#### Options

**`lsid`** ***`lsid-#`***

Subset option to filter displayed LSA database or advertisements to show only the AS-scope data having the specified (32-bit) IP address as a link-state ID.

Can also be filtered with the `router-id` option to further define the source of displayed information.

**`router-id`** ***`rtr-id-#`***

Subset option to filter displayed LSA database or advertisements to show only the AS-scope data having the specified router-ID.

Can also be filtered with the `lsid` option to further define the source of displayed information.

**`detail`**

Displays additional details for each LSA included in the range of displayed LSAs for any of the above options.

**`advertise`**

Displays the hexadecimal data in LSA packets (advertisements) within the OSPFv3 AS- scope on the routing switch.

The output can also be filtered by `lsid` and `router-id`.

#### Example output

To display link-state AS-scope link-state information, enter `show ipv6 ospf3 link-state as-scope` at any CLI level. When you enter this command, an output similar to the following is displayed:

```
Switch# show ipv6 ospf3 link-state as-scope
```

```
OSPFv3 AS Scope Link State Database

             Advertising
LSA Type     Router ID     Link State ID Age Sequence # Checksum
----------- ------------ ------------- --- ---------- ----------
As-External 15.1.1.2     0             57  0x80000037 0x0000b76b
As-External 15.1.1.2     2             945 0x80000062 0x0000b3d1
As-External 15.1.1.2     3             987 0x80000062 0x0000dfa0
As-External 15.255.155.1 1             250 0x8000006f 0x0000fae0
```

To display link-state AS-Scope LSA advertisements in hexadecimal format, use the `advertise` keyword. The following example displays output for router ID 15.1.1.2 in an AS.

**`show ipv6 ospf3 link-state as-scope` advertise output**

```
Switch# show ipv6 ospf3 link-state as-scope router-id 15.1.1.2 advertise

OSPFv3 AS Scope Link State Database

Raw Advertisements
------------------------------------------------------------------------
1d6e4005000000000f01010280000037b76b00280500000a400000002620000f00000000
00000000

19f64005000000020f01010280000062b3d100380700000a400000002620000b00000000
2620000a0000000000000000011000100000000

19cc4005000000030f01010280000062dfa000380700000a400000002621000e00000000
2620000a0000000000000000011000100000000
```

## ospf3 link-state area-scope

The commands in this section enable display of the routing-switch's OSPFv3 link-state area-scope database, with options for narrowing the scope of the output and increasing the range of settings included in the output.

### Syntax

`show ipv6 ospf3 link-state area-scope`

`area` *`area-id`* `lsid` *`lsid-#`* `router-id` *`rtr-#`* `type` *`lsa-type`*`[detail|advertise]`

### Description

Displays link-state database for all areas configured on the routing switch. The range of displayed data can be reduced by using one or more subset options:

### Options

**`area`** ***`area-id`***

Subset option to filter displayed LSA database or advertisements to show only the data from a specific OSPFv3 area. Can also be filtered with other subset options (`lsid`, `router-id` , and `type`) to further define the source of displayed information.

**`lsid`** ***`lsid-#`***

Subset option to filter displayed LSA database or advertisements to show only the data from sources having the specified IP address as a link-state ID.

Can also be filtered with other subset options (`area`, `lsid`, `router-id`, and `type`) to further define the source of displayed information.

**router-id *rtr-#***

Subset option to filter displayed LSA database or advertisements to show only the data from sources having the specified router ID.

Can also be filtered with other subset options (`area, lsid`, and `type`) to further define the source of displayed information.

**type *lsa-type***

Subset option to filter displayed LSA database or advertisements to show only the data from sources having the specified type. Can also be filtered with other subset options (`area, lsid`, and `router-id`) to further define the source of displayed information.

LSA type options include: `router|network|inter-area-prefix|inter-area-router|nssa|intra-area-prefix`

**detail**

Displays additional details for each LSA included in the range of displayed LSAs for any of the above options.

**advertise**

Displays the hexadecimal data in LSA packets (advertisements) for the OSPFv3 areas configured on the routing switch.

The output can also be filtered by area (`area-id`), `lsid`, `router-id`, and/or `type`.

Default: All OSPFv3 areas on the routing switch.

**Example output**

To display OSPFv3 link-state information, enter `show ipv6 ospf3 link-state area-scope` at any CLI level. When you enter this command, the switch displays an output similar to the following for all configured areas:

```
Switch# show ipv ospf3 link-state area-scope

OSPFv3 Area Scope Link State Database for area 0.0.0.0

                  Advertising  Link
LSA Type          Router ID    State ID Age Sequence # Checksum
----------------- ------------ -------- --- ---------- ----------
Router            1.0.0.4      0        2   0x80000037 0x00000a25
Router            1.1.1.1      0        20  0x8000038a 0x00004be4
Router            15.255.155.1 0        1   0x80000373 0x00006c10
Network           15.255.155.1 599      21  0x80000069 0x0000cd3e
Inter-Area-Prefix 1.0.0.4      1        22  0x80000002 0x00003cbf
Inter-Area-Prefix 1.0.0.4      3        22  0x80000002 0x00002870
Inter-Area-Prefix 1.0.0.4      5        22  0x80000002 0x00002273
Inter-Area-Prefix 1.0.0.4      7        22  0x80000002 0x00003c54
Inter-Area-Prefix 15.255.155.1 2        61  0x80000002 0x0000ba35
Inter-Area-Prefix 15.255.155.1 3        22  0x80000002 0x000080d1
Inter-Area-Prefix 15.255.155.1 5        62  0x80000002 0x00006689
Inter-Area-Router 1.0.0.4      10       23  0x80000002 0x00003139
Inter-Area-Router 1.0.0.4      12       23  0x80000002 0x00005da3
Intra-Area-Prefix 15.255.155.1 599      22  0x80000068 0x00002bd7

OSPFv3 Area Scope Link State Database for area 0.0.0.1

                  Advertising  Link
LSA Type          Router ID    State ID Age Sequence # Checksum
----------------- ------------ -------- --- ---------- ----------
Router            1.0.0.4      0        14  0x80000004 0x000085ef
```

```
Router             15.255.155.1 0        13  0x80000004 0x0000ddee
Network            15.255.155.1 632      23  0x80000002 0x00005ef2
Inter-Area-Prefix 1.0.0.4      6        25  0x80000002 0x0000187c
Inter-Area-Prefix 1.0.0.4      14       25  0x80000002 0x000095f9
Inter-Area-Prefix 15.255.155.1 4        64  0x80000002 0x000046ad
Inter-Area-Prefix 15.255.155.1 6        19  0x80000002 0x0000707c
Inter-Area-Router 1.0.0.4      13       25  0x80000002 0x000053ac
Inter-Area-Router 15.255.155.1 7        19  0x80000002 0x0000eb72
Intra-Area-Prefix 1.0.0.4      0        15  0x80000004 0x00006d56
Intra-Area-Prefix 15.255.155.1 0        14  0x80000004 0x000070a1
Intra-Area-Prefix 15.255.155.1 632      24  0x80000002 0x00003ae9
```

To display area-scope LSA advertisements in hexadecimal format, use the `advertise` keyword.

**Output for `show ipv6 ospf3 link-state area-scope advertise`**

```
Switch# show ipv ospf3 link-state area-scope advertise area 1 router-id 1.0.0.4

OSPFv3 Area Scope Link State Database for area 0.0.0.1

Raw Advertisements
------------------------------------------------------------------------
0e6020010000000001000004800000583f0002801000013020000010000027800000278
0fff9b01

0e3820030000000601000004800000003167d002400000001400000002620000f00000000
0e38200300000000e010000048000000393fa002400000002400000002620000b00000000

0e38200400000000d0100000480000000351ad00200000000130000000010f010102
0e6a200900000000010000048000000056b57003400012001000000000100000480020000
```

## show ospf3 link-state link-scope information

The commands in this section enable display of the routing-switch's OSPFv3 link-state link-scope database, with options for narrowing the scope of the output and increasing the range of settings included in the output.

### Syntax

`show ipv6 ospf3 link-state link-scope interface` *`vlan-id`* `lsid` *`lsid-#`* `router-id` *`rtr-id-#`*`[detail|advertise]`

### Description

Displays link-state link-scope database for all links configured on the routing switch. The range of displayed data can be reduced by using one or more subset options:

### Options

**`interface`** ***`vlan-id`***

Subset option to filter displayed LSA database or advertisements to show only the link-scope data from a specific VLAN.

Can also be filtered with other subset options (`lsid` and `router-id`) to further define the source of displayed information.

**`lsid`** ***`lsid-#`***

Subset option to filter displayed LSA database or advertisements to show only the link-scope data having the specified (32-bit) IP address as a link-state ID.

Can also be filtered with other subset options (`router-id` and `interface`) to further define the source of displayed information.

**router-id *rtr-id-#***

Subset option to filter displayed LSA database or advertisements to show only the link-scope data having the specified router-ID.

Can also be filtered with other subset options (`lsid` and `interface`) to further define the source of displayed information.

**detail**

Displays additional details for each LSA included in the range of displayed LSAs for any of the above options.

**advertise**

Displays the hexadecimal data in LSA packets (advertisements) within the OSPFv3 link scope on the routing switch.

The output can also be filtered by `lsid` and `router-id`.

**Example output**

To display link-scope link-state information, enter `show ipv6 ospf3 link-state link-scope` at any CLI level. When you enter this command, an output similar to the following is displayed:

```
Switch# show ipv6 ospf3 link-state link-scope

 OSPFv3 Link Scope Link State Database for LS index 599

           Advertising
  LSA Type Router ID    Link State ID Age Sequence # Checksum
  -------- ------------ ------------- --- ---------- ----------
  Link     1.1.1.1      599           842 0x80000009 0x0000fb93
  Link     15.255.155.1 599           903 0x80000009 0x00006185
  Link     1.0.0.4      632           845 0x80000009 0x00002a0c
  Link     15.255.155.1 632           874 0x80000009 0x0000662a
```

**Link-State Link-Scope Output for a Specific Tunnel**

```
Switch(config)# show ipv6 ospf3 link-state link-scope interface tunnel 3

 OSPFv3 Link Scope Link State Database for LS index for Tunnel 3

           Advertising
  LSA Type Router ID   Link State ID Age Sequence # Checksum
  -------- ----------- ------------- --- ---------- ----------
  Link     1.1.1.1     578           106 0x80000004 0x0000583b
  Link     2.2.2.2     578           114 0x80000004 0x000065d5
```

To display link-scope LSA advertisements in hexadecimal format, use the `advertise` keyword:

**`show ipv6 ospf3 link-state link-scope advertise` output**

```
Switch# show ipv6 ospf3 link-state link-scope advertise

OSPFv3 Link Scope Link State Database for LS index 599

Raw Advertisements
-----------------------------------------------------------------------
11a0000800000257010101018000000 9fb93003801000013fe8000000000000000000
0022000100000001400000002620000b00000000
```

```
11630008000002570fff9b0180000009618500380100013fe800000000000000000000
0022000200000001400000002620000b00000000

119e000800000278010000048000000092a0c003801000013fe80000000000000000000
0055000200000001400000002620000e00000000

11810008000002780fff9b018000000009662a003801000013fe8000000000000000000
0055000100000001400000002620000e00000000
```

## show ipv6 ospf3 redistribute

As described in **Configuring for external route redistribution in an OSPFv3 domain** on page 227, you can configure the routing switch to redistribute connected and static routes into OSPFv3. When you redistribute a route, the routing switch can use OSPFv3 to advertise the route to its OSPFv3 neighbors.

### Syntax

```
show ipv6 ospf3 redistribute
```

### Description

Displays the route types currently enabled for route redistribution on the routing switch:

### Example output

This example displays the route types currently enabled for route redistribution.

```
Switch# show ipv6 ospf3 redistribute

 OSPFv3 redistributing

 Route Type RouteMap
 ---------- ---------------------
 Connected  enabled
 Static     enabled
 ripng      enabled
```

## show ipv6 ospf3 virtual-link

### Syntax

`show ipv6 ospf3 virtual-link[ `*`rtr-id`*` ][ area `*`area-id`*` ]`

### Description

Displays OSPFv3 information learned about all virtual links detected by the routing switch.

### Options

***rtr-id***

Displays virtual link information for a specific virtual-neighbor router detected by the routing switch.

**area *area-id***

Displays information learned from a virtual neighbor detected in a specific area.

### Example output

This example displays output for all virtual links detected on the routing switch:

```
Switch# show ipv6 ospf3 virtual-link

 OSPFv3 Virtual Interface Status
```

```
 Transit AreaID  Neighbor Router Interface State
 --------------- --------------- ---------------
 0.0.0.1         1.0.0.4         P2P
```

**Display output for a specific virtual link**

```
Switch# show ipv6 ospf3 virtual-link 1.0.0.4
 Transit AreaID  : 0.0.0.1
 Neighbor Router : 1.0.0.4

 Interface State : P2P                     Transit Delay   : 1
 Events          : 1                       Rtr Interval    : 5
                                           Hello Interval  : 10
                                           Dead Interval   : 40
```

## show ipv6 ospf3 virtual-neighbor

**Syntax**

`show ipv6 ospf3 virtual-neighbor[` *`rtr-id`* `][ area` *`area-id`*`]`

**Description**

Displays OSPFv3 information learned about all virtual neighbor routers detected by the routing switch.

**Options**

**`rtr-id`**

Displays information for a specific virtual-neighbor router detected by the routing switch.

**`area`** ***`area-id`***

Displays information learned from a virtual neighbor detected in a specific area.

**Example output**

This example displays output for all virtual neighbors detected on the routing switch:

```
Switch# show ipv6 ospf3 virtual-neighbor

 OSPFv3 Virtual Interface Neighbor Information

 Router ID     State    IPv6 Addr                 Events
 ------------- -------- ------------------------- ---------
 1.0.0.4       FULL     2620:e::55:2              7
```

**Display output for a specific virtual neighbor**

```
Switch# show ipv6 ospf3 virtual-neighbor 1.0.0.4
 Router ID : 1.0.0.4
 State     : FULL

 IPv6 Addr : 2620:e::55:2
 RtQLen    : 5
 Events    : 7
```

## show ipv6 ospf3 spf-log

Enter this command to display the log used to record SPF calculations on an OSPFv3-enabled routing switch. The SPF algorithm recalculates the routes in an OSPFv3 domain when a change in the area topology is received.

#### Syntax

```
show ipv6 ospf3 spf-log
```

#### Description

Displays the event that resulted in the last 100 executions of the SPF algorithm on the routing switch. Possible events (reasons) are:

#### Options

**`Re-Init`**

OSPFv3 was enabled or disabled on the routing switch.

**`Router LS Update`**

A Router-LSA was received.

**`Network LS Update`**

A Network-LSA was received.

**`Generated RTR LSA`**

A Router-LSA was generated on the routing switch.

**`Generated NTW LSA`**

A Network-LSA was generated on the routing switch.

#### Example

This example displays the OSPFv3 SPF log:

```
Switch(ospf3)# show ipv6 ospf3 spf-log

 OSPFv3 SPF (SHORTEST PATH FIRST) LOG

 spf instance    Reason
 --------------- ---------------------------
 1               Router LS Update
 2               Router LS Update
 3               Generated RTR LSA
 4               Generated NTW LSA
 5               Network LS Update
 6               Network LS Update
 7               Generated RTR LSA
 8               Router LS Update
 9               Generated RTR LSA
 10              Re-Init
 11              Incremental LS Update
 ...              ...
```

## show running

#### Syntax

```
show running
```

#### Description

Use `show running` to view the currently active load-sharing configuration and `show config` to view the load-sharing configuration in the startup-config file. (While in its default configuration [`ip load-sharing 4`] load-sharing does not appear in the command output.) If load sharing is configured with non-default settings (disabled or configured for either two or three equal-cost next-hop paths), the current settings are displayed in the command output.

**Example output**

This example displays a non-default IP load-sharing configuration:

```
Switch(config)# show running
 Running configuration:
 ; J8697A Configuration Editor; Created on release #K.15.xx

 hostname "Switch"
 module 1 type J8702A
 snmp-server community "public" Unrestricted
 vlan 1
    name "DEFAULT_VLAN"
    untagged 1-24
    ip address dhcp-bootp
    exit
 ip load-sharing 3
```

**NOTE**: `ip load-sharing 3` indicates a non-default load-sharing configuration allowing three equal-cost next-hop paths for routed traffic with different subnet destinations. If the routing switch is configured with the default load-sharing configuration, load-sharing does not appear in the `show config` or `show running` command output.

## router ospf3 trap

**Syntax**

```
router ospf3 trap <tab>
```

**Description**

This command displays the router OSPF3 traps.

**Options**

**all-traps**

Enable all the OSPFv3 traps.

**`disable`**

Disable OSPFv3 traps.

**`enable`**

Enable OSPFv3 traps.

**`interface-state-change`**

Send a trap when the state of a non-virtual interface changes.

**`virtual-interface-state-change`**

Send a trap when the state of a virtual interface changes.

**`neighbor-state-change`**

Send a trap when the state of a non-virtual neighbor changes.

**virtual-neighbor-state-change**

Send a trap when the state of a virtual neighbor changes.

**interface-config-error**

Send a trap when a configuration conflict occurs for a non-virtual interface.

**virtual-interface-config-error**

Send a trap when a configuration conflict occurs for a virtual interface.

**interface-receive-bad-packet**

Send a trap when an invalid packet is received on a non-virtual interface.

**virtual-interface-receive-bad-packet**

Send a trap when an invalid packet is received on a virtual interface.

**restart-status-change**

Send a trap when the graceful restart state changes for the device.

**nbrrestart-helper-status-change**

Send a trap when the graceful restart helper state changes for the neighbor.

**virtual-nbrrestart-helper-status-change**

Send a trap when the graceful restart helper state changes for the virtual neighbor.

**interface-retransmit-packet**

[Deprecated] A packet has been retransmitted on a non-virtual interface.

**virtual-interface-retransmit-packet**

[Deprecated] A packet has been retransmitted on a virtual interface.

**originate-lsa**

[Deprecated] A new LSA has been sent.

**max-age-lsa**

[Deprecated] An LSA expired as its maximum age has been reached.

**unknown-lsa**

[Deprecated] An LSA with unknown code has been received.

**lsdb-overflow**

[Deprecated] Send a trap when number of LSAs in router's lsdb has exceeded the limit.

**lsdb-approach-overflow**

[Deprecated] Send a trap when number of LSAs in router's lsdb has exceeded ninety percent of the limit.

**asbr-status-change**

[Deprecated] Send a trap when ASBR status changes for the device.

**abr-status-change**

[Deprecated] Send a trap when ABR status changes for the device.

# debug ipv6 ospf3

**Syntax**

```
debug ipv6 ospf3 [adj|event|flood|lsa-generation|packet|retransmission|spf]
```

**Description**

Turns on the tracing of OSPFv3 packets. For more information, see the "Debug Command" section in the "Troubleshooting" appendix of the *Aruba-OS Switch Management and Configuration Guide* for your routing switch.

# Graceful shutdown of OSPFv3 Routing

It is now possible to gracefully shut down OSPF routing on HPE switches without losing packets that are in transit. OSPF neighbors are informed that the router should not be used for forwarding traffic, which allows for maintenance on the switch without interrupting traffic in the network. There is no effect on the saved switch configuration.

Prior to a switch shutdown, the CLI/SNMP `reload` command or the CLI `boot` command is executed to initiate the sending of OSPF “empty Hello list” messages on the interfaces that are part of the OSPF routing configuration. After a small delay (approximately 2 seconds) that allows the messages to be transmitted on all applicable interfaces, the `boot` or `reload` command continues.

## Modules operating in non-stop mode

When a switch is in standalone mode and OSPF routing is enabled, the “empty Hello list” is transmitted whenever the `boot` or `reload` commands are executed.

When the switch is operating in nonstop switching mode (redundant) and a single module is being reloaded or booted, the standby module will notify neighboring switches of the management module failover. If the failover fails, the “empty Hello list” is transmitted before the switch is rebooted.

When a switch is operating with multiple management modules in warm standby mode, the “empty Hello list” is sent when a `reload` or `boot` command is executed. The standby management module sends out OSPF Hello packets after becoming the active management module.

# ip load-sharing

**Syntax**

`ip load-sharing` *`2 - 4`*

`no ip load-sharing` *`2 - 4`*

**Description**

When OSPF is enabled and multiple, equal-cost, next-hop routes are available for traffic destinations on different subnets, this feature, by default, enables load-sharing among up to four next-hop routes.

**Specifiers**

***2 - 4***

Specifies the maximum number of equal-cost next-hop paths the router allows.

Default: Enabled with four equal-cost, next-hop routes allowed

The `no` form of the command disables this load-sharing so that only one route in a group of multiple, equal-cost, next-hop routes are used for traffic that could otherwise be load-shared across multiple routes.

NOTE

Disabling load-sharing means that router "1" selects only one next-hop router for traffic that is actually eligible for load-sharing through different next-hop routers.

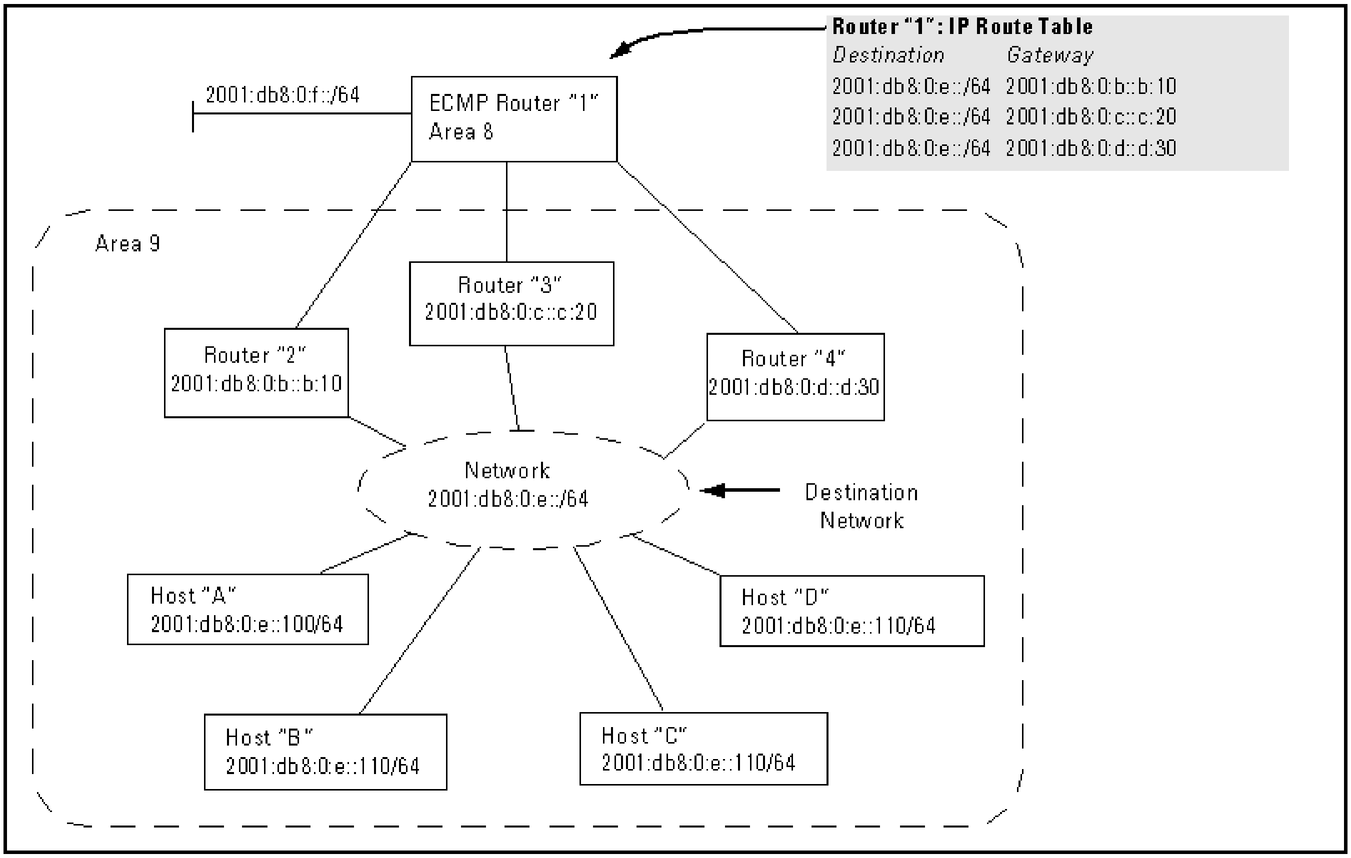
In the default configuration, load-sharing is enabled by default for both IPv4 and IPv6. However, it has no effect unless routing and OSPF are enabled.

**Example**

In the figure, the next-hop routers "B," "C," and "D" are available for equal-cost load-sharing of eligible traffic. Disabling IP load-sharing means that router "A" selects only one next-hop router for traffic that is actually eligible for load-sharing through different next-hop routers. The ECMP inter-area routes to destination network 2001:db8:0:e::/64 consist of the following next-hop gateway addresses:

- 2001:db8:0:b::b:101
- 2001:db8:0:c::c:101
- 2001:db8:0:d::d:101

**Figure 25:** *OSPFv3 ECMP multiple next-hop routing (inter-area)*




However, the forwarding software distributes traffic across the three possible next-hop routes in such a way that all traffic for a specific host is sent to the same next-hop router.

As shown in the figure, one possible distribution of traffic to host devices is:

- Traffic to host "A" passes through next-hop router "3"
- Traffic to host "B" passes through next-hop router "2"
- Traffic to host "C" passes through next-hop router "3"
- Traffic to host "D" passes through next-hop router "4"

| IP packet destination | Next hop used |
|---|---|
| 2001:db8:0:e::100 | 2001:db8:0:b::b:10 |
| 2001:db8:0:e::110 | 2001:db8:0:c::c:20 |
| 2001:db8:0:e::120 | 2001:db8:0:b::b:10 |
| 2001:db8:0:e::130 | 2001:db8:0:d::d:30 |

# Equal-cost multi-path routing

The ECMP feature allows OSPFv3 to add routes with multiple next-hop addresses and with equal costs to a given destination in the forwarding information base (FIB) on the routing switch. For example, if multiple, equal-cost, next-hop routes exist on a routing switch for a destination in a network with the prefix 2620:e::/64, these routes would appear similar to the following in the IPv6 Route Entries Table:

**`show ipv6 route` command output with multiple next-hop routes**

```
Switch(config)# show ipv6 route

IPv6 Route Entries

Destination : ::1/128
Gateway : lo0
Type: connected  Sub-Type: NA         Distance: 0    Metric: 1

Destination : 2620:c::/64
Gateway : 2620:e::55:2
Type: static     Sub-Type: NA         Distance: 200  Metric: 1

Destination : 2620:a::/64
Gateway : fe80::22:3%vlan22
Type: ospf3      Sub-Type: InterArea  Distance: 110 1
Metric: 2

Destination : 2620:a::/64
Gateway : fe80::22:5%vlan22
Type: ospf3      Sub-Type: InterArea  Distance: 110  Metric: 2

Destination : 2620:a::/64
Gateway : fe80::22:11%vlan22
Type: ospf3      Sub-Type: InterArea  Distance: 110  Metric: 2

Destination : 2620:b::/64
Gateway : VLAN22
Type: connected  Sub-Type: NA         Distance: 0    Metric: 1
```

[1] Multiple next-hop gateway addresses are displayed for the destination network 2620:a::/64

For a given destination network in an OSPFv3 domain, multiple ECMP next-hop routes can be **one** of the following types.

- Intra-area (routes to the destination in the same OSPFv3 area)
- Inter-area (routes to the destination through another OSPFv3 area)
- External (routes to the destination through another autonomous system)

Multiple ECMP next-hop routes cannot be a mixture of intra-area, inter-area, and external routes. In the above example, the multiple next-hop routes to network 2620:a::/64 are all inter-area.

Also, according to the distributed algorithm used in the selection of ECMP next-hop routes:

- Intra-area routes are preferred to inter-area routes.
- Inter-area routes are preferred to external routes through a neighboring AS.

In addition, ECMP ensures that all traffic forwarded to a given host address follows the same path, which is selected from the possible next-hop routes.

ECMP load-sharing does not affect routed traffic to different hosts on the same subnet. That is, all traffic for different hosts on the same subnet will go through the same next-hop router. For example, if subnet 2001:db8:0:1f::/64 includes two servers at 2001:db8:0:1f::1ab:101 and 2001:db8:0:1f::1ab:93, all traffic from router "A" to these servers will go through the same next-hop router.

# Influencing route choices by changing the default administrative distance

The administrative distance value can be left in its default configuration setting (110) unless a change is needed to improve OSPFv3 performance for a specific network configuration.

The switch can learn about networks from various protocols. Consequently, the routes to a network may differ depending on the protocol from which the routes were learned. On the routing switches, the administrative distance for OSPFv3 routes is set at 110 for all route types (external, inter-area, and intra-area).

The routing switch selects routes on the basis of route source information. To enable this operation, the administrative distance assigned to each source is used to influence route choices. You can change the distance settings in the **OSPFv3 global** context to enable preference of one route typeover another.

# Adjusting performance by changing the VLAN interface settings

The following OSPFv3 interface parameters are automatically set to their default values. No change to the defaults is usually required unless needed for specific network configurations.

| Parameter | Default | Page |
|---|---|---|
| `cost` | 1 | **ipv6 ospf3 cost** on page 229 |
| `dead-interval` | 40 seconds | **ipv6 ospf3 dead-interval** on page 229 |
| `hello-interval` | 10 seconds | **ipv6 ospf3 hello-interval** on page 229 |
| `priority` | 1 | **ipv6 ospf3 priority** on page 230 |

*Table Continued*

| Parameter | Default | Page |
| --- | --- | --- |
| `retransmit-interval` | 5 seconds | **ipv6 ospf3 retransmit-interval** on page 230 |
| `transit-delay` | 1 second | **ipv6 ospf3 transit-delay** on page 230 |
| `passive` | disabled | **ipv6 ospf3 passive** on page 208 |

Settings are configured on a per-interface basis.

NOTE

Most of these parameters also apply to virtual link configurations. However, when used on a virtual link configuration, the **OSPFv3** context requirement is different and the parameters are applied only to the interfaces included in the virtual link. See **Adjusting virtual link performance by changing the interface settings** on page 199.

# IPv6 tunneling over IPv4 using manually configured tunnels

All commands previously in the Summary of commands table are indexed under the entry *Command syntax*.

IPv6 over IPv4 tunneling is a way to establish point-to-point tunnels by encapsulating IPv6 packets within IPv4 headers so that they can be carried over the IPv4 routing infrastructure. IPv6 over IPv4 tunneling provides a mechanism for utilizing the existing IPv4 routing infrastructure to carry IPv6 traffic between IPv6 networks.

There are a number of IPv6 tunneling mechanisms. Currently only tunneling IPv6 traffic over an IPv4 network through 6in4 manually configured tunnel endpoints is supported.

Tunnels are an additional routing interface type, similar to a VLAN interface or a loopback interface. Routing into 6in4 tunnels is supported for:

- A standard route table lookup
- Static Routes
- Policy Based Routing (PBR)
- Running OFPFv3 over the point-to-point tunnel interface

See *RFC 4213 Basic Transition Mechanisms for IPv6 Hosts and Routers* for more information about tunneling.

Tunneling can be used for:

- Router to router—IPv4 routers connected by an IPv4 infrastructure can tunnel IPv6 packets among themselves. The tunnel spans one segment of the end-to-end path.
- Host to router—IPv4 and IPv6 hosts can tunnel IPv6 packets to an intermediary IPv6 or IPv4 router that is reachable through an IPv4 infrastructure. The tunnel spans the first segment of the end-to-end path.
- Host to host—IPv6 or IPv4 hosts that are interconnected by an IPv4 infrastructure can tunnel IPv6 packets among themselves. The tunnel spans the entire end-to-end path.
- Route to host— IPv6 or IPv4 routers can tunnel IPv6 packets to their final destination IPv6 or IPv4 host. This tunnel spans only the last segment of the end-to-end path.

Configured tunnels are in the router-to-router configuration because the tunnel endpoints need to be explicitly configured.

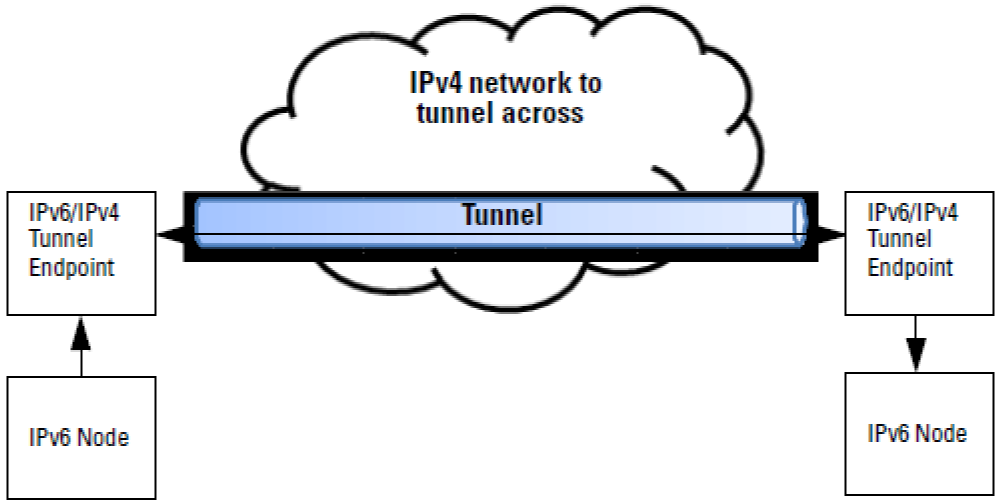
The tunnel endpoint includes:

- The entry node of the tunnel (the encapsulator), which creates an encapsulating IPv4 header and sends the encapsulated packet. Which packets to tunnel is determined by a routing table lookup based on the IPv6 address.
- The exit node of the tunnel (the decapsulator):
    - receives the encapsulated packet
    - reassembles the packet if needed
    - removes the IPv4 encapsulating header
    - processes the IPv6 packet in the usual manner

The decapsulator matches received packets to the tunnels it has configured, and only processes packets where the IPv4 source and destination addresses match the endpoint addresses of the configured tunnels. A tunnel's IPv4 address must be the same on both the encapsulator and the decapsulator. IPv4 routing switches route the packet based on the IPv4 header.

IPv6 traffic can travel the tunnel in either direction. Each end node can be either the encapsulator or the decapsulator depending on the flow of the IPv6 traffic.

**Figure 26:** *Conceptual Example of a Tunnel*




A tunnel is treated as a single point-to-point link; the encapsulator and decapsulator behave as IPv6 neighbors on that link. The encapsulator and decapsulator assign IPv6 link-local addresses to the interface and may also assign IPv6 global addresses. Neighbor discovery and duplicate address detection are implemented as they are on any other IPv6 interface.

# Commands to configure a tunnel interface

An IPv6 address is configured on the tunnel interface in the same way that it would be on other IP routing interfaces, such as VLANs. IPv4 addresses are configured as the tunnel source and tunnel destination endpoint addresses.

To create a tunnel, enter the following commands in the **global config** context.

## interface tunnel

### Syntax

`interface tunnel` *`1 - 128`*

`no interface tunnel` *`1 - 128`*

### Description

Creates a tunnel. **Tunnel interface** context is entered. The `no` form of the command removes the tunnel configuration.

### Usage

To enable or disable the tunnel, enter this command in **tunnel** context.

## tunnel [enable|disable]

### Syntax

`tunnel [enable|disable].`

### Description

Enables or disables the tunnel. The `enable` command only succeeds if all mandatory parameters such as source and destination addresses for the tunnel are configured.

### Usage

If `disable` is specified, the tunnel configuration is not removed. To optionally configure a name for the tunnel, enter this command in **tunnel** context.

## tunnel name string

### Syntax

`tunnel name` *`string`*

`no tunnel name` *`string`*

### Description

Optional; Provides a name for the tunnel. The name must be unique for all existing tunnels. The `no` form of the command removes the name for the tunnel.

### Usage

If `disable` is specified, the tunnel configuration is not removed. To optionally configure a name for the tunnel, enter this command in **tunnel** context.

### Example input

Creating, enabling, and naming a tunnel:

```
Switch(config)# interface tunnel 3
Switch(tunnel-3)# tunnel enable
Switch(tunnel-3)# tunnel name Redtunnel
```

## tunnel mode

### Syntax

`tunnel mode [6in4|unspecified]`

`no tunnel mode [6in4|unspecified]`

### Description

Configures the type of tunnel.

### Options

**`6in4`**

IPv6 packets encapsulated and transported over an IPv4 network.

**`unspecified`**

Set the tunnel to be unspecified.

### Example input

Configuring tunnel mode

```
Switch(tunnel-3)# tunnel mode 6in4
```

## tunnel source

When encapsulating a packet, the source IP address is used in the encapsulating IPv4 header. When decapsulating a packet, this address is matched against the destination IP address in the encapsulating IPv4 header to determine if the packet was received on a valid, configured tunnel. The command is executed in **tunnel** context.

### Syntax

`tunnel source [ipv4-addr|ipv6-addr]`

```
no tunnel source [ipv4-addr|ipv6-addr]
```

### Description

Configures the IPv4 or IPv6 address of the source (local) end of the tunnel. Must not be the same address as the tunnel destination.

### Usage

Tunnel mode must be configured before tunnel source.

You cannot configure the same source and destination address pair on more than one tunnel interface.

## tunnel destination

When an encapsulating packet is sent into a tunnel, the tunnel destination address is used in the encapsulating IPv4 header. When decapsulating a packet, this address is matched against the source IP address in the encapsulating IPv4 header to determine if the packet was received on a valid, configured tunnel. The command is executed in **tunnel** context.

### Syntax

```
tunnel destination [ipv4-addr|ipv6-addr]
```

```
no tunnel destination [ipv4-addr|ipv6-addr]
```

### Description

Configures the IPv4 or IPv6 address of the remote end of the tunnel. Must not be the same address as the tunnel source. Tunnel mode must be configured before tunnel destination.

### Example input

Configuring Destination and Source Addresses

```
Switch(tunnel-3)# tunnel source 20.30.30.3
Switch(tunnel-3)# tunnel destination 10.20.20.2
```

## tunnel mtu

### Syntax

tunnel mtu [ *1280 - 9198* ]

no tunnel mtu [ *1280 - 9198* ]

### Description

Configures the static MTU for the tunnel. Default: 1280

### Example

Configuring a Static MTU

```
Switch(tunnel-3)# tunnel mtu 1500
```

## tunnel tos

To configure a value to use for the TOS byte in the encapsulating IPv4 header when encapsulating a packet to send over the tunnel interface, enter this command in **tunnel** context. If no value is specified, the value of the TOS byte is determined from the traffic class field of the IPv6 header.

### Syntax

tunnel tos [ *0 - 63*|*copy*|*use-gos* ]

no tunnel tos [ *0 - 63*|*copy*|*use-gos* ]

### Description

Configures a value to use for the TOS byte.

**Options**

***0 - 63***

Range of static values.

**`copy`**

The TOS bits are copied from the IPv6 header. This is the default.

**`use-qos`**

Use the value returned by the QoS classifier.

**Example input**

Configuring a value for TOS

```
Switch(tunnel-3)# tunnel tos use-qos
```

## tunnel ttl

Use this command to configure the TTL in the encapsulating IPv4 header when encapsulating a packet to send over the tunnel. Enter this command in **tunnel** context.

**Syntax**

`tunnel ttl [`*`0 - 255|copy`*`]`

`no tunnel ttl [`*`0 - 255|copy`*`]`

**Description**

Configures the time-to-live. Default : 64 seconds.

**Options**

**`copy`**

When specified, the value of the TTL field from the IPv6 header is used in the IPv4 header.

**Example input**

Configuring a TTL for the Packet

```
Switch(tunnel-3)# tunnel ttl 100
```

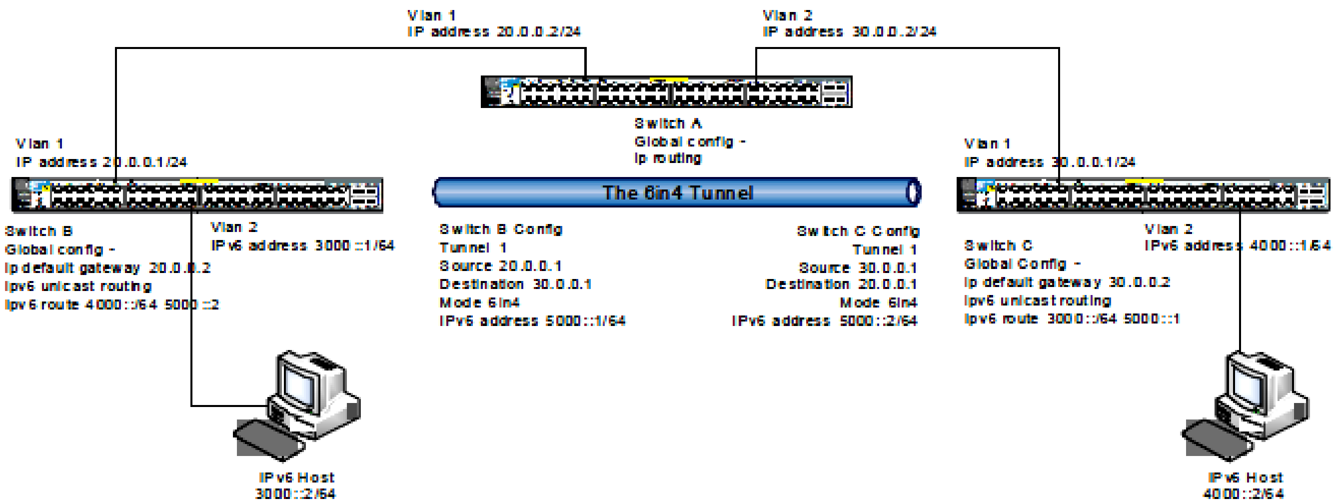
# Manual 6in4 tunneling

This example creates an IPv6 6in4 tunnel, which allows IPv6 hosts in one network to exchange IPv6 data with hosts in another IPv6 network by using the IPv4 tunneling infrastructure. In this example:

- All IPv6 traffic between hosts in the 3000::/64 and the 4000::/64 networks is tunneled through Switch B and Switch C, respectively.
- Switch B and Switch C are the manually configured endpoints.

- IPv6 traffic entering the tunnel endpoints is encapsulated with an IPv4 header based on the 6in4 configuration, and then routed through the IPv4 network using Switch A as the first hop.
- Static IPv6 routes are used to route into the tunnel interfaces.

**Figure 27:** *Manual 6in4 tunneling*




## Configuring switch B for manual 6in4 tunneling

**Procedure**

1. Configure the Tunnel endpoint for Switch B

   The tunnel endpoint for Switch B is configured with:

   a. Mode-6in4
   b. Source-encapsulation information for the IPv4 header indicating the source IPv4 address
   c. Destination-encapsulation information for the IPv4 header indicating the destination IPv4 address
   d. IPv6 Address-IPv6 network assigned to the interface (much like VLAN assignments). Enables IPv6 on the interface. It is required in order to route over the interface. To configure a tunnel endpoint:

   ```
   Switch(config)# interface tunnel 1
   Switch(tunnel-1)# tunnel mode 6in4
   Switch(tunnel-1)# tunnel source 20.0.0.1
   Switch(tunnel-1)# tunnel destination 30.0.0.1
   Switch(tunnel-1)# ipv6 address 5000::1/64
   ```

   5000::1/64 is the IPv6 network assigned to the interface. Enables IPv6 on the interface.
2. Configure VLAN 2 for the IPv6 Hosts on the 3000::/64 Network

   This step configures IPv6 address 3000::1/64 on VLAN 2. When IPv6 traffic from IPv6 host 3000::2/64 that is destined for the IPv6 4000::/64 network is transported over VLAN 2, it will be routed into the tunnel.

   The command is executed in **VLAN** context.

   ```
   Switch(config)# vlan 2
   Switch(vlan-2)# ipv6 address 3000::1/64
   ```
3. Configure IPv6 Routing to Route into the Tunnel

IPv6 unicast routing is enabled. The static route is configured to route packets destined for the 4000::/64 network into the tunnel.

```
Switch(config)# ipv6 unicast routing
Switch(config)# ipv6 route 4000::/64 1 5000::2 2
```

[1] Configures static route for destinations in the 4000::/64 network

[2] IPv6 address of tunnel endpoint

4. Configure the VLAN Used to Reach the First Hop Router in the IPv4 Network

   This step configures the tunnel endpoint IPv4 address 20.0.0.1/24 (VLAN 1) in **VLAN 2** context so that traffic from VLAN 2 can reach the first hop router in the IPv4 network through the tunnel.

   The command is performed in **VLAN** context.

   ```
   Switch(vlan-2)# ip address 20.0.0.1/24
   ```

5. Configure the IPv4 Default Gateway

   The IPv4 default gateway 20.0.0.2 is configured.

   ```
   Switch(config)# ip default-gateway 20.0.0.2
   ```

## Configuring switch C for 6in4 tunneling

Configures Switch C in a manner similar to switch B, using the appropriate IPv6 and IPv4 addresses.

### Procedure

1. Configure the Tunnel endpoint for Switch C

   The tunnel endpoint for Switch C is configured with a mode, IPv4 source address, IPv4 destination address, and an IPv6 interface address.

   ```
   Switch(config)# interface tunnel 1
   Switch(tunnel-1)# tunnel mode 6in4
   Switch(tunnel-1)# tunnel source 30.0.0.1
   Switch(tunnel-1)# tunnel destination 20.0.0.1
   Switch(tunnel-1)# ipv6 address 5000::2/64
   ```

2. Configure VLAN 2 for the IPv6 Hosts on the 4000::/64 Network

   This step configures IPv6 address 4000::1/64 on VLAN 2. When IPv6 traffic from IPv6 host 4000::2/64 that is destined for the IPv6 3000::/64 network is transported over VLAN 2, it will be routed into the tunnel.

   This step is performed in **VLAN** context.

   ```
   Switch(config)# ipv6 unicast routing
   Switch(config)# ipv6 route 3000::/64 5000::1
   ```

3. Configure IPv6 Routing to Route into the Tunnel

   IPv6 unicast routing is enabled. The static route is configured to route packets destined for the 3000::/64 network into the tunnel.

   ```
   Switch(config)# vlan 2
   Switch(vlan-2)# ipv6 address 4000::1/64
   ```

   3000::/64: Configures static route for destinations in the 3000::/64 network .

   5000::1: IPv6 address of tunnel endpoint

4. Configure the VLAN Used to Reach the First Hop Router in the IPv4 Network

   This step configures the tunnel endpoint IPv4 address 30.0.0.1/24 (VLAN1) in **VLAN 2** context so that traffic from VLAN 2 can reach the first hop router in the IPv4 network through the tunnel.

The command is executed in **VLAN** context.

```
Switch(vlan-2)# ip address 30.0.0.1/24
```

5. Configure the IPv4 Default Gateway

   The default gateway 30.0.0.2 is configured.

```
Switch(config)# ip default-gateway 30.0.0.2
```

# Example: Tunneling using Policy-Based Routing (PBR)

The following example uses the configuration shown in step 5, "Configure the IPv4 Gateway," in "**Configuring switch C for 6in4 tunneling** on page 258." The routing configuration uses PBR to route into the tunnel. The configuration steps are similar to the prior example, with the addition of the PBR configuration.

## Configuring switch B for PBR-based routing

**Procedure**

1. Configure the Tunnel endpoint for Switch B

   The tunnel endpoint for Switch B is configured with a mode, source and destination, and an IPv6 address.

```
Switch(config)# interface tunnel 1
Switch(tunnel-1)# tunnel mode 6in4
Switch(tunnel-1)# tunnel source 20.0.0.1
Switch(tunnel-1)# tunnel destination 30.0.0.1
Switch(tunnel-1)# ipv6 address 5000::1/64
```

2. Configure VLAN 2 for the IPv6 Hosts on the 3000::/64 Network

   This step is performed in **VLAN** context.

```
Switch(config)# vlan 2
Switch(vlan-2)# ipv6 address 3000::1/64
```

3. Configure the VLAN Used to Reach the First Hop Router in the IPv4 Network

   This command is performed in **VLAN** context.

```
Switch(vlan-2)# ip address 20.0.0.1/24
```

4. Configure the IPv4 Default Gateway and Enable Unicast Routing

```
Switch(config)# ip default-gateway 20.0.0.2
Switch(config)# ipv6 unicast routing
```

5. Configure IPv6 PBR-Based Routing

   Execute these steps to configure the IPv6 PBR to route into the tunnel, and apply it to the inbound VLAN.

```
Switch(config)# class ipv6 PBR_Class
Switch(config-class)# match ipv6 any 4000::/64
Switch(config-class)# exit

Switch(config)# policy pbr PBR_Policy
Switch(policy-pbr)# class ipv6 PBR_Class
Switch(policy-pbr)# action interface tunnel 1
Switch(policy-pbr)# exit

Switch(config)# vlan 2
Switch(vlan-2)# service-policy PBR_Policy in
```

## Configuring switch C for PBR-based routing

**Procedure**

1. Configure the Tunnel endpoint for Switch C

   The tunnel endpoint for Switch C is configured with a mode, source and destination, and an IPv6 address.

```
Switch(config)# interface tunnel 1
Switch(tunnel-1)# tunnel mode 6in4
Switch(tunnel-1)# tunnel source 30.0.0.1
Switch(tunnel-1)# tunnel destination 20.0.0.1
Switch(tunnel-1)# ipv6 address 5000::2/64
```

2. Configure VLAN 2 for the IPv6 Hosts on the 4000::/64 Network

   This step is performed in **VLAN** context.

```
Switch(config)# vlan 2
Switch(vlan-2)# ipv6 address 4000::1/64
```

3. Configure the VLAN Used to Reach the First Hop Router in the IPv4 Network

   This step is performed in **VLAN** context.

```
Switch(vlan-2)# ip address 30.0.0.1/24
```

4. Configure the IPv4 Default Gateway and Enable Unicast Routing

```
Switch(config)# ip default-gateway 30.0.0.2
Switch(config)# ipv6 unicast routing
```

5. Configure IPv6 PBR-Based Routing

   Execute these steps to configure the IPv6 PBR-based routing to route into the tunnel, and apply it to the inbound VLAN.

```
Switch(config)# class ipv6 PBR_Class
Switch(config-class)# match ipv6 any 3000::/64
Switch(config-class)# exit

Switch(config)# policy pbr PBR_Policy
Switch(policy-pbr)# class ipv6 PBR_Class
Switch(policy-pbr)# action interface tunnel 1
Switch(policy-pbr)# exit

Switch(config)# vlan 2
Switch(vlan-2)# service-policy PBR_Policy in
```

# Viewing tunnel configuration and status

The `show interface tunnel` *`list`* command displays information about the configuration and status of the specified tunnels.

```
Switch(config)# show interface tunnel 3,4

 Tunnel Configuration :

  Tunnel                : tunnel-3
  Tunnel Name           : Redtunnel
  Tunnel Status         : Enabled
  Source Address        : 120.22.33.44
  Destination Address   : 121.23.34.2
  Mode                  : 6in4
```

```
  TOS                     : 5            IPv6    : Enabled
  TTL                     : 100          MTU     : 1282

 Current Tunnel Status :

  Tunnel State                   : Down
  Destination Address Route      : 0.0.0.0/0
  Next Hop IP                    : 15.255.128.1
  Next Hop Interface             : vlan-1
  Next Hop IP Link Status        : Up
  Source Address                 : Not Configured
  IP Datagrams Received          : 0
  IP Datagrams Transmitted       : 0

 Tunnel Configuration :

  Tunnel                   : tunnel-4
  Tunnel Name              : Blue_tunnel
  Tunnel Status            : Enabled
  Source Address           : 20.30.30.4
  Destination Address      : 10.20.20.3
  Mode                     : 6in4
  TOS                      : 10            IPv6   : Disabled
  TTL                      : 64            MTU    : 1280

Current Tunnel Status :
  Tunnel State                  : Down
  Destination Address Route     : 0.0.0.0/0
  Next Hop IP                   : 15.255.128.1
  Next Hop Interface            : vlan-1
  Next Hop IP Link Status       : Up
  Source Address                : Not Configured
```

## Show interface tunnel brief

**Brief output for tunnel configuration information**

```
Switch(config)# show interface tunnel brief

Status - Tunnel Information Brief

 Tunnel              : tunnel-3        Mode    : 6in4
 Source Address      : 120.22.33.44
 Destination Address : 121.23.34.2
 Configured Tunnel Status : Enabled  Current Tunnel Status : Down
```

## show ipv6 nd ra prefix tunnel

**IPv6 neighbor discovery prefix information for a tunnel**

This command displays IPv6 neighbor discovery prefix information for the specified tunnel.

```
Switch(tunnel-3)# show ipv6 nd ra prefix tunnel 3
```

```
 IPv6 Neighbor Discovery Prefix Information

 Tunnel Name          : Tunnel3
  IPv6 Prefix         : Default
  Valid Lifetime      : 15 days
  Preferred Lifetime  : 14 days
  On-link Flag        : On
  Autonomous Flag     : On
  Advertise Flag      : On
```

## Show IP counters for a tunnel

#### Output showing counters for a tunnel

```
Switch(config)# show ip counters tunnel 3

Address Family :    IPv4
Interface      :    Tunnel 3
IP In Datagrams Received                           : 0
IP In Octets Received                              : 0
IP In Datagrams Broadcast Received                 : 0
IP In Octets Broadcast Received                    : 0
IP In Datagrams Multicast Received                 : 0
IP In Octets Multicast Received                    : 0
IP In Datagrams Discarded Datagram Header Error    : 0
IP In Datagrams Discarded No Route                 : 0
IP In Datagrams Discarded Invalid Address          : 0
IP In Datagrams Discarded Unknown Protocol         : 0
IP In Datagrams Discarded Truncation               : 0
IP In Datagrams Discarded Processing Error         : 0
IP In Datagrams Forwarding Required                : 0
IP In Datagrams Delivery to Protocols Successful   : 0
IP Datagrams Reassembly Required                   : 0
IP Datagrams Reassembly Successful                 : 0
IP Datagrams Reassembly Failed                     : 0
IP Out Datagrams Transmitted                       : 0
IP Out Octets Transmitted                          : 0
IP Out Datagrams Broadcast Transmitted             : 0
IP Out Octets Broadcast Transmitted                : 0
IP Out Datagrams Multicast Transmitted             : 0
IP Out Octets Multicast Transmitted                : 0
IP Out Datagrams Discarded Processing Error        : 0
IP Out Datagrams Forwarded                         : 0
IP Out Datagrams Transmit Requests from Protocols  : 0
IP Out Datagrams Fragmentation Required            : 0
IP Out Datagrams Fragmentation Successful          : 0
IP Out Datagrams Fragmentation Failed              : 0
IP Out Datagrams Fragments Created                 : 0
IP In Datagrams Received                           : 0
IP In Octets Received                              : 0
IP In Datagrams Broadcast Received                 : 0
IP In Octets Broadcast Received                    : 0
IP In Datagrams Multicast Received                 : 0
```

```
IP In Octets Multicast Received                       : 0
IP In Datagrams Discarded Datagram Header Error       : 0
IP In Datagrams Discarded No Route                    : 0
IP In Datagrams Discarded Invalid Address             : 0
IP In Datagrams Discarded Unknown Protocol            : 0
IP In Datagrams Discarded Truncation                  : 0
IP In Datagrams Discarded Processing Error            : 0
IP In Datagrams Forwarding Required                   : 0
IP In Datagrams Delivery to Protocols Successful      : 0
IP Datagrams Reassembly Required                      : 0
IP Datagrams Reassembly Successful                    : 0
IP Datagrams Reassembly Failed                        : 0
IP Out Datagrams Transmitted                          : 0
IP Out Octets Transmitted                             : 0
IP Out Datagrams Broadcast Transmitted                : 0
IP Out Octets Broadcast Transmitted                   : 0
IP Out Datagrams Multicast Transmitted                : 0
IP Out Octets Multicast Transmitted                   : 0
IP Out Datagrams Discarded Processing Error           : 0
IP Out Datagrams Forwarded                            : 0
IP Out Datagrams Transmit Requests from Protocols     : 4
IP Out Datagrams Fragmentation Required               : 0
IP Out Datagrams Fragmentation Successful             : 0
IP Out Datagrams Fragmentation Failed                 : 0
IP Out Datagrams Fragments Created                    : 0
```

# Chapter 13 Service insertion

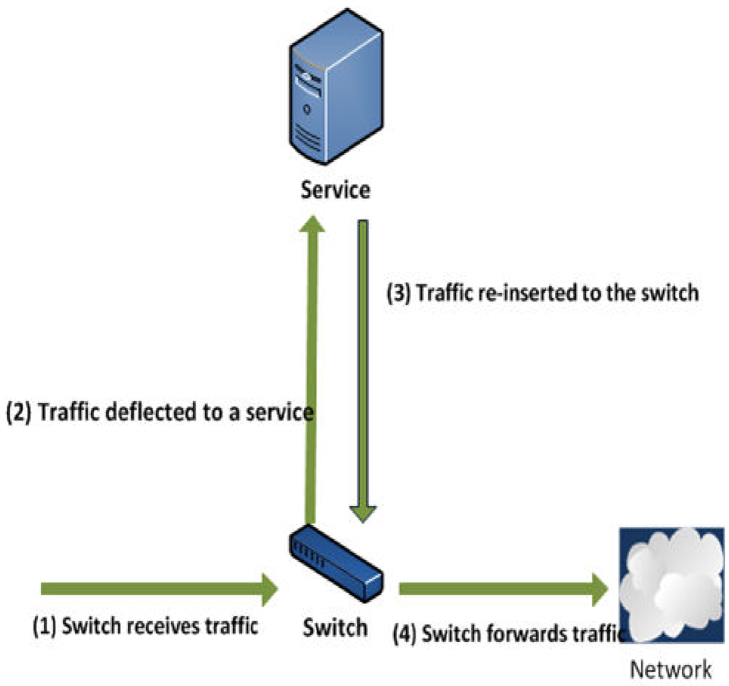
> **NOTE**
>
> Service insertion is supported on the following switches:
>
> HPE 2920 Switch
> HPE 3800 Switch
> HPE 5400 Switch
> HPE 5400R Switch
>
> Service Insertion is not supported in v1-compatible mode.

Service Insertion is transparently inserting an external service into a traffic flow or into the traffic processing pipeline:

- Flows are redirected to a service for inspection and then reinjected to the forwarding pipeline
- Possible services include IPS, HPE Network Protector SDN Application, web filtering, and traffic analyzers

Service Insertion is handled by the ASIC via a tunnel or Fast Path, and does not incur any processor processing overhead.

**Figure 28:** *Inspection Service*



NOTE

All commands previously in the Summary of commands table are indexed under the entry *Command syntax*.

The IPv6 ICMP feature enables control over the error and informational message rate for IPv6 traffic, which can help mitigate the effects of a Denial-of-service attack. Ping6 enables verification of access to a specific IPv6 device, and traceroute6 enables tracing the route to an IPv6-enabled device on the network.

# ICMP rate-limiting

ICMP rate-limiting controls the rate at which ICMPv6 generates error and informational messages for features such as:

- neighbor solicitations
- neighbor advertisements
- multicast listener discovery (MLD)
- path MTU discovery (PMTU)
- duplicate address discovery (DAD)
- neighbor unreachability detection (NUD)
- router discovery
- neighbor discovery (NDP)

ICMPv6 error message generation is enabled by default. The rate of message generation can be adjusted, or message generation can be disabled.

Controlling the frequency of ICMPv6 error messages can help to prevent DoS (Denial- of- Service) attacks. With IPv6 enabled on the switch, you can control the allowable frequency of these messages with ICMPv6 rate-limiting.

## ipv6 icmp error-interval

### Syntax

`ipv6 icmp error-interval` *`0 - 2147483647`* `[ bucket-size` *`1 - 200`* `]`

`no ipv6 icmp error-interval`

### Description

This command is executed from the global configuration level, and uses a “token bucket” method for limiting the rate of ICMP error and informational messages. Using this method, each ICMP message uses one token, and a message can be sent only if there is a token available. In the default configuration, a new token can be added every 100 milliseconds, and a maximum of 10 tokens are allowed in the token bucket. If the token bucket is full, a new token cannot be added until an existing token is used to enable sending an ICMP message. You can increase or decrease both the frequency with which used tokens can be replaced and (optionally) the number of tokens allowed to exist.

### Options

**`error-interval`**

Specifies the time interval in milliseconds between successive token adds. Increasing this value decreases the rate at which tokens can be added. A setting of `0` disables ICMP messaging.

Default : 100; Range: *0 - 2147483647*

**bucket-size**

This optional keyword specifies the maximum number of tokens allowed in the token bucket at any time. Decreasing this value decreases the maximum number of tokens that may be available at any time.

Default : 10; Range: *1 - 200*

You can change the rate at which ICMP messages are allowed by changing the error-interval with or without a corresponding change in the bucket-size.

The `no ipv6 icmp error-interval` command resets both the `error-interval` and the `bucket-size` values to their defaults.

**Usage**

Use the `show run` command to view the current ICMP error interval settings.

**Example input**

The following command limits ICMP error and informational messages to no more than 20 every 1 second:

```
Switch(config)# ipv6 icmp error-interval 1000000 bucket-size 20
```

# Ping for IPv6 (Ping6)

The Ping6 test is a point-to-point test that accepts an IPv6 address or IPv6 hostname to see if an IPv6 switch is communicating properly with another device on the same or another IPv6 network. A ping test checks the path between the switch and another device by sending IP packets (ICMP Echo Requests). To use a `ping6` command with an IPv6 hostname or fully qualified domain names, see **DNS resolver for IPv6** on page 270. You can issue single or multiple ping tests with varying repetitions and timeout periods to wait for a ping reply. Replies to each ping test are displayed on the console screen. To stop a ping test before it finishes, press `[Ctrl] [C]`.

For more information about using a ping test, see the current *Aruba-OS Switch Management and Configuration Guide* for your switch.

## ping6 ipv6–address

**Syntax**

```
ping6 [ipv6-address|hostname|switch-number][repetitions 1 - 10000][timeout 1 - 60]
[date-size 0 - 65507][data-fill 0 -1024][ source [ipv6-addr|vid]][ oobm]
```

```
ping6 [link-local-address %vlan vid|hostname|switch-number][repetitions 1 - 10000]
[timeout 1 - 60][date-size 0 - 65507][data-fill 0 -1024][ source [ipv6-addr|vid]]
[ oobm]
```

**Description**

Pings the specified IPv6 host by sending ICMP version 6 (ICMPv6) echo request packets to the specified host.

**Options**

***ipv6-address***

IPv6 address of a destination host device.

***link-local-address* %vlan *vid***

IPv6 link-local address, where `%vlan` *vid* specifies the VLAN ID number.

***hostname***

Hostname of an IPv6 host device configured on an IPv6 DNS server.

***switch-number***

Number of an IPv6-based switch that is a member of a switch stack (IPv6 subnet). Valid values: *1 - 16*.

**oobm**

For switches that have a separate out-of-band management (OOBM) port, oobm specifies that the traffic originates from the out-of-band management port.

**repetitions *1 - 10000***

Number of times that IPv6 ping packets are sent to the destination IPv6 host.

Default: 1.

**timeout *1 - 60***

Number of seconds within which a response is required from the destination host before the ping test times out.

Valid values : *1 - 60*. Default: 1 second.

**data-size *0 - 65471***

Size of data (in bytes) to be sent in ping packets.

Valid values: *0 - 65471*.

Default: 0.

**data-fill *0 - 1024***

Text string used as data in ping packets.

Range: up to 1024 alphanumeric characters; Default: 0

**source [ *ipv6-addr* | *vid* ]**

The IPv6 address of the pinging device or the VLAN ID on which the ping is being sent.

Default : 0 (no text is used).

**Example output**

IPv6 ping tests

```
Switch# ping6 fe80::2:1%vlan10
fe80:0000:0000:0000:0000:0000:0002:0001 is alive, time = 975 ms

Switch# ping6 2001:db8::a:1c:e3:3 repetitions 3
2001:0db8:0000:0000:000a:001c:00e3:0003 is alive, iteration 1, time = 15 ms
2001:0db8:0000:0000:000a:001c:00e3:0003 is alive, iteration 2, time = 15 ms
2001:0db8:0000:0000:000a:001c:00e3:0003 is alive, iteration 3, time = 15 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip (ms) min/avg/max = 15/15/15

Switch# ping6 2001:db8::214:c2ff:fe4c:e480 repetitions 3 timeout 2
2001:db8:0000:0000:0214:c2ff:fe4c:e480 is alive, iteration 1, time = 15 ms
2001:db8:0000:0000:0214:c2ff:fe4c:e480 is alive, iteration 2, time = 10 ms
2001:db8:0000:0000:0214:c2ff:fe4c:e480 is alive, iteration 3, time = 15 ms

Switch# ping6 2001:db8::10
Request timed out.
```

# Traceroute for IPv6

The `traceroute6` command enables you to trace the route from a switch to a host device that is identified by an IPv6 address or IPv6 hostname. In the command output, information on each (router) hop between the switch and the destination IPv6 address is displayed. To use a `traceroute6` command with an IPv6 hostname or fully qualified domain names, see **DNS resolver for IPv6** on page 270. Each time you perform a traceroute operation, the `traceroute` command uses the default settings unless you enter different values with each instance of the command. Replies to each traceroute operation are displayed on the console screen. To stop a traceroute operation before it finishes, press **[Ctrl] [C]**.

For more information about how to configure and use a traceroute operation, see the *Aruba-OS Switch Management and Configuration Guide*.

## traceroute6

### Syntax

```
traceroute6 [ipv6-address|hostname][minttl 1 - 255][maxttl 1 - 255][timeout 1 -
120][probes 1 - 5][source ipv6-addr|vid|loopback 0 - 7|oobm][dstport 1 - 34000]
[srcport 1 - 34000]
```

### Syntax

```
traceroute6 [link-local-address %vlan vid|hostname][minttl 1 - 255][maxttl 1 - 255]
[timeout 1 - 120][probes 1 - 5][source ipv6-addr|vid|oobm]
```

### Description

Lists the IPv6 address of each hop in the route to the specified destination host device with the time (in microseconds) required for a packet reply to be received from each next-hop device.

### Options

**`ipv6-address`**

IPv6 address of a destination host device.

**`link-local-address %vlan vlan-id`**

IPv6 link-local address, where %vlan *vlan-id* specifies the VLAN ID number.

**`hostname`**

Hostname of an IPv6 host device configured on an IPv6 DNS server.

**`oobm`**

For switches that have a separate out-of-band management (OOBM) port, `oobm` specifies that the traffic originates from the out-of-band management port.

**`minttl`**

Minimum number of hops allowed for each probe packet sent along the route. Default : 1; Range : 1 - 255

- If the minttl value is greater than the actual number of hops, the traceroute output displays only the hops equal to or greater than the configured minttl threshold value. The hops below the threshold value are not displayed.
- If the minttl value is the same as the actual number of hops, only the final hop is displayed in the command output.
- If the minttl value is less than the actual number of hops, all hops to the destination host are displayed.

**`maxttl`**

Maximum number of hops allowed for each probe packet sent along the route. Valid values: *1 - 255*. Default: 30

If the `maxttl` value is less than the actual number of hops required to reach the host, the traceroute output displays only the IPv6 addresses of the hops detected by the configured `maxttl` value.

**`timeout`**

Number of seconds within which a response is required from the IPv6 device at each hop in the route to the destination host before the traceroute operation times out.

Default : 5 seconds; Range : *1 - 120*

**`probes`**

Number of times a traceroute is performed to locate the IPv6 device at any hop in the route to the specified host before the operation times out.

Default : 3; Range: *1 - 5*

**`source [ ipv6-addr | vid ]`**

The source IPv6 address or VLAN of the traceroute device or the VLAN ID on which the traceroute packet is being sent.

**`dstport`** ***1 - 34000***

Destination port.

**`srcport`** ***1 - 34000***

Source port.

**Example output**

IPv6 traceroute probes

```
Switch# traceroute6 2001:db8::10
traceroute to 2001:db8::10
      1 hop min, 30 hops max, 5 sec. timeout, 3 probes
 1 2001:db8::a:1c:e3:3           0 ms   0 ms   0 ms
 2 2001:db8:0:7::5               7 ms   3 ms   0 ms
 3 2001:db8::214:c2ff:fe4c:e480  0 ms   1 ms   0 ms
 4 2001:db8::10                  0 ms   1 ms   0 ms
```

**First three hops**

Intermediate router hops with the time (in milliseconds) for the switch to receive a response from each of the three probes sent to each router.

**Last hop**

Destination IPv6 address

```
Switch# traceroute6 2001:db8::10 maxttl 7
traceroute to fe80::1:2:3:4
      1 hop min, 7 hops max, 5 sec. timeout, 3 probes
1 2001:db8::a:1c:e3:3           0 ms   0 ms   0 ms
2 2001:db8:0:7::5               0 ms   0 ms   0 ms
3 * 2001:db8::214:c2ff:fe4c:e480 *
4 * * *
5 * * *
6 * * *
7 * * *
```

At hop 3, the first and third probes timed out, but the second probe reached the router. Each timed-out probe is displayed with an asterisk (*). The four remaining probes within the configured seven-hop maximum (maxttl) also timed out without finding a next-hop router or the destination IPv6 address.

# DNS resolver for IPv6

The DNS (DNS) resolver is designed for local network domains where it enables use of a hostname or fully qualified domain name to support DNS-compatible commands from the switch. Beginning with software release K.13.01, DNS operation supports these features:

- dual-stack operation: IPv6 and IPv4 DNS resolution
- DNS-compatible commands: ping, ping6, traceroute, and traceroute6
- multiple, prioritized DNS servers (IPv4 and IPv6)

## DNS configuration

You can configure up to three addresses for DNS servers in the same or different domains. However, you can configure only one domain name suffix. This means that a fully qualified domain name must be used to resolve addresses for hosts that do not reside in the same domain as the one you configure with this command. That is, if the domain name suffix and the address of a DNS server for that same domain are both configured on the switch, then you need to enter only the hostname of the desired target when executing a command that supports DNS operation. But if the DNS server used to resolve the hostname for the desired target is in a different domain than the domain configured with this command, then you need to enter the fully qualified domain name for the target.

NOTE

This section describes the commands for configuring DNS operation for IPv6 DNS applications. For further information and examples on using the DNS feature, see “DNS Resolver” in appendix, “Troubleshooting”, in the current *Management and Configuration Guide* for your switch.

### ip dns server-address priority

**Syntax**

`ip dns server-address priority` *`1 - 3 ip-addr`* `[ oobm ]`

`no ip dns server-address priority` *`1 - 3 ip-addr`* `[ oobm ]`

**Description**

Used at the global config level to configure the address and priority of a DNS server. Allows for configuring up to three servers providing DNS service. (The servers must all be accessible to the switch.) The command allows both IPv4 and IPv6 servers in any combination and any order of priority.

The `no` form of the command removes the specified address from the server address list configured on the switch.

**Options**

Used at the global config level to configure the address and priority of a DNS server. Allows for configuring up to three servers providing DNS service. (The servers must all be accessible to the switch.) The command allows both IPv4 and IPv6 servers in any combination and any order of priority.

**`priority`** ***`1 - 3`***

Identifies the order in which the specified DNS server will be accessed by a DNS resolution attempt. A resolution attempt tries each configured DNS server address, in ascending order of priority, until the attempt is successful or all configured server options have been tried and failed. To change the priority of an existing server option, you must remove the option from the switch configuration and re-enter it with the new priority. If another server address is configured for the new priority, you must also remove that address from the configuration before reassigning its priority to another address.

**`ip-addr`**

Specifies the address of an IPv6 or IPv4 DNS server.

**oobm**

For switches that have a separate out-of-band management (OOBM) port, this parameter specifies that communication with the DNS server goes through that OOBM port.

### ip dns domain-name

**Syntax**

`ip dns domain-name` *`domain-name-suffix`*

`no ip dns domain-name` *`domain-name-suffix`*

**Description**

Used at the global config level to configure the domain suffix that is automatically appended to the hostname entered with a command supporting DNS operation. Configuring the domain suffix is optional if you plan to use fully qualified domain names in all cases instead of just entering hostnames. The `no` form of the command removes the configured domain name suffix.

**Example input**

Suppose you want to configure the following on the switch:

- the address 2001:db8::127:10 which identifies a DNS server in the domain named mygroup.hpnetworking.net
- a priority of 1 for the above server
- the domain suffix mygroup.hpnetworking.net

Assume that the above, configured DNS server supports an IPv6 device having a hostname of “mars-1” (and an IPv6 address of fe80::215:60ff:fe7a:adc0) in the “mygroup.hpnetworking.net” domain. In this case you can use the device's hostname alone to ping the device because the mygroup.hpnetworking.net domain has been configured as the domain name on the switch and the address of a DNS server residing in that domain is also configured on the switch. The commands for these steps are as follows:

```
Switch(config)# ip dns server priority 1 2001:db8::127:10
Switch(config)# ip dns domain-name mygroup.hpnetworking.net
Switch(config)# ping6 mars-1
fe80::215:60ff:fe7a:adc0 is alive, time = 1 ms
```

However, for the same “mars-1” device, if mygroup.hpnetworking.net was not the configured domain name, you would have to use the fully qualified domain name for the device named mars-1:

```
Switch# ping6 mars-1.mygroup.hpnetworking.net
```

For further information and examples on using the DNS feature, see “DNS Resolver” in the Troubleshooting appendix, in the current *Management and Configuration Guide* for your switch.

## Commands to view the current DNS configuration

Use the `show ip dns` command to view the current DNS server configuration.

Use the `show run` command to view both the current DNS server addresses and the current DNS domain name in the active configuration.

NOTE

In software release K.13.01, DNS addressing is not configurable from a DHCPv6 server.

# Debug/Syslog for IPv6

The Debug/System logging (Syslog) for IPv6 feature provides logging functions similar to those of the IPv4 version, allowing you to record IPv4 and IPv6 Event Log and debug messages on a remote device to troubleshoot switch or network operation. For example, you can send messages about routing mis-configurations and other network protocol details to an external device, and later use them to debug network-level problems.

NOTE: This section describes the commands for Debug/Syslog configuration in an IPv6 environment. For information on using the Debug/Syslog feature in an IPv4 environment, see "Debug/Syslog Operation" in the Troubleshooting appendix in the current *Management and Configuration Guide* for your switch.

## Configuring debug and event log messaging

To specify the types of debug and Event Log messages that you want to send to an external device:

- Use the `debug ipv6` command to send messaging reports for the following types of switch events:
  - DHCPv6 client
  - DHCPv6 relay
  - forwarding
  - neighbor discovery
  - OSPFv3
  - packets
- Use the `logging [ severity severity-level | system-module system-module ]` command to select a subset of Event Log messages to send to an external device for debugging purposes according to:
  - Severity level
  - System module

## debug ipv6

### Syntax

`debug ipv6 [ debug-type ]`

`no debug ipv6 [ debug-type ]`

### Description

Configures the types of IPv6 messages that are sent to Syslog servers or other configured debug destinations.

### Options

Debug type is any of the following event types:

**(none)**

all IPv6 events

**dhcpv6-client [ events | packets ]**

one of the following IPv6 client debug message types

**events**

DHCPv6 client events

**packets**

DHCPv6 client packets

**dhcpv6-relay [ events | packets ]**

one of the following IPv6 relay debug message types

**events**

DHCPv6 relay events

**packets**

DHCPv6 relay packets

**forwarding**

IPv6 forwarding events

**nd**

IPv6 neighbor discovery events

**ospf3**

one of the following OSPFv3 message types:

**(none)**

all OSPFv3 debug events

**adj**

adjacency changes

**event**

events

**flood**

flooding

**lsa-generation**

link state advertisement generation

**packet**

one of the following OSPFv3 packet types:

**(none)**

all OSPFv3 packets sent or received

**DD**

DD packets sent or received

**Hello**

Hello packets sent or received

**LSA**

LSA packets sent or received

**LSR**

LSR packets sent or received

**LSU**

LSU packets sent or received

**retransmission**

retransmissions

**spf**

SPF computations

**packet**

all IPv6 packet messages

## debug destination

An IPv6-based debug/syslog destination device can be a Syslog server (up to six maximum) and/or a console session.

### Syntax

`debug destination [ logging | session | buffer ]`

### Description

command to enable (and disable) Syslog messaging on a Syslog server or to a CLI session for the debug message types configured with the `debug` and `logging` commands (see **Configuring debug and event log messaging** on page 272).

### Options

**`debug destination logging`**

Enables the configured debug message types to be sent to Syslog servers configured with the `logging [ syslog-ipv4addr | syslog-ipv6-addr ]` command.

**`debug destination session`**

Enables the configured debug message types to be sent to the CLI session that executed this command. The session can be on any one terminal emulation device with serial, Telnet, or SSH access to the CLI at the Manager level prompt.

**`debug destination buffer`**

Enables the configured debug message types to be sent to a buffer in switch memory.

## logging syslog-ipv4–address

### Syntax

`logging` *`syslog-ipv4-address`*

`no logging` *`syslog-ipv4-address`*

### Description

Enables or disables Syslog messaging to the specified IPv4 address. You can configure up to six addresses. If you configure an address when none are already configured, this command enables destination logging (Syslog) and the Event debug type. Therefore, at a minimum, the switch begins sending Event Log messages to configured Syslog servers. If other debug message types are configured, they are also sent to the Syslog server.

### Options

**`no logging`**

Removes all currently configured Syslog logging destinations from the running configuration.

**`no logging`** ***`syslog-ipv4-address`***

Removes only the specified Syslog logging destination from the running configuration.

## show debug

### Syntax

`show debug`

### Description

Use this command to display the currently configured settings for:

- Debug message types and Event Log message filters (severity level and system module) sent to debug destinations
- IPv4/IPv6 debug destinations (Syslog servers or CLI session) and Syslog server facility to be used

**Example output**

Syslog Configuration to Receive Event Log Messages at Specified System Module and Severity Levels on an IPv6 Syslog Server

```
Switch(config)# show debug
Debug Logging
Destination: None
Enabled debug types:
None are enabled
```

Displays the default debug configuration when no Syslog server IP addresses or debug types are configured.

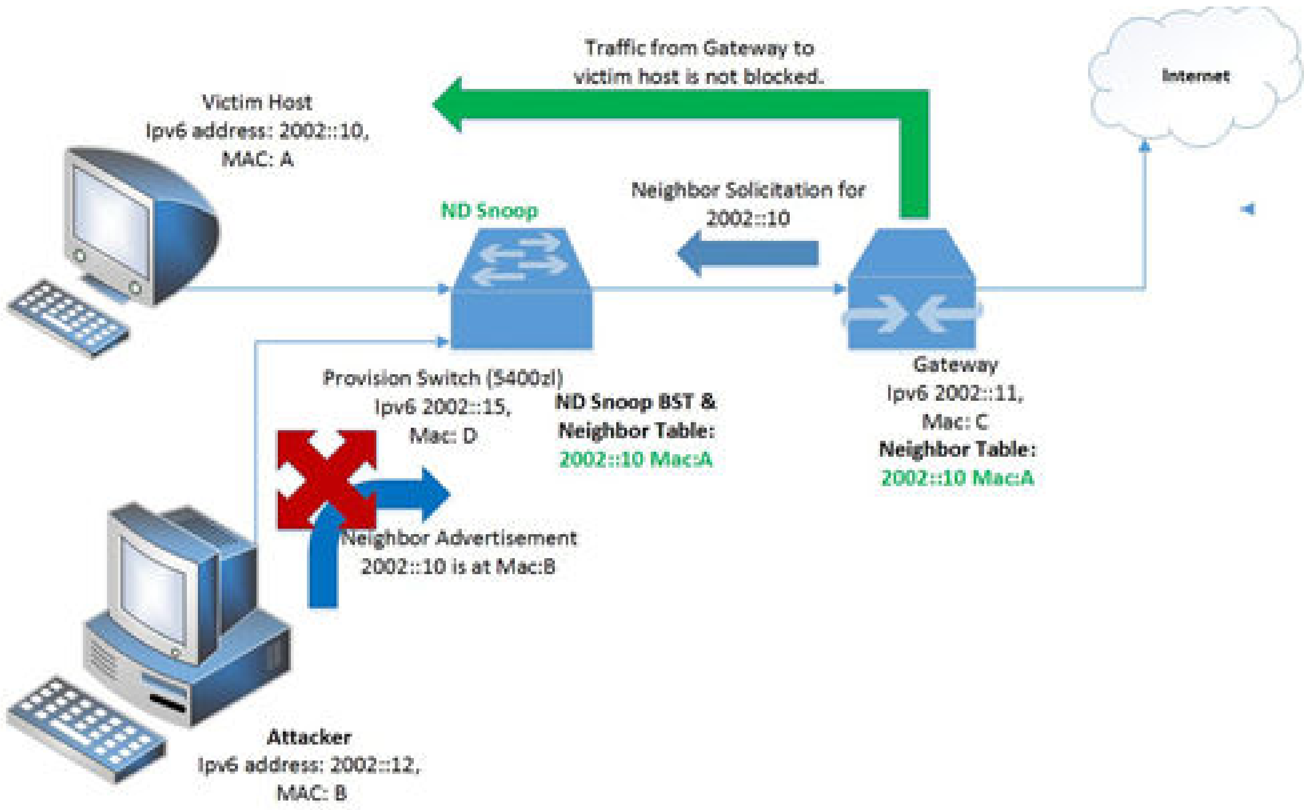
```
Switch(config)# logging fe80:215:60ff:fe7a:adc0
Switch(config)# write memory
Switch(config)# show debug

Debug Logging
Destination:
Logging --
fe80:215:60ff:fe7a:adc0
Facility=user
Severity=debug
System module=all-pass
Enabled debug types:
event
```

When you configure a Syslog IPv6 address with the logging command, by default, the switch enables debug messaging to the Syslog address and the user facility on the Syslog server, and sends Event Log messages of all severity levels from all system modules.

# Overview of IPv6 network defense ND snooping and detection

Enabling the ND Snooping feature on your switches prevents ND attacks. ND Snooping does not just snoop but also detect attacks by default. ND Snooping drops invalid ND packets and, together with DIPLDv6, blocks data traffic from invalid hosts.

**Figure 29:** *ND Snooping enabled on a device*




ND Snooping provides the following:

- Drops ND packets if the Ethernet source MAC address is mismatched with the one contained in the ND packet's link-layer address field.
- Drops ND packets where the global IPv6 address in the source address field is mismatched with the ND Snooping prefix filter table.
- Drops ND packets where the global IPv6 address or the link-local IPv6 address in the source IP address field is mismatched with the ND Snooping binding table.
- Drops the router advertisement on the untrusted ports. This is similar to RA Guard. To block RAs and RRs on a particular port using the RA Guard feature, RA Guard must be enabled on each of those ports. When ND

Snooping is enabled with a trusted port configuration, RAs and RRs are dropped on all ports that ND Snooping enabled VLAN, other than the trusted port.

- Dynamic IPv6 lockdown is performed for ND snooping entries. Based on the DAD NS received from the hosts by the switch, ND Snooping entries are programmed to the SAVI BST and the hardware (as allowed). Hence, data packets from invalid hosts and transit traffic are blocked.

DIPLDV6 is an existing feature that adds a static or a dynamic binding based on the dsnoopv6 database. RA guard is an existing feature that can be configured per port on which the router advertisements and router redirects are blocked. Together with DIPLDv6 and RA guard, NDSnoop provides a high level of Network Defense at the hands of the Network administrator and makes the network more secure.

## ICMPv6 messages

The IPv6 Neighbor Discovery Protocol (ND) consists of five types of ICMPv6 messages:

- Neighbor Solicitation (NS) - An IPv6 node (a host or network device running IPv6) sends NS packets to obtain the link-layer addresses of its neighbors and to detect neighbor reachability and duplicate addresses.
- Neighbor Advertisement (NA) - An IPv6 host sends an NA packet in response to an NS packet. An IPv6 node also sends NA packets when the link-layer topology changes.
- Router Solicitation (RS) - When an IPv6 node starts, it sends an RS packet to a router to request prefixes and other configuration information, and waits for the router to respond with an RA packet.
- Router Advertisement (RA) - A router periodically advertises RA packets, including network configurations such as network prefix to IPv6 nodes. The router also returns RA packets as the responses to RS packets.
- Redirect (RR) - When detecting that the inbound interface and outbound interface of a packet are the same, a router sends a Redirect packet to request the IPv6 node to select a better next hop address.

These ICMPv6 messages help to achieve these five functions:

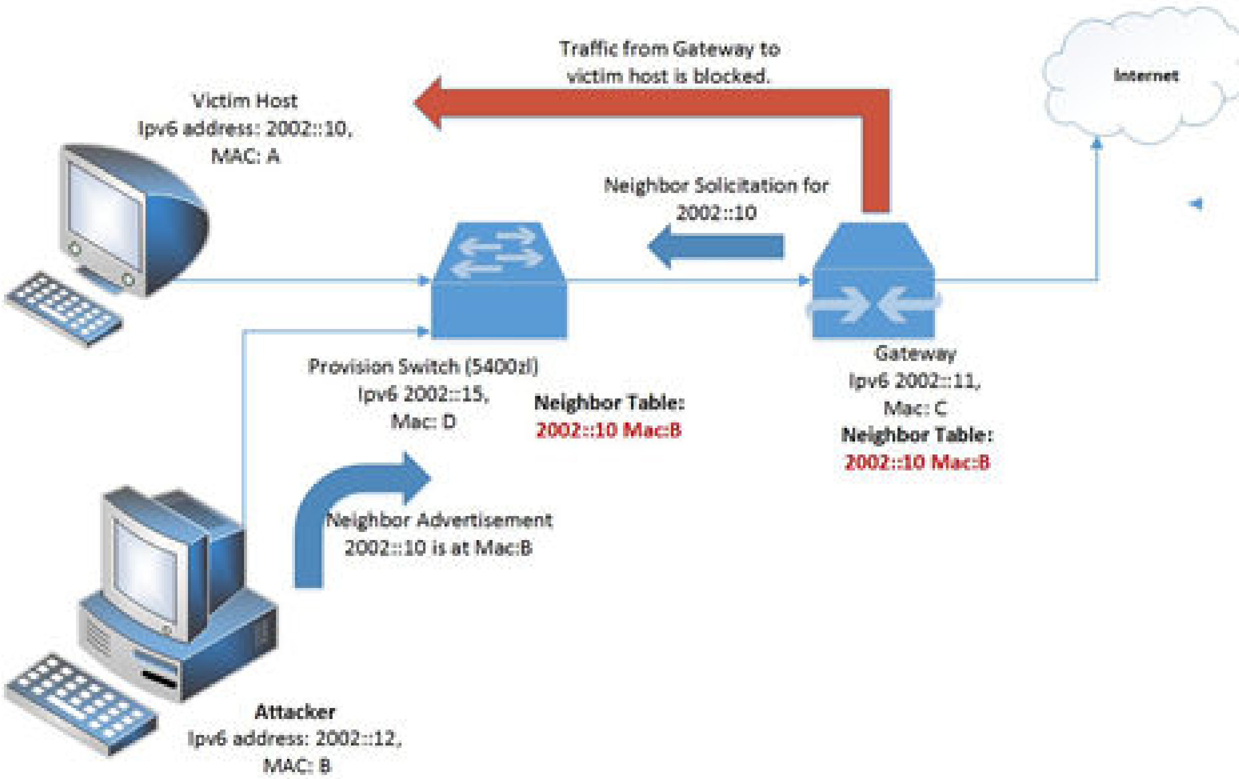
- address resolution
- neighbor reachability detection
- router/prefix discovery
- address auto-configuration
- redirection

## ND attacks

ND messages are easy to be exploited by the spoofers/attackers in the IPv6 network if there are no security mechanisms. The attackers could send forged ND packets to redirect the traffic meant for a host from a router/gateway to them. The ND attacks include the following types:

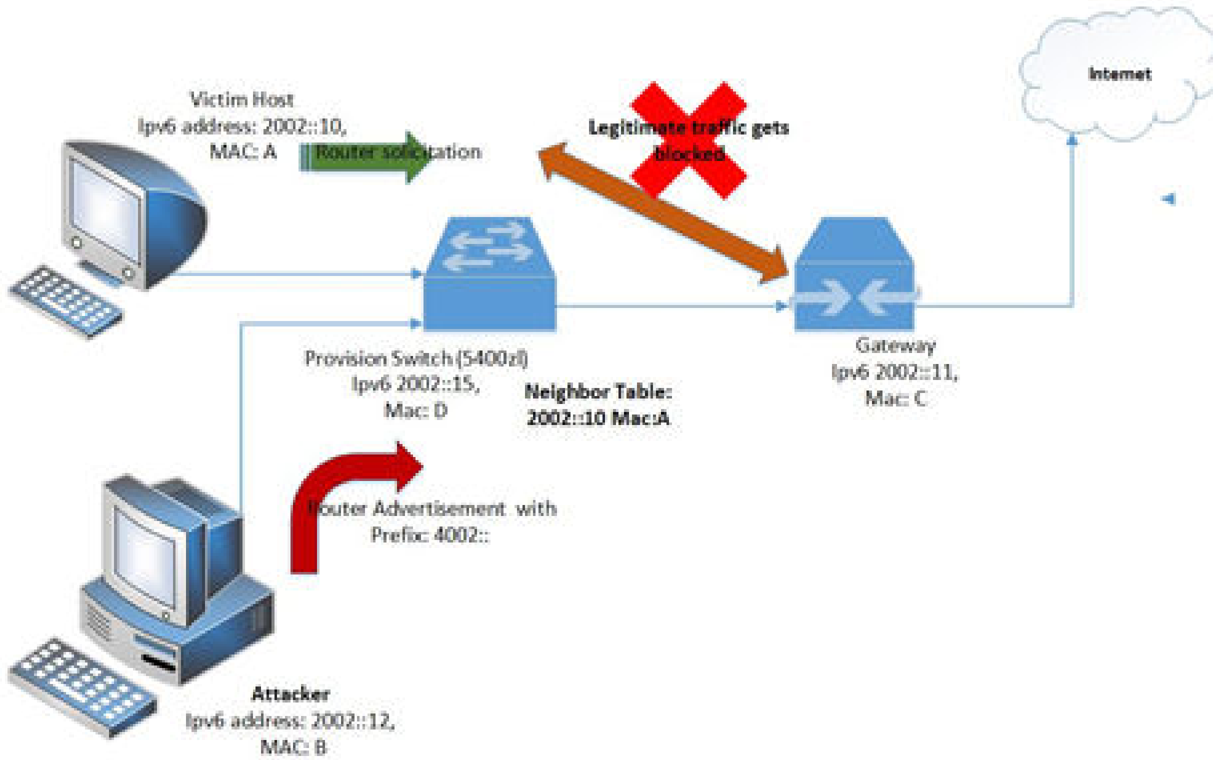
- Address Spoofing AttackAn attacker could send forged NS/NA packets with the IPv6 address of a victim host. The ND entry maintained by the gateway and other hosts for the victim host will be updated with the wrong address information (of that of the attacker). As a result, all packets intended for the victim host will be sent to the attacking host rather than the victim host. In figure 14, the gateway sends a Neighbor Solicitation for the IPv6 address 2002::10. An attacker could send a Neighbor Advertisement as a reply causing the gateway to learn 2002::10 is at Mac B. The traffic gets redirected to the attacker.There can be other kind of DOS Attacks where the spoofer sends Neighbor Advertisement packets with different source IPv6 addresses to fill up the neighbor cache of the device, resulting in no room for valid clients.

**Figure 30:** *ND attack on device*




- RA AttackAn attacker could send forged RA packets with the IPv6 address of a victim gateway. This can cause all hosts attached to the victim gateway to maintain incorrect IPv6 configuration parameters and ND entries.In Figure 15, when the victim host sends a router solicitation, the attacker could send a route advertisement as a reply causing the victim host to receive the wrong network parameters. Hence the legitimate traffic to the victim hosts gets blocked.

**Figure 31:** *RA attack on device*




# Commands

These commands configure ND Snooping and ND Inspection as well as their related attributes.

NOTE

Run these commands from within the switch configuration.

## ipv6 nd mac-check

**Syntax**

```
ipv6 nd mac-check
```

**Description**

Enable global administrative status of ND MAC-check. The MAC-check is not performed on any ND packets if the global administrative status is disabled. The default is disabled.

**Options**

**`mac-check`**

Enable global administrative status of ND MAC-check.

## ipv6 nd snooping

**Syntax**

`ipv6 nd snooping`

`no ipv6 nd snooping`

**Description**

Enable global administrative status of ND Snooping. ND packets are not snooped and inspected if the global administrative status of ND Snooping is disabled.

**Options**

**`snooping`**

Enable global administrative status of ND Snooping.

## ipv6 nd snooping vlan

**Syntax**

`ipv6 nd snooping vlan` *`VLAN-ID-RANGE`*

`no ipv6 nd snooping vlan` *`VLAN-ID-RANGE`*

**Description**

Enable or disable ND Snooping on a VLAN. When ND Snooping is enabled on a VLAN, the ND packets are inspected to form the ND Binding table and the invalid packets are dropped.

When `no` is specified, the ND Snooping feature is disabled on the VLANs.

Default is disabled.

**Options**

**`snooping`**

Enable the administrative status of ND Snooping.

## clear ipv6 nd snooping bindings ipv6-address

**Syntax**

`clear ipv6 nd snooping bindings ipv6-address | vlan` *`vlan-id`* `| mac` *`mad-addr`*

**Description**

Clear all ND Snooping binding entries or binding entries on the specified IPv6-address, VLAN, or MAC address.

**Options**

**`nd`**

Neighbor Discovery.

**`snooping`**

Clear ND Snooping information.

## ipv6 nd snooping prefix-list

**Syntax**

`ipv6 nd snooping prefix-list` *`IPV6-ADDR`*

**Description**

Configure IPv6 prefix-list for ND Snooping. Prefix-list allows the user to specify network prefix ranges that are allowed. The prefix-list should be configured when there are no router advertisements received by the switch.

**Options**

**Prefix-list**

Configure IPv6 prefix-list for ND Snooping. Configures network prefix ranges to be specified as allowed. The prefix-list should be configured when no router advertisements are received by the switch.

## About configuring maximum learn entries on a port

Configures the maximum number of ND Snooping entries that can be learned on a port.

If the maximum bindings value is configured before enabling ND Snooping, the limit is immediately applied and the bindings are not allowed to exceed the max-bindings value. If the max-bindings value is set after enabling ND Snooping, the following applies:

- The current bindings are greater than the max-binding value, the configuration is applied as and when the ND Snooping entry ages out.
- Current bindings are less than that of the value entered, the configuration is immediately applied.

NOTE

This is similar to the features DSNOOPv4, DSNOOPv6 in some switches.

### ipv6 nd snooping max-binding

**Syntax**

`ipv6 nd snooping max-binding` *`PORT-LIST 1-16384`*

**Description**

Configures the maximum number of ND Snooping entries that can be learned by an interface.

The default for number of interfaces learned entries is 16384.

**Options**

**`max-bindings`**

Maximum number of ND Snooping entries that can be learned on a port.

## ipv6 nd snooping trust ethernet

Allows users to configure the trusted port for the ND Snooping.

**Syntax**

`ipv6 nd snooping trust ethernet` *`PORT-LIST`*

**Description**

Configure trusted ports for ND Snooping. Routers are connected to the trusted ports. Router advertisements and Router redirects are processed in the trusted ports and dropped in the untrusted ports.

**Options**

**`snooping`**

Neighbor discovery snooping.

**`trust`**

Configure trusted ports.

## clear ipv6 nd snooping statistics

**Syntax**

```
clear ipv6 nd snooping statistics ethernet PORT-LIST
```

**Description**

Clears the IPv6 ND Snooping statistics on the given port. The statistics can also be cleared for a port list.

**Options**

**snooping**

Clears all types of statistics related to ND Snooping.

**statistics**

Clears all ND Snooping port statistics.

## snmp-server enable traps nd-snooping

**Syntax**

```
snmp-server enable traps nd-snooping | out-of-resources | violations
```

**Description**

Allows user to configure traps for ND Snooping. Enable traps for out-of-resources or for ND Snooping violations. If traps are enabled for out-of-resources, traps are sent when no hardware resources are available to apply ND Snooping. If traps are enabled for ND Snooping violations, traps are sent for any ND Snooping violations.

**Options**

**out-of-resources**

when the number of bindings exceed the maximum limit of 16384 bindings

**violations**

when an invalid ND packet is detected.

## debug ipv6 nd

**Syntax**

```
debug ipv6 nd snooping
```

**Description**

Debug messages are useful while debugging error cases. Generally enabled only during the debug session.

**Options**

**snooping**

Display all IPv6 ND Snooping messages.

# Show commands

This section describes the `show` commands used for IPv6 ND snooping.

## show ipv6 nd snooping

**Syntax**

```
show ipv6 nd snooping
```

**Description**

Shows IPv6 ND Snooping configurations.

**Example output**

Show IPv6 ND Snooping configurations

```
Switch(config)# show ipv6 nd snooping

ND Snooping Information:
  ND Snooping                  : Yes
  Enabled VLANs                : 1 100 300
  MAC check                    : Yes

               Max       Current
  Port  Trust  Bindings  Bindings
  ----  -----  --------  ---------
   1     Yes      -         -
   2     No       20*       20
   4     No       543       231
   13    No       -         3
   48    Yes      -         -

   Ports 3,5-12,14-47 are untrusted

*)The Max-Bindings limit will be enforced at a higher number
until the number of bindings drop down to the limit set on the port.
```

## show ipv6 nd snooping bindings

### Syntax

```
show ipv6 nd snooping bindings
```

### Description

Shows IPv6 ND Snooping binding entries.

### Example output

Show IPv6 ND Snooping binding entries

```
Switch(config)# show ipv6 nd snooping bindings

Mac-Address           123456-789101
IPv6-Address          3000:abbb:1234:3456:1234:1234:1234:1234
VLAN                  1000
Port                  A1
Time-Left             65535

Mac-Address           abcdef-123455
IPv6-Address          300:ab::2
VLAN                  4092
Port                  F23
Time-Left             1000
```

## show ipv6 nd snooping statistics

### Syntax

```
show ipv6 nd snooping statistics
```

### Description

Shows IPv6 ND Snooping statistics.

### Example output

Show IPv6 ND Snooping statistics.

```
Switch(config)# show ipv6 nd snooping statistics

Port  PacketType   Action   Reason                        Count
---------------------------------------------------------------
A1        RA       forward  received on trusted port       5
A1        RA       drop     received on validating port    5
A1        NS       forward  received on trusted port       5
A1        NS       forward  skipped MAC check              3
A1        NS       forward  passed ND Snooping validation
                                     checks                5
A1        NS       drop     failed verify MAC check        3
A1        NS       drop     failed on max-binding limit    6
A1        NS       drop     failed ND Snooping validation
                                     checks                5
A1        NA       forward  received on trusted port       5
A1        NA       forward  verify MAC check is skipped    5
A1        NA       forward                                 7
A1        NA       forward  passed ND Snooping validation
                                     checks                5
A1        NA       drop     failed verify MAC check        4
A1        NA       drop     failed on max-binding limit    6
A1        NA       drop     failed ND Snooping validation
                                     checks                5
```

## show snmp-server traps

**Syntax**

show snmp-server *COMMUNITY-STR*|traps

**Description**

Display information on all SNMP communities, trap receivers and SNMP response/trap source -ip policy configured on the switch. If *COMMUNITY-STR* is specified, only information for that community is displayed.

**Example output**

Show SNMP-server traps

```
Switch(config)# sh snmp-server traps

Trap Receivers
Link-Change Traps Enabled on Ports [All] : All

Traps Category                      Current Status
---------------------------         --------------
SNMP Authentication                     Extended
Password change                         Enabled
Login failures                          Enabled
Port-Security                           Enabled
Authorization Server Contact            Enabled
DHCP-Snooping                           Enabled
DHCPv6-Snooping Out of Resource         Enabled
DHCPv6-Snooping Errant Replies          Enabled
Dynamic ARP Protection                  Enabled
Dynamic IP Lockdown                     Enabled
Dynamic IPv6 Lockdown Out of Resource   Enabled
Dynamic IPv6 Lockdown Violations        Enabled
ND Snooping Out of Resources            Enabled
```

```
Startup Config change                   Disabled
Running Config Change                   Disabled
MAC address table changes               Disabled
MAC Address Count                       Disabled

Address         Community      Events   Type  Retry  Time out
---------- --------------- -------- ----- ------ --------
Excluded MIBs
```

# Change to existing command: show qos resources

**Syntax**

`show qos resources`

**Description**

Show policy engine resource usage and availability.

**Options**

**`resources`**

Show policy engine resource usage and availability.

**Example output**

Show policy engine resource usage and availability

```
Switch# show qos resources

Resource usage in Policy Enforcement Engine

      | Rules |Rules Used
Ports | Avail |ACL |QoS |IDM |VT |Mirr |PBR |OF |Other |
------+-------+----+----+----+---+-----+----+---+------|
1-24  |  3055 |  0 |  0 |  0 | 0 |  0  |  0 | 0 |    0 |
25-48 |  3055 |  0 |  0 |  0 | 0 |  0  |  0 | 0 |    0 |

      | Meters|Meters Used
Ports | Avail |ACL |QoS |IDM |VT |Mirr |PBR |OF |Other  |
------+-------+----+----+----+---+-----+----+---+-------|
1-24  |   255 |    |  0 |  0 |   |     |    | 0 |     0 |
25-48 |   255 |    |  0 |  0 |   |     |    | 0 |     0 |

0 of 8 Policy Engine management resources used.

 Key:
 ACL = Access Control Lists
 QoS = Device & Application Port Priority, QoS Policies, ICMP rate limits
 IDM = Identity Driven Management
 VT  = Virus Throttling blocks
 Mirr = Mirror Policies, Remote Intelligent Mirror endpoints
 PBR = Policy Based Routing Policies
 OF = OpenFlow
 Other = Management VLAN, DHCP Snooping, DHCPv6 Snooping, ND Snooping,
         ARP Protection, Jumbo IP-MTU, Transparent Mode, RA Guard,
         Control Plane Protection, Service Tunnel.
 Resource usage includes resources actually in use, or reserved for future
 use by the listed feature.  Internal dedicated-purpose resources, such as
 port bandwidth limits or VLAN QoS priority, are not included.
```

# Validation rules for QOS resources

| Validation | Error/Warning/Prompt |
|---|---|
| If the h/w resources are not available for copying the ND packets to the CPU. | Cannot enable ND Snooping because hardware resources are unavailable. |
| If ND packets are received with VRRP MAC, they may get dropped if the ND MAC-check is enabled. | Enabling ND MAC-check may drop ND packets with the VRRP MAC in the ND packets. |
| If the VLAN that is being configured for ND Snooping is an SVLAN and the bridge mode is mixed mode. | ND Snooping is not supported on SVLANs and SVLAN ports in QinQ mixed VLAN mode. |
| If an SVLAN is being created with VLAN ID that corresponds to an ND Snooping enabled VLAN. | Cannot create SVLAN with VLAN ID %d because ND Snooping is enabled on this VLAN. |
| If the VLAN which is being configured for ND Snooping that has a Smart Link enabled port. | Cannot configure ND Snooping on a VLAN containing Smart Link ports. |
| If Smart Link is being configured on a port which is a part of ND Snooping VLAN. | Cannot configure the Smart Link feature on a port that belongs to the ND Snooping enabled VLAN. |
| If number of snooped VLAN counts is greater than max supported | ND Snooping can be enabled on %d VLANs only. |
| If ND Snooping is not enabled globally. | ND Snooping is disabled globally. |
| If the port is not a part of an ND Snooping enabled VLAN. | Port %s is not a part of an ND Snooping VLAN. |
| Check if ND Snooping is enabled globally | ND Snooping is disabled. |
| Verify if ND Snooping is enabled globally. | ND Snooping is disabled globally. |
| If an invalid address is being configured as prefix. | Invalid prefix: %s. |
| If the limit on configuring the number of prefixes had reached. | Cannot configure more than %d prefixes. |
| Verify whether the VLAN ID is proper | Invalid input: %s. |
| Verify whether the IPv6 address is valid | Invalid input: %s. |
| Verify if ND Snooping is enabled globally. | ND Snooping is disabled globally. |
| Verify if ND Snooping is enabled on a VLAN when VLAN option is provided. | ND Snooping is disabled on VLAN %d. |
| If any other addresses other than global unicast address or link local addresses are entered | Invalid IPv6 address. |

*Table Continued*

| Validation | Error/Warning/Prompt |
|---|---|
| If a given IPv6-address or VLAN or MAC address is not found in the binding table. | Binding for %s not found. %s -- for IPv6 address. %d – for VLAN ID %d – for MAC address |
| Verify max-bindings value entered is in the range. | Invalid input: %d. |
| Verify the interface specified is available in Hardware. | Module not present for port or invalid port: %s. |
| If ND Snooping is already configured before entering the command and current bindings are greater than the value being set. | Existing bindings %d are more than the max-bindings being configured, and the maximum limit will be applied once the number of existing bindings falls below this limit. |
| If the value is being configured for a port which is not a part of a nd-snooped VLAN. | Port %s is not a part of an ND Snooping enabled VLAN. |
| If the max-binding value is being set for a Dynamic trunk. | Cannot configure ND Snooping Max-Binding on a port when Dynamic Trunking is enabled on that port. |
| If the number of static bindings is greater than the max-binding value being set. | Cannot configure the maximum binding value because the number of static bindings on the port exceeds the maximum binding value. |
| If a port on which max-binding is enabled is being put into a trunk. | Cannot add a port to a trunk group when ND Snooping Max-Binding is configured on that port. |
| If a trunk has max-bindings configured on it and the trunk is being removed. | Cannot remove the port %s from the trunk group because ND Snooping Max-Binding is configured on the trunk and removing the port will delete the trunk. |
| If DT trunk is being configured on a max-binding enabled port. | Cannot configure Distributed Trunking on a port when ND Snooping Max-Binding is configured on that port. |
| Verify whether the port exists in the device. | Module not present for port or invalid port: <port-list>. |
| If the port is a part of an SVLAN and the bridge mode is mixed mode. | Port %s cannot be configured as trusted port as it is part of an SVLAN in QinQ mixed VLAN mode. |
| If the port is not a part of an ND Snooping enabled VLAN. | Port %s is not a part of an ND Snooping enabled VLAN. |
| If a Dynamic trunk is being configured as an ND Snooping trusted port. | Cannot configure a port as an ND Snooping trusted port when Dynamic Trunking is enabled on that port. |
| If trusted attribute is being configured on a port on which max-binding has been already configured. | Disable max-binding feature configured on the port before configuring it as a trusted port. |
| If a Smart Link port is being configured as an ND Snooping trusted port. | Cannot configure a Smart Link port as an ND Snooping trusted port. |

*Table Continued*

| Validation | Error/Warning/Prompt |
|---|---|
| If an ND Snooping trusted port is being configured as a Smart Link port. | Cannot configure an ND Snooping trusted port as a Smart Link port. |
| If RA Guard enabled port is being configured as an ND Snooping trusted port. | Cannot configure a port as an ND Snooping trusted port when RA Guard is enabled on that port. |
| If ND Snooping is not enabled globally. | ND Snooping is disabled globally. |
| If RA Guard is being enabled on an ND Snooping trusted port. | Cannot configure RA Guard on an ND Snooping trusted port. |

# Event log messages for ND snooping

| Event | Message |
|---|---|
| RMON_NDSNOOP_RA_UNTRUSTED | `%s: RA message received on the untrusted port %s is dropped.` |
| RMON_NDSNOOP_RR_UNTRUSTED | `%s: RR message received on the untrusted port %s is dropped.` |
| RMON_NDSNOOP_BIND_MISMATCH | `%s: Illegal ND packet of type %s from %s on port %s is dropped; Reason %s.` |
| RMON_NDSNOOP_PREFIX_MISMATCH | `%s: Prefix mismatch. ND packet of type %s from %s on port %s is dropped.` |
| RMON_NDSNOOP_MAC_CHECK_FAIL | `%s: ND packet of type %s is dropped because the Ethernet MAC address of the packet %02X%02X%02X-%02X%02X%02X did not match the Source link layer address of the packet %02X%02X%02X-%02X%02X%02X.` |
| RMON_NDSNOOP_TABLE_FULL | `%s: Unable to add the ND Snooping binding for ND packet of type %s with the IPv6 address %s as the binding table is full.` |
| RMON_NDSNOOP_MAX_BIND_LIMIT_FULL | `%s: The max-binding limit is reached on the port %s. Unable to add binding for the ND packet of type: %s with the IPv6 address %s.` |

# Debug messages for ND snooping

| Event | Debug message |
|---|---|
| The ND Snooping BST becomes full, further ND packets are dropped. | `Unable to add binding for %s, %02x%02x %02x-%02x%02x%02x on port %s. ND Snooping BST is full.` |
| ND validation fails (packets are received on which they are not expected to – RA/RR could be received on untrusted port). | `Dropping packet as validation failed, reason %s.` |
| ND Snooping is disabled globally. | `ND Snooping disabled globally.` |
| ND Snooping is disabled per VLAN | `ND Snooping disabled on VLAN %s.` |
| A DIPLDv6 enabled port is removed from an ND Snooping enabled VLAN. | `Port %s is removed from a ND Snooping enabled VLAN.` |
| ND Snooping is disabled globally while DIPLDv6 is enabled. | `ND Snooping disabled globally, dynamic IPv6 lockdown is enabled.` |
| ND Snooping is disabled on a particular VLAN while DIPLDv6 is enabled. | `ND Snooping disabled on VLAN %s, dynamic IPv6 lockdown is enabled.` |
| A port moved from SAVI-Trusted to validating port. | `Port %s is now a validating port.` |
| Adding a port to a trunk for which ND Snooping is already enabled. | `Unable to add port %s to trunk, ND Snooping is enabled on it.` |
| Enabling ND Snooping on a port that is added to a trunk. | `Unable to configure ND Snooping on port %s that is a part of a trunk.` |
| Max-binding limit is reached on a Port | `Max-binding limit reached on port %s.` |

# Websites

**Networking Websites**

**Hewlett Packard Enterprise Networking Information Library**

**www.hpe.com/networking/resourcefinder**

**Hewlett Packard Enterprise Networking Software**

**www.hpe.com/networking/software**

**Hewlett Packard Enterprise Networking website**

**www.hpe.com/info/networking**

**Hewlett Packard Enterprise My Networking website**

**www.hpe.com/networking/support**

**Hewlett Packard Enterprise My Networking Portal**

**www.hpe.com/networking/mynetworking**

**Hewlett Packard Enterprise Networking Warranty**

**www.hpe.com/networking/warranty**

**General websites**

**Hewlett Packard Enterprise Information Library**

**www.hpe.com/info/EIL**

For additional websites, see **Support and other resources**.

# Accessing Hewlett Packard Enterprise Support

- For live assistance, go to the Contact Hewlett Packard Enterprise Worldwide website:

  **http://www.hpe.com/assistance**
- To access documentation and support services, go to the Hewlett Packard Enterprise Support Center website:

  **http://www.hpe.com/support/hpesc**

**Information to collect**

- Technical support registration number (if applicable)
- Product name, model or version, and serial number
- Operating system name and version
- Firmware version
- Error messages
- Product-specific reports and logs
- Add-on products or components
- Third-party products or components

# Accessing updates

- Some software products provide a mechanism for accessing software updates through the product interface. Review your product documentation to identify the recommended software update method.
- To download product updates:

  **Hewlett Packard Enterprise Support Center**

  **www.hpe.com/support/hpesc**

  **Hewlett Packard Enterprise Support Center: Software downloads**

  **www.hpe.com/support/downloads**

  **Software Depot**

  **www.hpe.com/support/softwaredepot**
- To subscribe to eNewsletters and alerts:

  **www.hpe.com/support/e-updates**
- To view and update your entitlements, and to link your contracts and warranties with your profile, go to the Hewlett Packard Enterprise Support Center **More Information on Access to Support Materials** page:

  **www.hpe.com/support/AccessToSupportMaterials**

> Access to some updates might require product entitlement when accessed through the Hewlett Packard Enterprise Support Center. You must have an HPE Passport set up with relevant entitlements.

# Customer self repair

Hewlett Packard Enterprise customer self repair (CSR) programs allow you to repair your product. If a CSR part needs to be replaced, it will be shipped directly to you so that you can install it at your convenience. Some parts do not qualify for CSR. Your Hewlett Packard Enterprise authorized service provider will determine whether a repair can be accomplished by CSR.

For more information about CSR, contact your local service provider or go to the CSR website:

**http://www.hpe.com/support/selfrepair**

# Remote support

Remote support is available with supported devices as part of your warranty or contractual support agreement. It provides intelligent event diagnosis, and automatic, secure submission of hardware event notifications to Hewlett Packard Enterprise, which will initiate a fast and accurate resolution based on your product's service level. Hewlett Packard Enterprise strongly recommends that you register your device for remote support.

If your product includes additional remote support details, use search to locate that information.

**Remote support and Proactive Care information**

**HPE Get Connected**

**www.hpe.com/services/getconnected**

**HPE Proactive Care services**

**www.hpe.com/services/proactivecare**

**HPE Proactive Care service: Supported products list**

**www.hpe.com/services/proactivecaresupportedproducts**

**HPE Proactive Care advanced service: Supported products list**

**www.hpe.com/services/proactivecareadvancedsupportedproducts**

**Proactive Care customer information**

**Proactive Care central**

**www.hpe.com/services/proactivecarecentral**

**Proactive Care service activation**

**www.hpe.com/services/proactivecarecentralgetstarted**

# Warranty information

To view the warranty for your product or to view the *Safety and Compliance Information for Server, Storage, Power, Networking, and Rack Products* reference document, go to the Enterprise Safety and Compliance website:

**www.hpe.com/support/Safety-Compliance-EnterpriseProducts**

**Additional warranty information**

**HPE ProLiant and x86 Servers and Options**

**www.hpe.com/support/ProLiantServers-Warranties**

**HPE Enterprise Servers**

**www.hpe.com/support/EnterpriseServers-Warranties**

**HPE Storage Products**

**www.hpe.com/support/Storage-Warranties**

**HPE Networking Products**

**www.hpe.com/support/Networking-Warranties**

# Regulatory information

To view the regulatory information for your product, view the *Safety and Compliance Information for Server, Storage, Power, Networking, and Rack Products*, available at the Hewlett Packard Enterprise Support Center:

**www.hpe.com/support/Safety-Compliance-EnterpriseProducts**

### Additional regulatory information

Hewlett Packard Enterprise is committed to providing our customers with information about the chemical substances in our products as needed to comply with legal requirements such as REACH (Regulation EC No 1907/2006 of the European Parliament and the Council). A chemical information report for this product can be found at:

**www.hpe.com/info/reach**

For Hewlett Packard Enterprise product environmental and safety information and compliance data, including RoHS and REACH, see:

**www.hpe.com/info/ecodata**

For Hewlett Packard Enterprise environmental information, including company programs, product recycling, and energy efficiency, see:

**www.hpe.com/info/environment**

## Documentation feedback

Hewlett Packard Enterprise is committed to providing documentation that meets your needs. To help us improve the documentation, send any errors, suggestions, or comments to Documentation Feedback (**docsfeedback@hpe.com**). When submitting your feedback, include the document title, part number, edition, and publication date located on the front cover of the document. For online help content, include the product name, product version, help edition, and publication date located on the legal notices page.