AT&T

PIM Configuration Guide, 17.2.0

November 2017

Supporting AT&T Vyatta Network Operating System

Contents

Copyright Statement

The training materials and other content provided herein for assistance in training on the Vyatta vRouter may have references to Brocade as the Vyatta vRouter was formerly a Brocade product prior to AT&T's acquisition of Vyatta. Brocade remains a separate company and is not affiliated to AT&T.

About This Guide

This guide describes how to configure PIM on AT&T products that run on the AT&T Vyatta Network Operating System (referred to as a virtual router, vRouter, or router in the guide).

PIM Overview

Multicast forwarding and PIM

In a multicast network, hosts are responsible for informing routers that they want to receive a particular multicast stream. In IPv4 networks, they do this by using IGMP. In IPv6 networks, they do this by using MLD, which is part of Internet Control Message Protocol for Internet Protocol version 6 (ICMPv6).

> **Note:** For an overview of multicast routing in general, see AT&T Vyatta Network Operating System Multicast Routing Configuration Guide. For an overview of the Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD), see AT&T Vyatta Network Operating System IGMP and MLD Configuration Guide.

The multicast-capable routers are responsible for replicating messages and forwarding them to the appropriate recipients. To do this replication and forwarding, multicast routers create distribution trees that control the path that IP multicast traffic takes through the network to deliver traffic to all receivers. The AT&T Vyatta vRouter supports the use of Protocol Independent Multicast (PIM) to manage the communication between multicast routers.

Prior to PIM routing protocols, such as DVMRP, included a unicast routing protocol and operated without any dependency on other unicast routing protocols. The independent nature of PIM means that it performs multicast routing independent of a dependency on any specific unicast routing protocol. PIM makes use of whatever unicast routing mechanism (static routing, RIP, OSPF, etc) that is used.

Distribution trees

The main purpose of PIM is to dynamically construct and maintain efficient multicast routing trees. These trees maintain the state for multicast sources (S, the IP address of the source) and groups (G, the multicast group represented as a multicast IP address) in (S, G) entries; hence, the source tree is often referred to as the (S, G) state. These routing trees control the distribution of multicast traffic through the network and so are called distribution trees.

Multicast distribution trees define how multicast packets are forwarded from a source to all receivers. PIM constructs these multicast distribution trees by referencing unicast routing to the determine Reverse Path Forwarding (RPF) upstream routers along the path from a receiver to a sender. This creates optimal paths, avoids routing loops and can change as network topology changes.

Multicast distribution trees map a multicast source to multicast groups. PIM uses the existing unicast routing able to find the best path from receivers back to the source. They then forward traffic down appropriate paths by using the distribution tree to avoid routing loops.

PIM uses two types of multicast distribution trees: source distribution trees and shared distribution trees.

Source distribution trees

The simplest multicast distribution tree is a source tree. When a router creates a source distribution tree, it puts the multicast source as the root of the tree and creates a spanning tree through the network to each device in the group of receivers.

The source distribution tree represents the shortest path from the source to each multicast group member. For this reason, it is also a shortest path tree (SPT). Because it represents an optimal path, a source distribution tree minimizes the latency in the network. At the same time, the multicast router must track all sources and maintain state information for each source. As a result, source trees can become a burden to the multicast router, especially as the number of sources grows.

Shared distribution trees and the rendezvous point

The Rendezvous Point (RP) is placed at a selected location that sees the traffic between the multicast source and the receiver groups. The RP maintains state for the respective groups. Multicast receivers initiate IGMP Join messages to their upstream router, which is named the designated router. The designated router forwards the packet to the RP.

In this model, the RP must be configured on all designated routers and must be reachable from those routers through a PIM-enabled interface.

Shared distribution trees consume much less memory than source distribution trees. At the same time, because a complete set of shortest paths is not maintained, the paths represented in a shared distribution tree may not be optimal and network latency may be greater than with a source tree.

Reverse path forwarding

PIM makes use of unicast routing to determine neighbor routers and create multicast distribution trees.

The PIM router compares the ingress interface for each multicast packet with the unicast route to source. If the ingress interface is the correct RPF interface the packet is forwarded on the best matching multicast distribution tree. If the RPF check fails the packet is dropped. It is useful to note that PIM assumes unicast routes are symmetrical.

PIM operational modes

PIM can operate in a number of modes:

- Dense mode (PIM-DM)
- Sparse mode (PIM-SM)
- Sparse-dense mode (PIM-SDM)
- Bidirectional PIM (BIDIR-PIM)
- Nonbroadcast multiaccess (NBMA) mode

Each mode is suited to a particular environment. The AT&T Vyatta vRouter supports PIM-DM, PIM-SM, and sparse-dense mode.

PIM also supports Any Source Multicast (ASM) and Source-Specific Multicast (SSM). ASM and SSM are described in AT&T Vyatta Network Operating System Multicast Routing Configuration Guide.

PIM-DM

PIM-DM routers flood multicast traffic to all possible downstream neighbors and then prunes downstream interfaces from the distribution tree as requested by downstream routers. This “flood and prune” approach allows for lighter weight configuration on each router. However, this approach causes some unnecessary packet forwarding.

If a router receives a multicast packet but has no downstream receivers for that group it sends a PIM prune message to the upstream router asking it to stop forwarding traffic for that group. The path remains pruned for about three minutes, after which the router with a pruned interface sends a PIM source refresh message downstream. If the downstream router still has no downstream listeners, it sends a refreshing PIM prune upstream. The downstream router may also, at any time, send a PIM join message to effectively cancel a previous prune.

PIM-DM is recommended for networks where some multicast flooding is an acceptable trade off for reduced configuration complexity. Dense networks, such as LANs, are good candidates for PIM-DM.

The PIM-DM protocol is specified in RFC 3973, *Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)*.

PIM-SM

PIM-SM uses a shared distribution tree built rooted at an RP to determine paths from source to receiver groups. The RP must be administratively configured on the network.

The multicast source registers with the RP and sends the traffic to the RP. Receivers join the group and are added to the tree that is rooted at the RP. The RP distributes the traffic down the tree to the hosts that have joined the multicast group.

By using a shared tree, PIM-SM reduces the amount of information the router needs to maintain. Only information about multicast groups is maintained; state information about sources is not. Unlike PIM-DM, a PIM-SM router with no receivers need not maintain pruning information.

In addition, unlike PIM-DM, multicast data is not flooded to the network. Multicast traffic is not sent to a network segment unless the downstream router specifically requests it. This can significantly reduce the amount of traffic on the network.

Because of these advantages, PIM-SM is currently the multicast protocol of choice in wide-area, inter-domain networks.

PIM-SDM

With PIM SDM configured on an interface, the operational mode for a given multicast group is determined by the presence or otherwise of a PIM Rendezvous Point (RP) for the group, designated as RP(G). Select operational mode as Sparse-Mode (SM) if an RP(G) is present for a given multicast group, otherwise select Dense-Mode (DM).

The presence or otherwise of RP(G) is determined at the start of traffic flow. Change of RP(G) when traffic is flowing may result in indeterminate behavior until traffic ceases to flow. RP(G) can be configured statically making use of route access-lists for given multicast groups or learned dynamically for example, via BSR. RP(G) can be modified via the configuration of PIM dense-groups, forcing designated groups to operate in dense mode and over-riding any RP(G). SDM operates correctly for SM and DM flows when PIM is configured appropriately and consistently at each router in a network (minimum of a single RP). Use existing show commands to elicit PIM operational state - statistics continue to be displayed as SM or DM for a given group.

PIM-SSM

The PIM protocol supports source-specific multicast (SSM) with the assistance of filtering provided by IGMP version 3 and MLD version 2, which are described in AT&T Vyatta Network Operating System IGMP and MLD Configuration Guide. With PIM-SSM, neither an RP nor a shared distribution tree is used. Instead, a source-specific distribution tree is constructed for each (S,G).

Bootstrap routers

PIM version 2 implements a bootstrap router (BSR) mechanism as an alternative to the statically configured RP. In this topology, the RP is not configured. Instead, a number of multicast-enabled routers are configured as BSR and RP candidates.

BSR candidates announce themselves to other routers by using the PIM Multicast version 2 IP address (224.0.0.13). The candidacy message is forwarded through the network, and the BSR is elected from the candidates on the basis of its configured BSR priority. BSR becomes the source of the RP mapping information, calculates the RPs from the RP candidates information. BSR then disseminates the RP mapping information throughout the whole PIM enabled network.

Passive interfaces

A routing-enabled interface that is configured to operate in passive mode does not send routing advertisements out the interface and does not accept them. In the AT&T Vyatta vRouter, both PIM-SIM and PIM-DM support a passive mode.

Embedded RP

The PIM implementation within the AT&T Vyatta vRouter supports the embedded-rendezvous (RP) feature. This mechanism defines an address-allocation policy in which the address of the RP is encoded in a multicast group address. When PIM sparse mode is configured, the embedded rendezvous-point IPv6 feature can be used as the specification for a group-to-RP mapping mechanism.

To support embedded RP, the router configured as the RP must use a configured access list that permits the embedded RP group ranges derived from the embedded RP address. If embedded RP support is available, only the RP must be statically configured as the RP for the embedded RP ranges: No additional configuration is required on other PIMv6 routers. The other routers discover the RP address from the IPv6 group address. For these routers to select a static RP instead of the embedded RP, the specific embedded RP group range must be configured in the access list of the static RP, and embedded RP support must be disabled.

Supported standards

The AT&T Vyatta vRouter implementation of multicast routing complies with the following standards:

- RFC 1112: *Host Extensions for IP Multicasting*
- RFC 2362: *Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification*
- RFC 3973: *Protocol Independent Multicast-Dense Mode (PIM-DM): Protocol Specification (Revised)*
- RFC 4607: *Source-Specific Multicast for IP*

Supported MIBs

The AT&T Vyatta vRouter implementation of PIM supports the following Simple Management Network Protocol (SNMP) management information bases (MIBs):

- IPMROUTE, RFC 2932: *IPv4 Multicast Routing MIB*

For a list of all MIBs supported on the AT&T Vyatta vRouter, see AT&T Vyatta Network Operating System Remote Management Configuration Guide.

PIM configuration

PIM configurations depend on other multicast-related commands. For this reason, the configuration examples are located elsewhere. For PIM configuration examples, see AT&T Vyatta Network Operating System Multicast Routing Configuration Guide.

PIM Commands for IPv4

interfaces <interface> ip pim

Enables PIM on an interface.

Syntax:
`set interfaces` *interface* **`ip pim`**

Syntax:
`delete interfaces` *interface* **`ip pim`**

Syntax:
`show interfaces` *interface* **`ip pim`**

interface

A type of interface. For detailed keywords and arguments for interfaces that support multicast routing, see to Supported Interface Types *(page 77)*.

> **Note:** The *interface* parameter throughout this document consists of <interface-type> <interface-name>

Configuration mode

```
interfaces interface {
    ip {
        pim {
        }
    }
}
```

Use this command to enable PIM on an interface.

> **Note:** To use PIM for multicast routing, multicast routing must be enabled on the router. For information about multicast routing in general, see AT&T Vyatta Network Operating System Multicast Routing Configuration Guide.

Use the `set` form of this command to enable PIM for IPv4 on an interface.

Use the `delete` form of this command to remove all PIM configuration and disable PIM for IPv4 on an interface.

Use the `show` form of this command to display the configuration of PIM for IPv4.

interfaces <interface> ip pim bsr-border

Prevents bootstrap router (BSR) messages from being sent or received through an interface.

Syntax:
`set interfaces` *interface* **`ip pim bsr-border`**

Syntax:
`delete interfaces` *interface* **`ip pim bsr-border`**

Syntax:
`show interfaces` *interface* **`ip pim bsr-border`**

BSR messages can be sent or received through an interface.

interface
The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ip pim *(page 14)*.

Configuration mode

```
interfaces interface {
    ip {
        pim {
            bsr-border
        }
    }
}
```

Use this command to prevent PIM Version 2 (PIMv2) BSR messages from being sent or received through an interface. This is used to configure an interface bordering another PIM domain to avoid the exchange of BSR messages between the two domains. BSR messages should not be exchanged between different domains because routers in one domain may elect rendezvous points (RPs) in the other domain, resulting in a protocol malfunction or loss of isolation between the domains.

> **Note:** This command does not set up multicast boundaries. It only sets up a PIM domain BSR message border.

Use the `set` form of this command to restrict the flow of BSR messages through an interface.

Use the `delete` form of this command to restore the default behavior.

Use the `show` form of this command to display BSR border configuration.

interfaces <interface> ip pim dr-priority

Specifies the designated router (DR) priority.

Syntax:
`set interfaces` *interface* **`ip pim dr-priority`** *priority*

Syntax:
`delete interfaces` *interface* **`ip pim dr-priority`**

Syntax:
`show interfaces` *interface* **`ip pim dr-priority`**

The designated router priority is 1.

interface
The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ip pim *(page 14)*.

priority
The designated router priority. The range is 0 to 4294967294. The default is 1.

Configuration mode

```
interfaces interface {
    ip {
        pim {
            dr-priority priority
        }
    }
}
```

Use this command to specify the designated router priority. The router with the highest priority is elected as the DR by PIM.

Use the `set` form of this command to specify the designated router priority.

Use the `delete` form of this command to restore the designated router priority to its default priority.

Use the `show` form of this command to display the designated router priority.

interfaces <interface> ip pim exclude-genid

Specifies that the generated ID (GenID) option is to be excluded from PIM Hello packets sent on an interface.

Syntax:
`set interfaces` *interface* **`ip pim exclude-genid`**

Syntax:
`delete interfaces` *interface* **`ip pim exclude-genid`**

Syntax:
`show interfaces` *interface* **`ip pim`**

The GenID option is included in Hello packets.

interface

The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ip pim *(page 14)*.

Configuration mode

```
interfaces interface {
    ip {
        pim {
            exclude-genid
        }
    }
}
```

Use this command to exclude the GenID option from PIM Hello packets sent on an interface. This command is used to accommodate operations with older Cisco IOS versions.

Use the `set` form of this command to exclude the GenID option from Hello packets.

Use the `delete` form of this command to restore the default behavior for the GenID option in Hello packets.

Use the `show` form of this command to display the GenID exclusion configuration.

interfaces <interface> ip pim hello-holdtime <holdtime>

Specifies the hello holdtime.

Syntax:
`set interfaces` *interface* **`ip pim hello-holdtime`** *holdtime*

Syntax:
`delete interfaces` *interface* **`ip pim hello-holdtime`**

Syntax:
`show interfaces` *interface* **`ip pim hello-holdtime`**

The hello holdtime is 3.5 times the **`hello-interval`**, in seconds (typically 105 seconds).

interface

The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ip pim *(page 14)*.

holdtime

The hello holdtime, in seconds. The range is 1 to 65535. The default is 3.5 * **hello-interval**.

Configuration mode

```
interfaces interface {
    ip {
        pim {
            hello-holdtime holdtime
        }
    }
}
```

Use this command to configure a hello holdtime, the amount of time the system waits for a PIM Hello message before dropping a neighbor. The holdtime cannot be less than the current **hello-interval**. When the **hello-interval** is updated, the **hello-holdtime** is reviewed. If the **hello-holdtime** either is not configured or is configured but less than the current **hello-interval**, it is set to 3.5 times the **hello-interval**. Otherwise, the current holdtime remains unchanged.

Use the `set` form of this command to specify the hello holdtime.

Use the `delete` form of this command to restore the hello holdtime to its default holdtime.

Use the `show` form of this command to display the hello holdtime configuration.

interfaces <interface> ip pim hello-interval <interval>

Specifies the hello interval.

Syntax:
`set interfaces` *interface* **`ip pim hello-interval`** *interval*

Syntax:
`delete interfaces` *interface* **`ip pim hello-interval`**

Syntax:
`show interfaces` *interface* **`ip pim hello-interval`**

Hello messages are sent every 30 seconds.

interface

The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ip pim *(page 14)*.

interval

The hello interval, in seconds. The hello interval ranges from 1 to 65535. The default interval is 30.

Configuration mode

```
interfaces interface {
    ip {
        pim {
            hello-interval interval
        }
    }
}
```

Use this command to configure a hello interval, the interval at which PIM Hello messages are sent on an interface. When the **hello-interval** is updated, the **hello-holdtime** is reviewed. If the **hello-holdtime** either is not configured or is configured but less than the current hello-interval, it is set to 3.5 times the **hello-interval**. Otherwise, the currently configured **hello-holdtime** remains unchanged.

Use the `set` form of this command to specify the hello interval.

Use the `delete` form of this command to restore the hello interval to its default interval.

Use the `show` form of this command to display the hello interval configuration.

interfaces <interface> ip pim mode <mode>

Specifies the PIM mode on an IPv4 interface.

Syntax:
`set interfaces` *interface* **`ip pim mode`** *mode*

Syntax:
`delete interfaces` *interface* **`ip pim mode`** *mode*

Syntax:
`show interfaces` *interface* **`ip pim mode`** *mode*

interface

The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ip pim *(page 14)*.

mode

The PIM mode that is enabled for an interface. The mode is one of the following:

`dense`: Enable PIM dense mode.

`dense-passive`: Enable passive operation for PIM dense mode.

`sparse`: Enable PIM sparse mode.

`sparse-passive`: Enable passive operation for PIM sparse mode.

`sparse-dense`: Enable PIM sparse-dense mode.

`sparse-dense-passive`: Enable passive operation for PIM sparse-dense mode.

Configuration mode

```
interfaces interface {
    ip {
        pim {
            mode [dense|dense-passive|sparse|sparse-dense|sparse-dense-passive|sparse-passive]
        }
    }
}
```

Use this command to specify the PIM mode on an interface. Use the **`dense`** or **`sparse`** keyword to enable PIM dense mode or PIM sparse mode on an interface. Use the **`dense-passive`** or **`sparse-passive`** keyword to stop PIM transactions on an interface, allowing only Internet Group Management Protocol (IGMP) to be active. Use the **`sparse-dense`** keyword to enable PIM sparse-dense mode on an interface. The mode of a particular group (G) depends on the presence of an RP for that group (denoted RP(G)). When RP(G) is present, the mode is sparse for a group; otherwise, it is dense.

Use the `set` form of this command to specify the PIM mode for an interface.

Use the `delete` form of this command to disable PIM on an interface.

Use the `show` form of this command to display the PIM mode configuration.

interfaces <interface> ip pim neighbor-filter <acl>

Enables filtering of neighbors on an interface.

Syntax:

`set interfaces` *interface* **`ip pim neighbor-filter`** *acl*

Syntax:
`delete interfaces` *interface* **`ip pim neighbor-filter`** *acl*

Syntax:
`show interfaces` *interface* **`ip pim neighbor-filter`**

interface

The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ip pim *(page 14)*.

acl

A standard IP access list number. The number ranges from 1 to 99. An ACL is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating one.

Configuration mode

```
interfaces interface {
    ip {
        pim {
            neighbor-filter acl
        }
    }
}
```

Use this command to enable filtering of neighbors on an interface based on an ACL. PIM terminates its adjacency with existing neighbors filtered by the ACL, and does not establish adjacency with potential neighbors filtered by the ACL.

Use the `set` form of this command to enable filtering of neighbors on an interface based on the specified ACL.

Use the `delete` form of this command to disable filtering of neighbors on an interface based on the specified ACL.

Use the `show` form of this command to display the neighbor filter configuration.

interfaces <interface> ip pim propagation-delay <delay>

Specifies the propagation delay for PIM on an interface.

Syntax:
`set interfaces` *interface* **`ip pim propagation-delay`** *delay*

Syntax:
`delete interfaces` *interface* **`ip pim propagation-delay`**

Syntax:
`show interfaces` *interface* **`ip pim propagation-delay`**

The propagation delay is 1000 milliseconds.

interface

The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ip pim *(page 14)*.

delay

The propagation delay, in milliseconds. The delay ranges from 1000 to 5000.

Configuration mode

```
interfaces interface {
    ip {
        pim {
            propagation-delay delay
        }
    }
}
```

Use this command to specify the expected message propagation delay on the link. It is used by upstream routers to determine how long to wait for a Join override message before pruning an interface.

Use the `set` form of this command to specify the propagation delay for PIM on an interface.

Use the `delete` form of this command to restore the default propagation delay for PIM on an interface.

Use the `show` form of this command to display the propagation delay configuration.

interfaces <interface> ip pim state-refresh origination-interval <interval>

Specifies the PIM-Dense Mode (PIM-DM) State Refresh origination interval on an interface.

Syntax:

`set interfaces` *interface* **`ip pim state-refresh origination-interval`** *interval*

Syntax:

`delete interfaces` *interface* **`ip pim state-refresh origination-interval`**

Syntax:

`show interfaces` *interface* **`ip pim state-refresh origination-interval`**

The PIM-DM State-Refresh origination interval is 60 seconds.

interface

The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ip pim *(page 14)*.

interval

The PIM-DM State Refresh origination interval, in seconds. The interval ranges from 1 to 100. The default interval is 60.

Configuration mode

```
interfaces interface {
    ip {
        pim {
            state-refresh {
                origination-interval interval
            }
        }
    }
}
```

Use this command to specify the PIM-DM State Refresh origination interval on an interface. This interval is the amount of time between PIM-DM State Refresh control messages.

Use the `set` form of this command to specify the PIM-DM State Refresh origination interval on an interface.

Use the `delete` form of this command to restore the PIM-DM State Refresh origination interval to its default interval.

Use the `show` form of this command to display the configuration of a PIM-DM State Refresh interval.

interfaces <interface> ip pim unicast-bsm

Enables the sending and receiving of unicast Bootstrap Messages (BSM) on an interface.

Syntax:
`set interfaces` *interface* **`ip pim unicast-bsm`**

Syntax:
`delete interfaces` *interface* **`ip pim unicast-bsm`**

Syntax:
`show interfaces` *interface* **`ip pim`**

Unicast bootstrap messaging is disabled on an interface.

interface
: The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ip pim *(page 14)*.

Configuration mode

```
interfaces interface {
    ip {
        pim {
            unicast-bsm
        }
    }
}
```

Use this command to enable the sending and receiving of unicast BSM on an interface. This command provides support for older versions of the Bootstrap Router (BSR) specification. This specification specifies the use of unicast BSM to refresh the state of new or restarting neighbors.

Use the `set` form of this command to enable support for unicast BSM on an interface.

Use the `delete` form of this command to disable support for unicast bootstrap messaging on an interface.

Use the `show` form of this command to display the configuration of unicast bootstrap messaging.

monitor protocol multicast pim

Sets debugging options for PIM.

Syntax:

```
monitor protocol multicast pim [ [ background { start | stop } ] | { enable | disable } ip [ events | mfc
| mib | mtrace | nexthop | nsm | packet [ in | out ] | state | timer [ assert [ at ] | bsr [ bst | crp ]
| hello [ ht | nlt | tht ] | joinprune [ et | jt | kat | ot | ppt ] | register [ rst ] ] ] ]
```

PIM debugging is disabled.

`background`
: Performs debugging operations in the background.

`start`
: Starts debugging in the background.

`stop`
: Stops debugging in the background.

`enable`
: Enables the specified debugging option.

`disable`
: Disables the specified debugging option.

`events`
: Enables debugging for PIM events.

`mfc`
: Enables debugging for Multicast Forwarding Cache (MFC) updates.

`mib`
: Enables debugging for Management Information Base (MIB) entries.

`mtrace`
Enables debugging for Multicast Traceroute (MTRACE) messages.

`nexthop`
Enables debugging for Reverse Path Forwarding (RPF) neighbor nexthop cache handling.

`nsm`
Enables debugging for Network Services Module (NSM) messages.

`packet`
Enables debugging for PIM packets.

`in`
Enables debugging for incoming PIM packets.

`out`
Enables debugging for outgoing PIM packets.

`state`
Enables debugging for PIM states.

`timer`
Enables debugging for PIM timers.

`assert`
Enables debugging for PIM assert timers.

`at`
Enables debugging for PIM assert timer.

`bsr`
Enables debugging for PIM BSR timers.

`bst`
Enables debugging for PIM bootstrap timer.

`crp`
Enables debugging for PIM Candidate-RP timer.

`hello`
Enables debugging for various PIM timers.

`ht`
Enables debugging for PIM Hello timer.

`nlt`
Enables debugging for PIM Neighbor Liveliness timer.

`tht`
Enables debugging for PIM Triggered Hello timer.

`joinprune`
Enables debugging for various PIM JoinPrune timers.

`et`
Enables debugging for PIM JoinPrune Expiry timer.

`jt`
Enables debugging for PIM JoinPrune Upstream Join timer.

`kat`
Enables debugging for PIM JoinPrune Keepalive timer.

`ot`
Enables debugging for PIM JoinPrune Upstream Override timer.

`ppt`
Enables debugging for PIM JoinPrune Prune Pending timer.

`register`
Enables debugging for various PIM register timers.

`rst`
Enables debugging for PIM Register Stop timer.

Operational mode

Use this command to enable or disable debugging for PIM and to set PIM debugging options.

The following example shows how to start debugging for IPv4 PIM events.

```
vyatta@vyatta:~$monitor protocol multicast pim enable ip event
```

protocols pim accept-register list <acl>

Allows the Rendezvous Point (RP) to accept Register messages only from multicast sources identified in a given access list.

Syntax:
`set protocols pim accept-register list` *acl*

Syntax:
`delete protocols pim accept-register list` *acl*

Syntax:
`show protocols pim accept-register list`

The RP accepts Register messages from all multicast sources.

acl

The number of an access list. The number ranges as follows:

`100` to **`199`**: Extended IP access list number.

`2000` to **`2699`**: Extended IP access list number in the expanded range.

Configuration mode

```
protocols {
    pim {
        accept-register {
            list acl
        }
    }
}
```

Use this command to configure the RP router to filter multicast sources identified by the specified access list. The RP accepts Register messages sent only by the sources specified in the access list. By default, the RP accepts Register messages from all multicast sources.

Use the `set` form of this command to enable the RP to accept Register messages only from multicast sources identified in a given access list.

Use the `delete` form of this command to restore the default behavior of Register handling.

Use the `show` form of this command to display accept-register list configuration.

protocols pim anycast-rp <rp-address> anycast-rp-peer <rp-peer-address>

Specifies the address of a Rendezvous Point (RP) member in an Anycast-RP set.

Syntax:
`set protocols pim anycast-rp` *rp-address* **`anycast-rp-peer`** *rp-peer-address*

Syntax:
`delete protocols pim anycast-rp` *rp-address* **`anycast-rp-peer`** *rp-peer-address*

Syntax:
`show protocols pim anycast-rp` *rp-address* **`anycast-rp-peer`**

rp-address

The unicast IPv4 address of an Anycast-RP set.

rp-peer-address

The IPv4 address of an RP member of an Anycast-RP set.

Configuration mode

```
protocols {
    pim {
        anycast-rp rp-address {
            anycast-rp-peer rp-peer-address
        }
    }
}
```

Use this command to specify the IP address of an RP member of an Anycast-RP set. Register messages are copied to and sent from this address.

Use the `set` form of this command to specify the IP address of an RP member of an Anycast-RP set.

Use the `delete` form of this command to remove the IP address of an RP member of an Anycast-RP set.

Use the `show` form of this command to display the RP member configuration.

protocols pim bsr-candidate <interface>

Sets the candidate bootstrap router (BSR) status by using the IP address of the specified interface.

Syntax:

`set protocols pim bsr-candidate` *interface* [**`hash-mask`** *mask*] [**`priority`** *priority*]

Syntax:

`delete protocols pim bsr-candidate` *interface* [**`hash-mask`**] [**`priority`**]

Syntax:

`show protocols pim bsr-candidate` *interface* [**`hash-mask`**] [**`priority`**]

mask

The hash mask length for RP selection. The length ranges from 0 to 32. The default length is 10.

priority

The priority for the BSR candidate. The priority ranges from 0 to 255. The default priority is 64.

Configuration mode

```
protocols {
    pim {
        bsr-candidate interface {
            hash-mask mask
            priority priority
        }
    }
}
```

Use this command to set the candidate BSR status by using the IP address of the specified interface.

Use the `set` form of this command to set the candidate BSR status by using the IP address of the specified interface.

Use the `delete` form of this command to remove the candidate BSR status from the interface.

Use the `show` form of this command to display the candidate BSR configuration.

protocols pim dense-group <group-ip-address>

Uses the dense-group keyword to force the interface PIM mode for a group to be dense. The groups can only be SM, DM, SSM, or Bidir; the sparse-dense interface mode allows the interface to flood traffic for DM groups

Syntax:
`set protocols pim` **`dense-group`** *group-ip-address*

Syntax:
`delete protocols pim` **`dense-group`** *group-ip-address*

Syntax:
`show protocols pim` **`dense-group`** *group-ip-address*

group-ip-address

The IPv4 address for the data plane. The format of the IPv4 address is x.x.x.x/x.

Configuration mode

```
protocols {

        pim {
                dense-group 225.0.0.3
                rp-address 5.5.5.5 {
                        list 1
                }
        }
}
```

PIM dense-groups are added to force designated groups to be dense when otherwise the presence of an RP(G) would cause all groups to be sparse.

Of particular significance is that SDM requires at least one RP to be present in the network to function, even in the absence of an RP(G) for any multicast group. Without an RP, DM traffic does not flow through SDM configured interfaces (unless connected interfaces on adjoining routers are configured for DM).

Use the `set` version of the command to force the interface PIM mode for a group to be dense.

Use the `delete` version of the command to undo the PIM configuration.

Use the `show` version of the command to display the PIM configuration.

The following example shows the use of dense-group and access-list in the context of SDM.

```
policy {
        route {
                access-list 1 {
                        rule 1 {
                                action permit
                                source {
                                        inverse-mask 0.0.0.3
                                        network 225.0.0.0
                                }
                        }
                }
         }
}

protocols {
        pim {
                dense-group 225.0.0.3
                rp-address 5.5.5.5 {
                        list 1
                }
        }
}
```

Note: The group must be defined as "source" rather than "destination".

protocols pim ignore-rp-set-priority

Specifies that the RP-SET priority is to be ignored in Rendezvous Point (RP) selection.

Syntax:
`set protocols pim ignore-rp-set-priority`

Syntax:
`delete protocols pim ignore-rp-set-priority`

Syntax:
`show protocols pim`

The RP-SET priority is used in RP selection.

Configuration mode

```
protocols {
    pim {
        ignore-rp-set-priority
    }
}
```

Use this command to specify that the RP-SET priority is to be ignored and that only the hash value is to be used in RP selection. This command provides interoperability with older Cisco IOS versions.

Use the `set` form of this command to specify that the RP-SET priority is to be ignored and that only the hash value is to be used in RP selection.

Use the `delete` form of this command to restore the default RP selection mechanism by using the RP-SET priority.

Use the `show` form of this command to display the PIM configuration.

protocols pim join-prune-timer <timer>

Sets the PIM join/prune timer.

Syntax:
`set protocols pim join-prune-timer` *timer*

Syntax:
`delete protocols pim join-prune-timer`

Syntax:
`show protocols pim join-prune-timer`

The join/prune timer is 210 seconds.

timer

The join/prune timer, in seconds. The timer ranges from 1 to 65535. The default timer is 210.

Configuration mode

```
protocols {
    pim {
        join-prune-timer timer
    }
}
```

Use this command to specify the PIM join/prune timer.

Use the `set` form of this command to specify the PIM join/prune timer.

Use the `delete` form of this command to remove the PIM join/prune timer.

Use the `show` form of this command to display the PIM join/prune timer configuration.

protocols pim legacy-register-checksum [group-list <acl>]

Specifies that the Register checksum should be calculated over the whole packet.

Syntax:
`set protocols pim legacy-register-checksum` [**`group-list`** *acl*]

Syntax:
`delete protocols pim legacy-register-checksum` [**`group-list`**]

Syntax:
`show protocols pim legacy-register-checksum` [**`group-list`**]

The Register checksum is calculated over only the packet header.

acl

An access list number that specifies the multicast groups on which to calculate the Register checksum over the whole packet. The number ranges as follows:

`1` to **`99`**: IP access list number.

`1300` to **`1999`**: IP access list number in the expanded range.

An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

Configuration mode

```
protocols {
    pim {
        legacy-register-checksum {
            group-list acl
        }
    }
}
```

Use this command to specify that the Register checksum should be calculated over the whole packet rather than only over the packet header. This command is used to accommodate operations with older Cisco IOS versions.

Use the `set` form of this command to specify that the Register checksum should be calculated over the whole packet rather than only over the packet header.

Use the `delete` form of this command to restore the default operation.

Use the `show` form of this command to display legacy register checksum configuration.

protocols pim log

Enables PIM logs.

Syntax:

set protocols pim log { **all** | **events** | **mfc** | **mib** | **mtrace** | **nexthop** | **nsm** | **packet** | **state** | **timer** }

Syntax:
delete protocols pim log { **all** | **events** | **mfc** | **mib** | **mtrace** | **nexthop** | **nsm** | **packet** | **state** | **timer** }

Syntax:
show protocols pim log { **all** | **events** | **mfc** | **mib** | **mtrace** | **nexthop** | **nsm** | **packet** | **state** | **timer** }

None

all

Enables all PIM logs.

events

Enables PIM debugging of general configuration and virtual routing.

mfc

Enables PIM debugging for MFC updates.

mib

Enables PIM debugging for MIB entries.

mtrace

Enables PIM debugging for MTRACE messages.

nexthop

Enables PIM debugging for nexthop cache handling for RPF neighbors .

nsm

Enables PIM debugging for PIM NSM.

packet

nables PIM debugging for PIM packets.

state

nables PIM debugging for PIM stes.

timer

nables PIM debugging for PIM timers.

None

```
protocols {
    pim {
        log {
            all
            events
            mfc
            mib
            msdp
            mtrace
            nexthop
            nsm
            packet
            state
            timer
        }
    }
}
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM) logs.

Use the `delete` form of this command to remove PIM logs.

Use the `show` form of this command to view PIM logs.

protocols pim log packet

Enables PIM debugging of PIM packets.

Syntax:

```
set protocols pim log packet { all | rcv | send }
```

Syntax:

```
delete protocols pim log packet { all | rcv | send }
```

Syntax:

```
show protocols pim log packet { all | rcv | send }
```

None

all
: Enables all PIM packet logs.

rcv
: Enables debugging for all incoming PIM packets.

send
: Enables debugging for all outgoing PIM packets.

Configuration mode

```
protocols {
     pim {
        log {
           packet {
              all
              rcv
              send
           }
        }
     }
}
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM) packet logs.

Use the `delete` form of this command to remove PIM packet logs.

Use the `show` form of this command to view PIM packet logs.

protocols pim log timer

Enables PIM debugging for PIM timers.

Syntax:

```
set protocols pim log timer{ all | assert | bsr | hello | joinprune | register }
```

Syntax:

```
delete protocols pim log timer{ all | assert | bsr | hello | joinprune | register }
```

Syntax:

```
show protocols pim log timer{ all | assert | bsr | hello | joinprune | register }
```

None

all
: Enables PIM debugging for all PIM timers.

assert
: Enables PIM debugging for all PIM assert timers.

bsr
: Enables PIM debugging for all PIM BSR timers.

hello
: Enables PIM debugging for various PIM timers.

joinprune

Enables PIM debugging for PIM join-prune timers.

register

Enables PIM debugging for various PIM register timers.

Configuration mode

```
protocols{
    pim {
       log {
          timer {
             all
             assert
             bsr
             hello
             joinprune
             register
          }
       }
    }
}
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM) timer logs.

Use the `delete` form of this command to remove PIM timer logs.

Use the `show` form of this command to view PIM timer logs.

protocols pim log timer assert

Enables PIM debugging for PIM assert timers.

Syntax:

```
set protocols pim log timer assert { all | at }
```

Syntax:

```
delete protocols pim log timer assert { all | at }
```

Syntax:

```
show protocols pim log timer assert { all | at }
```

None

all

Enables debugging of all PIM assert timers.

at

Enables debugging of PIM assert timers.

Configuration mode

```
protocols {
        pim {
          log {
            timer {
              assert {
                  all
                  at
                    }
                  }
                }
              }
            }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM) assert timer logs.

Use the `delete` form of this command to remove PIM assert timer logs.

Use the `show` form of this command to view PIM assert timer logs.

protocols pim log timer bsr

Enables PIM debugging for PIM BSR timer.

Syntax:

```
set protocols pim log timer bsr { all | bst | crp }
```

Syntax:

```
delete protocols pim log timer bsr { all | bst | crp }
```

Syntax:

```
show protocols pim log timer bsr { all | bst | crp }
```

None

all

Enables debugging of all PIM BSR timers.

bst

Enables debugging of only bootstrap timers.

crp

Enables debugging of only candidate-RP timers.

Configuration mode

```
protocols {
    pim {
     log {
      timer {
        bsr {
         all
         bst
         crp
         }
        }
       }
      }
     }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM) BSR timer logs.

Use the `delete` form of this command to remove PIM BSR timer logs.

Use the `show` form of this command to view PIM BSR timer logs.

protocols pim log timer hello

Enables PIM debugging for PIM hello timers.

Syntax:

```
set protocols pim log timer hello { all | ht | nlt| tht }
```

Syntax:

```
delete protocols pim log timer hello { all | ht | nlt| tht }
```

Syntax:

```
show protocols pim log timer hello { all | ht | nlt| tht }
```

None

all

Enables debugging of all PIM hello timers.

ht

Enables debugging of only PIM hello timers.

nlt

Enables debugging of only PIM neighbor liveliness timers.

tht

Enables debugging of only PIM triggerred hello timers.

Configuration mode

```
protocols {
        pim {
          log {
           timer {
             hello {
                all
                ht
                nlt
                tht
                   }
                 }
               }
             }
           }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM) timer hello logs.

Use the `delete` form of this command to remove PIM timer hello logs.

Use the `show` form of this command to view PIM hello logs.

protocols pim log timer joinprune

Enables PIM debugging for PIM join-prune timers.

Syntax:

set protocols pim log timer joinprune { **all** | **et** | **jt** | **kat** | **ot** | **ppt** }

Syntax:

show protocols pim log timer joinprune { **all** | **et** | **jt** | **kat** | **ot** | **ppt** }

Syntax:

delete protocols pim log timer joinprune { **all** | **et** | **jt** | **kat** | **ot** | **ppt** }

None

all

Enables debugging of all PIM join-prune timers.

et

Enables debugging of PIM join-prune expiry timers.

jt

Enables debugging of PIM join-prune upstream join timers.

kat

Enables debugging of PIM join-prune keep-alive timers.

ot

Enables debugging of PIM join-prune over-ride timers.

ppt

Enables debugging of PIM joinprune prune-pending timers.

Configuration mode

```
protocols {
        pim {
          log {
           timer {
            joinprune {
                  all
                  et
                  jt
                  kat
                  ot
                  ppt
                       }
                    }
                  }
                }
            }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM) join-prune timer logs.

Use the `delete` form of this command to remove PIM join-prune timer logs.

Use the `show` form of this command to view PIM join-prune timer logs.

protocols pim log timer register

Enables PIM debugging for PIM register timers.

Syntax:
`set protocols pim log timer register { all | rst }`

Syntax:
`delete protocols pim log timer register { all | rst }`

Syntax:
`show protocols pim log timer register { all | rst }`

None

`all`

Enables debugging of all PIM register timers.

`rst`

Enables debugging of only PIM register-stop timers.

Configuration mode

```
protocols {
       pim {
        log {
          timer {
            register {
              all
              rst
                  }
                }
              }
            }
           }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM) register timer logs.

Use the `delete` form of this command to remove PIM register timer logs.

Use the `show` form of this command to view PIM register timer logs.

protocols pim register-kat <timer>

Specifies the Keepalive Timer (KAT) for (S,G) states at the Rendezvous Point (RP).

Syntax:
`set protocols pim register-kat` *timer*

Syntax:
`delete protocols pim register-kat`

Syntax:
`show protocols pim register-kat`

The KAT for (S,G) states is 210 seconds.

timer

The KAT, in seconds. The timer ranges from 1 to 65535. The default timer is 210.

Configuration mode

```
protocols {
    pim {
        register-kat  timer
    }
}
```

Use this command to specify the KAT for (S,G) states at the RP to monitor Register messages.

Use the `set` form of this command to specify the KAT for (S,G) states at the RP to monitor Register messages.

Use the `delete` form of this command to restore the KAT to 210 seconds.

Use the `show` form of this command to display register KAT configuration.

protocols pim register-rate-limit <rate>

Specifies the rate at which Register messages are sent by this designated router (DR).

Syntax:
`set protocols pim register-rate-limit` *rate*

Syntax:
`delete protocols pim register-rate-limit`

Syntax:
`show protocols pim register-rate-limit`

There is no limit to the rate at which Register messages are sent by the DR.

rate

The rate at which Register messages are sent by the DR, in packets per second. The range is 0 to 65535. The default is 0, meaning “no limit.”

Configuration mode

```
protocols {
    pim {
        register-rate-limit rate
    }
}
```

Use this command to specify the rate of Register messages sent by this DR. This rate is for each (S,G) state. The rate is not system wide.

Use the `set` form of this command to specify the rate of Register messages sent by this DR.

Use the `delete` form of this command to restore no limit to the rate at which Register messages are sent by this DR.

Use the `show` form of this command to display Register rate limit configuration.

protocols pim register-rp-reachability

Enables Rendezvous Point (RP) reachability checking for PIM Registers at the designated router (DR).

Syntax:

```
set protocols pim register-rp-reachability
```

Syntax:

```
delete protocols pim register-rp-reachability
```

Syntax:

```
show protocols pim
```

RP reachability is not checked.

Configuration mode

```

protocols {
    pim {
         register-rp-reachability
    }
}
```

Use this command to enable RP reachability checking for PIM Registers at the DR.

Use the `set` form of this command to enable RP reachability checking for PIM Registers at the DR.

Use the `delete` form of this command to restore no checking for reachability.

Use the `show` form of this command to display the configuration of Register RP reachability.

protocols pim register-source

Specifies the source of Register messages sent by this designated router (DR).

Syntax:

```
set protocols pim register-source { address source | interface interface }
```

Syntax:

```
delete protocols pim register-source { address | interface }
```

Syntax:

```
show protocols pim register-source { address | interface }
```

The IPv4 address of the Reverse Path Forwarding (RPF) interface that faces the source host.

source

An IPv4 address to use as the source of Register messages.

interface

An interface to use as the source of Register messages. Note that it is not necessary for PIM to be enabled on this interface.

Configuration mode

```
protocols {
    pim {
        register-source {
            [address source | interface interface]
        }
    }
}
```

Use this command to specify the source of Register messages sent by this DR. The specified address must be reachable so that the Rendezvous Point (RP) router can send Register-Stop messages in response. The Register source address is usually the address of the loopback interface, though it can be another physical address. The specified address must be advertised by unicast routing protocols on the DR.

Use the `set` form of this command to specify the source of Register messages sent by this DR.

Use the `delete` form of this command to restore the default source of Register messages sent by the DR.

Use the `show` form of this command to display Register source configuration.

protocols pim register-suppression-timer <timer>

Specifies the register-suspension time.

Syntax:

```
set protocols pim register-suppression-timer timer
```

Syntax:

```
delete protocols pim register-suppression-timer
```

Syntax:

```
show protocols pim register-suppression-timer
```

The register-suppression time is 60 seconds.

timer

The register-suppression time, in seconds. The range is 1 to 65535. The default is 60.

Configuration mode

```
protocols {
    pim {
        register-suppression-timer timer
    }
}
```

Use this command to specify the register-suppression time. On a designated router (DR), this configuration modifies the register-suppression time. On a Rendezvous Point (RP) router, this configuration modifies the RPkeepalive-period if protocols pim register-kat <timer> *(page 34)* is not used.

Use the `set` form of this command to specify the register-suppression time.

Use the `delete` form of this command to restore the register-suppression time to 60 seconds.

Use the `show` form of this command to display the configuration of the register-suppression time.

protocols pim rp-address <rp-addr>

Specifies a static rendezvous point (RP) address for multicast groups.

Syntax:

`set protocols pim rp-address` *rp-addr* [**`list`** *acl* | **`override`**]

Syntax:
`delete protocols pim rp-address` *rp-addr* [**`list`** | **`override`**]

Syntax:
`show protocols pim rp-address` *rp-addr* [**`list`** | **`override`**]

rp-addr

The unicast IPv4 RP address of the RP set.

acl

An access list number used to specify the multicast groups for which the static RP address is valid. Supported ranges of values are:

`1` to **`99`**: IP access list number.

`1300` to **`1999`**: IP access list number in the expanded range.

An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

`override`

Specifies that static RP addresses take precedence over dynamically learned RP addresses.

Configuration mode

```
protocols {
    pim {
        rp-address rp-addr {
            list acl
            override
        }
    }
}
```

The PIM implementation supports multiple statically configured rendezvous points (RPs). It also supports the use of both statically configured RPs and those selected through the bootstrap router (BSR) mechanism simultaneously. Note the following:

- If multiple static RP addresses are available for a group range, then the one with the highest IP address is chosen.
- RP addresses configured for a multicast group through the BSR mechanism take precedence over those configured statically unless the **`override`** keyword is used. In those cases, a statically configured RP address takes precedence.
- Configuring multiple static RPs with the same RP address is not allowed.
- One static RP address can be configured for multiple group ranges by using access lists. The static RP address can either be configured for the whole multicast group range (that is, 224.0.0.0/4) or for specific group ranges if an access list is specified. When an access list is specified, the static RP address is configured for all the group ranges represented by Permit filters in the access list.
- Only Permit filters in access lists are considered as valid group ranges. The default Permit filter 0.0.0.0/0 is converted to the default multicast filter 224.0.0.0/4.
- After configuration, the RP address is inserted into a static RP group tree based on the configured group ranges. For each group range, multiple static RPs are maintained in a list. This list is sorted in descending order of IP addresses. When selecting static RPs for a group range, the first element of the list (the statically configured RP with the highest IP address) is selected.
- When an RP address is deleted, the static RP is removed from all the existing group ranges and RPs are recomputed for existing Tree Information Base (TIB) states if required.

Use the `set` form of this command to specify a static RP address for multicast groups.

Use the `delete` form of this command to remove the configuration of static RP addresses.

Use the `show` form of this command to display the configuration of static RP addresses.

protocols pim rp-candidate <interface>

Specifies that the router is a candidate Rendezvous Point (RP.)

Syntax:
`set protocols pim rp-candidate` *interface* [**`group-list`** *acl* | **`interval`** *interval* | **`priority`** *priority*]

Syntax:
`delete protocols pim rp-candidate` *interface* [**`group-list`** | **`interval`** | **`priority`**]

Syntax:
`show protocols pim rp-candidate` *interface* [**`group-list`** | **`interval`** | **`priority`**]

interface

The interface to set with candidate RP status.

acl

An access list number used to specify the group of ranges for this RP candidate. Supported ranges of values are:

`1` to **`99`**: IP access list number.

An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

interval

The candidate RP advertisement interval, in seconds. The range is 1 to 16383. The default is 60.

priority

The candidate RP priority. The range is 0 to 255. The default is 192.

Configuration mode

```
protocols {
    pim {
        rp-candidate interface {
            group-list acl
            interval interval
            priority priority
        }
    }
}
```

Use this command to specify that the router is a candidate RP by using the IP address of the specified interface.

Use the `set` form of this command to specify that this router is a candidate RP.

Use the `delete` form of this command to remove the router as a candidate RP.

Use the `show` form of this command to display the candidate RP configuration.

protocols pim spt-threshold

Enables the last-hop PIM router to switch to shortest-path tree (SPT).

Syntax:
`set protocols pim spt-threshold` [**`infinity`**] [**`group-list`** *acl*]

Syntax:
`delete protocols pim spt-threshold` [**`infinity`**] [**`group-list`**]

Syntax:
`show protocols pim spt-threshold` [**`infinity`**] [**`group-list`**]

`infinity`

Sets the SPT threshold to infinity, which disables the ability of the last-hop PIM router to switch to SPT.

acl

An access list number used to specify the multicast groups for the last-hop PIM router to switch to SPT for. Supported ranges of values are:

`1` to **`99`**: IP access list number.

`1300` to **`1999`**: IP access list number in the expanded range.

An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

Configuration mode

```
protocols {
    pim {
        spt-threshold
            infinity
                group-list acl
    }
}
```

Use this command to enable the last-hop PIM router to switch to SPT. To set the SPT threshold to infinity and disable the ability of the last-hop PIM router to switch to SPT, use the **`infinity`** option with this command.

Use the `set` form of this command to enable the last-hop PIM router to switch to SPT.

Use the `delete` form of this command to remove SPT threshold configuration. To remove the **`infinity`** option with this command, use the `delete` form of this command with this option, which enables the ability of the last-hop PIM router to switch to SPT.

Use the `show` form of this command to display the SPT threshold configuration.

protocols pim ssm default

Enables Source Specific Multicast (SSM) and uses a default range of IP multicast addresses.

Syntax:

```
set protocols pim ssm default
```

Syntax:

```
delete protocols pim ssm default
```

Syntax:

```
show protocols pim ssm default
```

Configuration mode

```
protocols {
    pim {
        ssm {
            default
        }
    }
}
```

Use this command to configure SSM and use the default range of IP multicast addresses (that is, 232.0.0.0/8). Use protocols pim ssm range <acl> *(page 40)* to define the SSM range to be other than the default. When an SSM range of multicast addresses is defined, the no (*,G) or (S,G,rpt) state is initiated for groups in the SSM range. Messages corresponding to these states are neither accepted nor originated in the SSM range.

Use the `set` form of this command to enable SSM and use a default range of IP multicast addresses.

Use the `delete` form of this command to disable SSM.

Use the `show` form of this command to display the SSM default configuration.

protocols pim ssm range <acl>

Enables Source Specific Multicast (SSM) and defines a range of IP multicast addresses based on an access list.

Syntax:

```
set protocols pim ssm range acl
```

Syntax:

```
delete protocols pim ssm range
```

Syntax:

```
show protocols pim ssm range
```

acl

An access list number used to specify the group of ranges for SSM. Supported ranges of values are:

`1` to **`99`**: IP access list number.

An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

Configuration mode

```
protocols {
    pim {
        ssm {
            range acl
        }
    }
}
```

Use this command to enable SSM and define a range of IP multicast addresses based on an access list.

Note: The SSM destination range must be specified in the ACL source parameter .

Use the `set` form of this command to enable SSM and define a range of IP multicast addresses based on an access list.

Use the `delete` form of this command to disable SSM.

Use the `show` form of this command to display the SSM range configuration.

reset ip mroute

Deletes multicast route table entries and multicast routes at the PIM protocol level.

Syntax:

```
reset ip mroute [ group group [ source source [ pim { dense-mode | sparse-mode } ] ] | pim { dense-mode
| sparse-mode } ]
```

group

A multicast group to delete. The format is an IPv4 multicast address.

source

A multicast source to delete. The format is an IPv4 host address.

`dense-mode`

Deletes the multicast route table for PIM dense-mode.

`sparse-mode`

Deletes the multicast route table for PIM sparse-mode.

Operational mode

Use this command to delete multicast route table entries and multicast routes at the PIM protocol level. Used with no options, the command deletes all multicast route table entries and multicast routes. Used with options, the command deletes a subset of the multicast route table entries and multicast routes.

The following example shows how to delete all IPv4 multicast route table entries and multicast routes at the PIM protocol level.

```
vyatta@vyatta:~$reset ip mroute
```

reset ip pim sparse-mode bsr rp-set

Deletes all rendezvous point (RP) sets learned from the PIM Bootstrap Router (BSR).

Syntax:

```
reset ip pim sparse-mode bsr rp-set
```

Operational mode

Use this command to delete all RP sets learned through the PIM BSR.

The following example shows how to delete all RP sets learned through the PIM BSR.

```
vyatta@vyatta:~$reset ip pim sparse-mode bsr rp-set
```

show ip pim bsr-router

Displays the bootstrap router (BSR) PIM Version 2 (PIMv2) address.

Syntax:

```
show ip pim bsr-router
```

Operational mode

Displays the BSR PIMv2 address.

The following example shows how to display the BSR PIMv2 address.

```
vyatta@vyatta:~$show ip pim bsr-router
PIMv2 Bootstrap information
  BSR address: 10.10.11.35 (?)
  Uptime:      00:00:38, BSR Priority: 0, Hash mask length: 10
  Expires:     00:01:32
  Role: Non-candidate BSR
  State: Accept Preferred
```

show ip pim interface

Displays PIM interface information.

Syntax:

`show ip pim interface` [**`detail`**]

`detail`

Displays detailed information about the PIM interface.

Operational mode

Use this command to display PIM interface information.

Example: Examples:

In the following example, the Ver/Mode column displays "SD" when configured in Sparse-Dense Mode (SDM).

```
vyatta@R2:~$ show ip pim interface
Address          Interface           VIFindex Ver/   Nbr    DR         DR
                                              Mode   Count  Prior
10.0.0.3         dp0s7              1        v2/SD   1      1          10.0.0.3
10.0.2.3         dp0s8              2        v2/SD   0      1          10.0.2.3
10.0.23.3        dp0s9              0        v2/SD   1      1          10.0.23.7
2.2.2.2          lo1                 4        v2/S   0      1          2.2.2.2
```

In this example, the Mode field shows "Sparse-Dense" when configured as SDM.

```
vyatta@R2:~$ show ip pim interface detail
dp0s7 (vif 1):
  Address 10.0.0.3, Mode: Sparse-Dense
  DR 10.0.0.3, DR's priority: 1
  Propagation delay is 1000 milli-seconds
  Router-ID:2.2.2.2 Local-ID 8
  Neighbors:
   10.0.0.2
  PIM neighbor count: 1
  PIM neighbor holdtime: 105
  PIM neighbor hello interval: 30
  PIM configured DR priority: 1
  PIM border interface: no
  PIM Neighbor policy: not configured
```

show ip pim local-members

Displays local membership information for a PIM interface.

Syntax:

`show ip pim local-members` [*interface*]

interface

An interface for which to display local membership information.

Operational mode

Use this command to display local membership information for a PIM interface.

show ip pim mroute

Displays the IP PIM multicast routing table.

Syntax:

`show ip pim mroute` [[**`group`** *group* [[**`source`** *source*] **`detail`**] | **`detail`**] | [**`rfc`** [**`group`** *group* [[**`source`** *source*] [**`detail`**]] | **`detail`**] | **`source`** *source* [**`detail`**] | **`detail`** | **`summary`**]

group

The multicast group entries to display. The format is an IPv4 multicast address.

source

The multicast source entries to display. The format is an IPv4 host address.

rfc

Displays information for a PIM multicast routing table (RFC style).

detail

Displays detailed information for a PIM multicast routing table.

summary

Displays summarized information for a PIM routing table.

Operational mode

Use this command to display the IP PIM multicast routing table. Used with no options, the command displays all entries for an IP PIM multicast routing table. Used with the **group** option, **source** options, or both options, the command displays a subset of entries for the IP PIM multicast routing table. Used with the **rfc** option, the command displays the PIM multicast routing table entries in an RFC style.

The following example shows how to display an IP PIM multicast routing table.

```
vyatta@vyatta:~$show ip pim mroute
IP Multicast Routing Table

(*,*,RP) Entries: 0
(*,G) Entries: 2
(S,G) Entries: 1
(S,G,rpt) Entries: 1
FCR Entries: 0

(*, 227.1.1.1)
RP: 10.15.0.1
RPF nbr: 0.0.0.0
RPF idx: None
Upstream State: JOINED
  Local    ................................
  Joined   j...............................
  Asserted ................................
FCR:

(10.17.0.7, 227.1.1.1)
RPF nbr: 0.0.0.0
RPF idx: None
SPT bit: 1
Upstream State: JOINED
  Local    ................................
  Joined   ................................
  Asserted ................................
  Outgoing o...............................

(10.17.0.7, 227.1.1.1, rpt)
RP: 10.15.0.1
RPF nbr: 0.0.0.0
RPF idx: None
Upstream State: NOT PRUNED
  Local    ................................
  Pruned   ................................
  Outgoing o...............................

(*, 239.255.255.250)
RP: 10.15.0.1
RPF nbr: 0.0.0.0
RPF idx: None
Upstream State: JOINED
```

```
  Local    ..i...............................
  Joined   ..................................
  Asserted ..................................
FCR:
```

show ip pim neighbor

Displays PIM neighbor information.

Syntax:

`show ip pim neighbor` [*interface* [*addr* [**`detail`**] | **`detail`**] | **`detail`**]

interface

An interface for which to display PIM neighbor information.

addr

The IPv4 address of a neighbor interface.

`detail`

Displays detailed PIM neighbor information.

Operational mode

Use this command to display PIM neighbor information. Used with no options, the command displays all PIM neighbor information. Used with the **`interface`** option, **`addr`** option, or both options, the command displays information for a subset of PIM neighbors.

show ip pim nexthop

Displays next-hop information used by PIM.

Syntax:

`show ip pim nexthop`

Operational mode

Displays next-hop information used by PIM.

show ip pim rp-hash <group>

Displays the rendezvous point (RP) to select based on the group specified.

Syntax:

`show ip pim rp-hash` *group*

group

A multicast group for which to determine the RP. The format is an IPv4 multicast address.

Operational mode

Displays the RP to select based on the group specified.

The following example shows the command output for a specified multicast group.

```
vyatta@vyatta:~$ show ip pim rp-hash 239.3.3.45
 RP: 45.45.45.1, v2
 Info source: 45.45.45.1, via bootstrap, priority 192, holdtime 150
 Uptime: 00:00:46, expires: 00:01:44
 BSR 45.45.45.1: PIMv2 Hash Value (mask 255.255.255.252)
 RP set received for group(s) 224.0.0.0/4:
 RP: 23.23.23.1, via bootstrap, priority 192, hash value 368873312
 RP set received for group(s) 239.3.3.0/24:
```

```
RP: 45.45.45.1, via bootstrap, priority 192, hash value 1929112928
```

show ip pim rp-mapping

Displays the group-to-Rendezvous Point (RP) mappings and the RP set.

Syntax:

`show ip pim rp-mapping`

Operational mode

Displays the group-to-RP mappings and the RP set.

show monitoring protocols multicast pim

Displays IPv4 multicast debugging information.

Syntax:

`show monitoring protocols multicast pim`

Operational mode

Use this command to display IPv4 multicast debugging information.

The following example shows how to display IPv4 multicast debugging information.

```
vyatta@vyatta:~$show monitoring protocols multicast pim
Debugging status:
PIM event debugging is on
PIM MFC debugging is on
PIM state debugging is on
PIM packet debugging is on
PIM Hello HT timer debugging is on
PIM Hello NLT timer debugging is on
PIM Hello THT timer debugging is on
PIM Join/Prune JT timer debugging is on
PIM Join/Prune ET timer debugging is on
PIM Join/Prune PPT timer debugging is on
PIM Join/Prune KAT timer debugging is on
PIM Join/Prune OT timer debugging is on
PIM Assert AT timer debugging is on
PIM Register RST timer debugging is on
PIM Bootstrap BST timer debugging is on
PIM Bootstrap CRP timer debugging is on
PIM mib debugging is on
PIM nexthop debugging is on
PIM mtrace debugging is on
PIM NSM debugging is on
```

PIM Commands for IPv6

interfaces <interface> ipv6 pim

Enables PIM for IPv6 on an interface.

Syntax:
`set interfaces` *interface* **`ipv6 pim`**

Syntax:
`delete interfaces` *interface* **`ipv6 pim`**

Syntax:
`show interfaces` *interface* **`ipv6 pim`**

interface
The type of interface. For detailed keywords and arguments for interfaces that support multicast routing, see Supported Interface Types *(page 77)*.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
        }
    }
}
```

Use this command to enable PIM for IPv6 on an interface.

> **Note:** To use PIM for multicast routing, multicast routing must be enabled on the router. For information about multicast routing in general, see AT&T Vyatta Network Operating System Multicast Routing Configuration Guide.

Use the `set` form of this command to enable PIM for IPv6 on an interface.

Use the `delete` form of this command to remove all PIM configuration and disable PIM for IPv6 on an interface.

Use the `show` form of this command to display the configuration of PIM for IPv6.

interfaces <interface> ipv6 pim bsr-border

Prevents bootstrap router (BSR) messages from being sent or received through an interface.

Syntax:
`set interfaces` *interface* **`ipv6 pim bsr-border`**

Syntax:
`delete interfaces` *interface* **`ipv6 pim bsr-border`**

Syntax:
`show interfaces` *interface* **`ipv6 pim`**

BSR messages can be sent or received through an interface.

interface

The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ipv6 pim *(page 46)*.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
            bsr-border
        }
    }
}
```

Use this command to prevent PIM Version 2 (PIMv2) BSR messages from being sent or received through an interface. This is used to configure an interface bordering another PIM domain to avoid the exchange of BSR messages between the two domains. BSR messages should not be exchanged between different domains because routers in one domain may elect rendezvous points (RPs) in the other domain, resulting in a protocol malfunction or loss of isolation between the domains.

> **Note:** This command does not set up multicast boundaries. It only sets up a PIM domain BSR message border.

Use the `set` form of this command to restrict the flow of BSR messages through an interface.

Use the `delete` form of this command to restore the default behavior.

Use the `show` form of this command to display BSR border configuration.

interfaces <interface> ipv6 pim dr-priority

Specifies the designated router (DR) priority.

Syntax:
`set interfaces` *interface* **`ipv6 pim dr-priority`** *priority*

Syntax:
`delete interfaces` *interface* **`ipv6 pim dr-priority`**

Syntax:
`show interfaces` *interface* **`ipv6 pim dr-priority`**

The designated router priority is 1.

interface
The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ipv6 pim *(page 46)*.

priority
The designated router priority. The range is 0 to 4294967294. The default is 1.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
            dr-priority priority
        }
    }
}
```

Use this command to specify the designated router priority. The router with the highest priority is elected as the DR by PIM.

Use the `set` form of this command to specify the designated router priority.

Use the **delete** form of this command to restore the designated router priority to its default priority.

Use the show form of this command to display the designated router priority.

interfaces <interface> ipv6 pim exclude-genid

Specifies that the generated ID (GenID) option is to be excluded from PIM Hello packets sent on an interface.

Syntax:
set interfaces *interface* **ipv6 pim exclude-genid**

Syntax:
delete interfaces *interface* **ipv6 pim exclude-genid**

Syntax:
show interfaces *interface* **ipv6 pim**

The GenID option is included in Hello packets.

interface
The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ipv6 pim *(page 46)*.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
            exclude-genid
        }
    }
}
```

Use this command to exclude the GenID option from PIM Hello packets sent on an interface. This command is used to accommodate operations with older Cisco IOS versions.

Use the set form of this command to exclude the GenID option from Hello packets.

Use the **delete** form of this command to restore the default behavior for the GenID option in Hello packets.

Use the show form of this command to display the GenID exclusion configuration.

interfaces <interface> ipv6 pim hello-holdtime <holdtime>

Specifies the hello holdtime.

Syntax:
set interfaces *interface* **ipv6 pim hello-holdtime** *holdtime*

Syntax:
delete interfaces *interface* **ipv6 pim hello-holdtime**

Syntax:
show interfaces *interface* **ipv6 pim hello-holdtime**

The hello holdtime is 3.5 times the **hello-interval**, in seconds (typically 105 seconds).

interface
The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ipv6 pim *(page 46)*.

holdtime
The hello holdtime, in seconds. The range is 1 to 65535. The default is 3.5 * **hello-interval**.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
            hello-holdtime holdtime
        }
    }
}
```

Use this command to configure a hello holdtime, the amount of time the system waits for a PIM Hello message before dropping a neighbor. The holdtime cannot be less than the current **`hello-interval`**. When the **`hello-interval`** is updated, the **`hello-holdtime`** is reviewed. If the **`hello-holdtime`** either is not configured or is configured but is less than the current **`hello-interval`**, it is set to 3.5 times the **`hello-interval`**. Otherwise, the current holdtime remains unchanged.

Use the `set` form of this command to specify the hello holdtime.

Use the `delete` form of this command to restore the hello holdtime to its default holdtime.

Use the `show` form of this command to display the hello holdtime configuration.

interfaces <interface> ipv6 pim hello-interval <interval>

Specifies the hello interval.

Syntax:
`set interfaces` *interface* **`ipv6 pim hello-interval`** *interval*

Syntax:
`delete interfaces` *interface* **`ipv6 pim hello-interval`**

Syntax:
`show interfaces` *interface* **`ipv6 pim hello-interval`**

Hello messages are sent every 30 seconds.

interface
The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ipv6 pim *(page 46)*.

interval
The hello interval, in seconds. The range is 1 to 65535. The default is 30.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
            hello-interval interval
        }
    }
}
```

Use this command to configure a hello interval, the interval at which PIM Hello messages are sent on an interface. When the **`hello-interval`** is updated, the **`hello-holdtime`** is reviewed. If the **`hello-holdtime`** either is not configured or is configured but is less than the current **`hello-interval`**, it is set to 3.5 times the **`hello-interval`**. Otherwise, the currently configured **`hello-holdtime`** remains unchanged.

Use the `set` form of this command to specify the hello interval.

Use the `delete` form of this command to restore the hello interval to its default interval.

Use the `show` form of this command to display the hello interval configuration.

interfaces <interface> ipv6 pim mode <mode>

Specifies the PIM mode that is enabled on an IPv6 interface.

Syntax:

`set interfaces` *interface* **`ipv6 pim mode`** *mode*

Syntax:

`delete interfaces` *interface* **`ipv6 pim mode`**

Syntax:

`show interfaces` *interface* **`ipv6 pim mode`**

interface

A type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ipv6 pim *(page 46)*.

mode

The PIM mode for an interface. The mode is one of the following:

`dense`: Enable PIM dense mode.

`dense-passive`: Enable passive operation for PIM dense mode.

`sparse`: Enable PIM sparse mode.

`sparse-passive`: Enable passive operation for PIM sparse mode.

`sparse-dense`: Enable PIM sparse-dense mode.

`sparse-dense-passive`: Enable passive operation for PIM sparse-dense mode.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
            mode [dense|dense-passive|sparse|sparse-dense|sparse-dense-passive|sparse-passive]
        }
    }
}
```

Use this command to specify the PIM mode on an interface. Use the **`dense`** or **`sparse`** keyword to enable PIM dense mode or PIM sparse mode on an interface. Use the **`dense-passive`** or **`sparse-passive`** keyword to stop PIM transactions on an interface, allowing only Internet Group Management Protocol (IGMP) to be active.

Use the `set` form of this command to specify the PIM mode for an interface.

Use the `delete` form of this command to disable PIM on an interface.

Use the `show` form of this command to display the PIM mode configuration.

interfaces <interface> ipv6 pim neighbor-filter <acl6>

Enables filtering of neighbors on an interface.

Syntax:

`set interfaces` *interface* **`ipv6 pim neighbor-filter`** *acl6*

Syntax:

`delete interfaces` *interface* **`ipv6 pim neighbor-filter`** *acl6*

Syntax:

show interfaces *interface* **ipv6 pim neighbor-filter**

interface

The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ipv6 pim *(page 46)*.

acl6

An IPv6 access list name. An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
            neighbor-filter acl6
        }
    }
}
```

Use this command to enable filtering of neighbors on an interface based on an access list. PIM terminates its adjacency with existing neighbors filtered by the access list, and does not establish adjacency with potential neighbors filtered by the access list.

Use the `set` form of this command to enable filtering of neighbors on an interface based on the specified access list.

Use the `delete` form of this command to disable filtering of neighbors on an interfaces based on the specified access list.

Use the `show` form of this command to display the neighbor filter configuration.

interfaces <interface> ipv6 pim propagation-delay <delay>

Specifies the propagation delay for PIM on an interface.

Syntax:

set interfaces *interface* **ipv6 pim propagation-delay** *delay*

Syntax:

delete interfaces *interface* **ipv6 pim propagation-delay**

Syntax:

show interfaces *interface* **ipv6 pim propagation-delay**

The propagation delay is 1000 milliseconds.

interface

The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ipv6 pim *(page 46)*.

delay

The propagation delay, in milliseconds. The range is 1000 to 5000.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
            propagation-delay delay
        }
    }
}
```

Use this command to specify the expected message propagation delay on the link. It is used by upstream routers to determine how long to wait for a Join override message before pruning an interface.

Use the `set` form of this command to specify the propagation delay for PIM on an interface.

Use the `delete` form of this command to restore the default propagation delay for PIM on an interface.

Use the `show` form of this command to display the propagation delay configuration.

interfaces <interface> ipv6 pim state-refresh origination-interval <interval>

Specifies the PIM-Dense Mode (PIM-DM) State Refresh origination interval on an interface.

Syntax:

```
set interfaces interface ipv6 pim origination-interval interval
```

Syntax:

```
delete interfaces interface ipv6 pim origination-interval
```

Syntax:

```
show interfaces interface ipv6 pim origination-interval
```

The PIM-DM State-Refresh origination interval is 60 seconds.

interface
: The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ipv6 pim *(page 46)*.

interval
: The PIM-DM State Refresh origination interval, in seconds. The range is 1 to 100. The default is 60.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
            state-refresh {
                origination-interval interval
            }
        }
    }
}
```

Use this command to specify the PIM-DM State Refresh origination interval on an interface. This interval is the amount of time between PIM-DM State Refresh control messages.

Use the `set` form of this command to specify the PIM-DM State Refresh origination interval on an interface.

Use the `delete` form of this command to restore the PIM-DM State Refresh origination interval to 60 seconds.

Use the `show` form of this command to display the configuration of a PIM-DM State Refresh interval.

interfaces <interface> ipv6 pim unicast-bsm

Enables the sending and receiving of unicast Bootstrap Messages (BSM) on an interface.

Syntax:

```
set interfaces interface ipv6 pim unicast-bsm
```

Syntax:

```
delete interfaces interface ipv6 pim unicast-bsm
```

Syntax:
`show interfaces` *interface* **`ipv6 pim`**

Unicast bootstrap messaging is disabled on an interface.

interface
: The type of interface. For a list of supported interfaces and detailed syntax, see interfaces <interface> ipv6 pim *(page 46)*.

Configuration mode

```
interfaces interface {
    ipv6 {
        pim {
            unicast-bsm
        }
    }
}
```

Use this command to enable the sending and receiving of unicast BSM on an interface. This command provides support for older versions of the Bootstrap Router (BSR) specification. This specification specifies the use of unicast BSM to refresh the state of new or restarting neighbors.

Use the `set` form of this command to enable support for unicast BSM on an interface.

Use the `delete` form of this command to disable support for unicast bootstrap messaging on an interface.

Use the `show` form of this command to display the configuration of unicast bootstrap messaging.

monitor protocol multicast pim

Sets debugging options for PIM.

Syntax:
`monitor protocol multicast pim` [[**background** { **start** | **stop** }] | { **enable** | **disable** } **ipv6** [**events** | **mfc** | **mib** | **mtrace** | **nexthop** | **nsm** | **packet** [**in** | **out**] | **state** | **timer** [**assert** [**at**] | **bsr** [**bst** | **crp**] | **hello** [**ht** | **nlt** | **tht**] | **joinprune** [**et** | **jt** | **kat** | **ot** | **ppt**] | **register** [**rst**]]]]

PIM debugging is disabled.

`background`
: Performs debugging operations in the background.

`start`
: Starts debugging in the background.

`stop`
: Stops debugging in the background.

`enable`
: Enables the specified debugging option.

`disable`
: Disables the specified debugging option.

`events`
: Enables debugging for PIM events.

`mfc`
: Enables debugging for Multicast Forwarding Cache (MFC) updates.

`mib`
: Enables debugging for Management Information Base (MIB) entries.

`mtrace`
: Enables debugging for Multicast Traceroute (MTRACE) messages.

`nexthop`
: Enables debugging for Reverse Path Forwarding (RPF) neighbor nexthop cache handling.

`nsm`
: Enables debugging for Network Services Module (NSM) messages.

packet
Enables debugging for PIM packets.

in
Enables debugging for incoming PIM packets.

out
Enables debugging for outgoing PIM packets.

state
Enables debugging for PIM states.

timer
Enables debugging for PIM timers.

assert
Enables debugging for PIM assert timers.

at
Enables debugging for PIM assert timer.

bsr
Enables debugging for PIM BSR timers.

bst
Enables debugging for PIM bootstrap timer.

crp
Enables debugging for PIM Candidate-RP timer.

hello
Enables debugging for various PIM timers.

ht
Enables debugging for PIM Hello timer.

nlt
Enables debugging for PIM Neighbor Liveliness timer.

tht
Enables debugging for PIM Triggered Hello timer.

joinprune
Enables debugging for various PIM JoinPrune timers.

et
Enables debugging for PIM JoinPrune Expiry timer.

jt
Enables debugging for PIM JoinPrune Upstream Join timer.

kat
Enables debugging for PIM JoinPrune Keepalive timer.

ot
Enables debugging for PIM JoinPrune Upstream Override timer.

ppt
Enables debugging for PIM JoinPrune Prune Pending timer.

register
Enables debugging for various PIM register timers.

rst
Enables debugging for PIM Register Stop timer.

Operational mode

Use this command to enable or disable debugging for PIM and to set PIM debugging options.

The following example shows how to start debugging for IPv6 PIM events.

```
        vyatta@vyatta:~$monitor protocol multicast pim enable ipv6 event
```

protocols pim6 accept-register list <acl6>

Allows the Rendezvous Point (RP) to accept Register messages only from multicast sources identified in a given access list.

Syntax:
`set protocols pim6 accept-register list` *acl6*

Syntax:
`delete protocols pim6 accept-register list` *acl6*

Syntax:
`show protocols pim6 accept-register list`

The RP accepts Register messages from all multicast sources.

acl6
An IPv6 access list name. An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

Configuration mode

```
protocols {
    pim6 {
        accept-register {
            list acl6
        }
    }
}
```

Use this command to configure the RP router to filter multicast sources identified by the specified access list. The RP accepts Register messages sent only by the sources specified in the access list. By default, the RP accepts Register messages from all multicast sources.

Use the **`set`** form of this command to enable the RP to accept Register messages only from multicast sources identified in a given access list.

Use the `delete` form of this command to restore the default behavior of Register handling.

Use the `show` form of this command to display accept-register list configuration.

protocols pim6 anycast-rp <rp-address> anycast-rp-peer <rp-peer-address>

Specifies the address of an Rendezvous Point (RP) member in an Anycast-RP set.

Syntax:
`set protocols pim6 anycast-rp` *rp-address* **`anycast-rp-peer`** *rp-peer-address*

Syntax:
`delete protocols pim6 anycast-rp` *rp-address* **`anycast-rp-peer`** *rp-peer-address*

Syntax:
`show protocols pim6 anycast-rp` *rp-address* **`anycast-rp-peer`**

rp-address
The unicast IPv6 address of an Anycast-RP set.

rp-peer-address
The IPv6 address of an RP member of an Anycast-RP set.

Configuration mode

```
protocols {
    pim6 {
        anycast-rp rp-address {
            anycast-rp-peer rp-peer-address
        }
    }
}
```

Use this command to specify the IPv6 address of an RP member of an Anycast-RP set. Register messages are copied to and sent from this address.

Use the `set` form of this command to specify the IPv6 address of an RP member of an Anycast-RP set.

Use the `delete` form of this command to remove the IPv6 address of an RP member of an Anycast-RP set.

Use the `show` form of this command to display the RP member configuration.

protocols pim6 bsr-candidate <interface>

Sets the candidate bootstrap router (BSR) status by using the IPv6 address of the specified interface.

Syntax:
`set protocols pim6 bsr-candidate` *interface* [**`hash-mask`** *mask*] [**`priority`** *priority*]

Syntax:
`delete protocols pim6 bsr-candidate` *interface* [**`hash-mask`**] [**`priority`**]

Syntax:
`show protocols pim6 bsr-candidate` *interface* [**`hash-mask`**] [**`priority`**]

mask
The hash mask length for RP selection. The range is 0 to 32. The default is 10.

priority
The priority for the BSR candidate. The range is 0 to 255. The default is 64.

Configuration mode

```
protocols {
    pim6 {
        bsr-candidate interface {
            hash-mask mask
            priority priority
        }
    }
}
```

Use this command to set the candidate BSR status by using the IPv6 address of the specified interface.

Use the `set` form of this command to set the candidate BSR status by using the IPv6 address of the specified interface.

Use the `delete` form of this command to remove the candidate BSR status from the interface.

Use the `show` form of this command to display the candidate BSR configuration.

`protocols pim6 dense-group <ip-address>`

Uses the dense-group keyword to force the interface PIM mode for a group to be dense - even if an RP(G) is present.

Syntax:
`set protocols pim6` **`dense-group`** *ip-address*

Syntax:
`delete protocols pim6` **`dense-group`** *ip-address*

Syntax:

```
show protocols pim6  dense-group ip-address
```

ip-address

The IPv6 address for the data plane. The format of the IPv6 address is h:h:h:h:h:h:h:h/x.

Configuration mode.

```
protocols {

        pim6 {
                dense-group ff1e::225:0:0:33
                rp-address beef::5:5:5:5 {
                        list 1
                }
        }
}
```

PIM dense-groups are added to force designated groups to be dense when otherwise an RP(G) would be present (useful given that access lists in the context of PIM do not support "deny" entries - just "permit").

Of particular significance is that SDM requires at least one RP to be present in the network in order to function - even in the absence of an RP(G) for any multicast group. Without an RP, DM traffic will not flow via SDM configured interfaces (unless connected interfaces on adjoining routers are configured for DM).

Use the `set` version of the command to force the interface PIM mode for a group to be dense.

Use the `delete` version of the command to undo the PIM configuration.

Use the `show` version of the command to display the PIM configuration.

The following example shows the use of dense-group and access-list in the context of SDM.

```
policy {
        route {

                access-list6 1 {
                        rule 1 {
                                action permit
                                source {
                                        network ff1e::225:0:0:0/120
                                }
                        }
                }
        }
}
protocols {

        pim6 {
                dense-group ff1e::225:0:0:33
                rp-address beef::5:5:5:5 {
                        list 1
                }
        }
}
```

Note: The group must be defined as "source" rather than "destination".

protocols pim6 ignore-rp-set-priority

Specifies that the RP-SET priority is to be ignored in Rendezvous Point (RP) selection.

Syntax:

`set protocols pim6 ignore-rp-set-priority`

Syntax:
`delete protocols pim6 ignore-rp-set-priority`

Syntax:
`show protocols pim6`

The RP-SET priority is used in RP selection.

Configuration mode

```
protocols {
    pim6 {
        ignore-rp-set-priority
    }
}
```

Use this command to specify that the RP-SET priority is to be ignored and that only the hash value is to be used in RP selection. This command provides interoperability with older Cisco IOS versions.

Use the `set` form of this command to specify that the RP-SET priority value is to be ignored and that only the hash value is to be used in RP selection.

Use the `delete` form of this command to restore the default RP selection mechanism by using the RP-SET priority.

Use the `show` form of this command to display the PIM configuration.

protocols pim6 join-prune-timer <timer>

Sets the PIM join/prune timer.

Syntax:
`set protocols pim6 join-prune-timer` *timer*

Syntax:
`delete protocols pim6 join-prune-timer`

Syntax:
`show protocols pim6 join-prune-timer`

The join/prune timer is 210 seconds.

timer

The join/prune timer, in seconds. The range is 1 to 65535. The default is 210.

Configuration mode

```
protocols {
    pim6 {
        join-prune-timer timer
    }
}
```

Use this command to specify the PIM join/prune timer.

Use the `set` form of this command to specify the PIM join/prune timer.

Use the `delete` form of this command to remove the PIM join/prune timer.

Use the `show` form of this command to display the PIM join/prune timer configuration.

protocols pim6 legacy-register-checksum [group-list <acl6>]

Specifies that the Register checksum should be calculated over the whole packet.

Syntax:
`set protocols pim6 legacy-register-checksum` [**`group-list`** *acl6*]

Syntax:
`delete protocols pim6 legacy-register-checksum` [**`group-list`**]

Syntax:
`show protocols pim6 legacy-register-checksum` [**`group-list`**]

The Register checksum is calculated over only the packet header.

acl6

An IPv6 access list name. An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

Configuration mode

```
protocols {
    pim6 {
        legacy-register-checksum {
            group-list acl6
        }
    }
}
```

Use this command to specify that the Register checksum should be calculated over the whole packet rather than only over the packet header. This command is used to accommodate operations with older Cisco IOS versions.

Use the `set` form of this command to specify that the Register checksum should be calculated over the whole packet rather than only over the packet header.

Use the `delete` form of this command to restore the default operation.

Use the `show` form of this command to display the configuration of legacy register checksums.

protocols pim6 log

Enables PIM6 logs.

Syntax:
`set protocols pim6 log` { **`all`** | **`events`** | **`mfc`** | **`mib`** | **`mtrace`** | **`nexthop`** | **`nsm`** | **`packet`** | **`state`** | **`timer`** }

Syntax:
`delete protocols pim6 log` { **`all`** | **`events`** | **`mfc`** | **`mib`** | **`mtrace`** | **`nexthop`** | **`nsm`** | **`packet`** | **`state`** | **`timer`** }

Syntax:
`show protocols pim6 log` { **`all`** | **`events`** | **`mfc`** | **`mib`** | **`mtrace`** | **`nexthop`** | **`nsm`** | **`packet`** | **`state`** | **`timer`** }

None

`all`

Enables all PIM6 logs.

`events`

Enables PIM6 debugging of general configuration and virtual routing.

mfc

Enables PIM6 debugging for MFC updates.

mib

Enables PIM6 debugging for MIB entries.

mtrace

Enables PIM6 debugging for MTRACE messages.

nexthop

Enables PIM6 debugging for nexthop cache handling for RPF neighbors .

nsm

Enables PIM6 debugging for PIM NSM.

packet

E nables PIM6 debugging for PIM packets.

state

E nables PIM6 debugging for PIM states.

timer

Enables PIM6 debugging for PIM timers.

Configuration mode

```
protocols {
      pim6 {
        log {
           all
           events
           mfc
           mib
           msdp
           mtrace
           nexthop
           nsm
           packet
           state
           timer
         }
      }
}
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM)6 logs.

Use the `delete` form of this command to remove PIM6 logs.

Use the `show` form of this command to view PIM6 logs.

protocols pim6 log timer

Enables PIM6 debugging for PIM timers.

Syntax:

`set protocols pim6 log timer` { **all** | **assert** | **bsr** | **hello** | **joinprune** | **register** }

Syntax:

`delete protocols pim6 log timer` { **all** | **assert** | **bsr** | **hello** | **joinprune** | **register** }

Syntax:

`show protocols pim6 log timer` { **all** | **assert** | **bsr** | **hello** | **joinprune** | **register** }

None

all

Enables PIM6 debugging for all PIM6 timers.

assert

Enables PIM6 debugging for all PIM assert timers.

bsr

Enables PIM6 debugging for all PIM BSR timers.

hello

Enables PIM6 debugging for various PIM timers.

joinprune

Enables PIM6 debugging for PIM join-prune timers.

register

Enables PIM6 debugging for various PIM register timers.

Configuration mode

```
protocols {
      pim6 {
        log {
          timer {
            all
            assert
            bsr
            hello
            joinprune
            register
          }
        }
      }
}
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM)6 timer logs.

Use the `delete` form of this command to remove PIM6 timer logs.

Use the `show` form of this command to view PIM6 timer logs.

protocols pim6 log packet

Enables PIM6 debugging of PIM packets.

Syntax:

set protocols pim6 log packet { **all** | **rcv** | **send** }

Syntax:

delete protocols pim6 log packet { **all** | **rcv** | **send** }

Syntax:

show protocols pim6 log packet { **all** | **rcv** | **send** }

None

all

Enables all PIM6 packet logs.

rcv

Enables PIM6 debugging for all incoming PIM packets.

send

Enables PIM6 debugging for all outgoing PIM packets.

Configuration mode

```
protocols {
        pim6 {
          log {
            packet {
              all
              rcv
              send
```

```
                   }
                 }
               }
             }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM) 6 packet logs.

Use the `delete` form of this command to remove PIM6 packet logs.

Use the `show` form of this command to view PIM6 packet logs.

protocols pim6 log timer assert

Enables PIM6 debugging for PIM assert timers.

Syntax:

```
set protocols pim6 log timer assert { all | at }
```

Syntax:

```
delete protocols pim6 log timer assert { all | at }
```

Syntax:

```
show protocols pim6 log timer assert { all | at }
```

None

`all`

Enables PIM6 debugging of all PIM assert timers.

`at`

Enables PIM6 debugging of PIM assert timers.

Configuration mode

```
protocols {
      pim6 {
        log {
         timer {
           assert{
             all
             at
                  }
                }
               }
             }
          }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM)6 assert timer logs.

Use the `delete` form of this command to remove PIM6 assert timer logs.

Use the `show` form of this command to view PIM6 assert timer logs.

protocols pim6 log timer bsr

Enables PIM6 debugging for PIM BSR timer.

Syntax:

```
set protocols pim6 log timer bsr { all | bst | crp }
```

Syntax:

```
delete protocols pim6 log timer bsr { all | bst | crp }
```

Syntax:

`show protocols pim6 log timer bsr { all | bst | crp }`

None

`all`
: Enables PIM6 debugging of all PIM BSR timers.

`bst`
: Enables PIM6 debugging of only bootstrap timers.

`crp`
: Enables PIM6 debugging of only candidate-RP timers.

Configuration mode

```
protocols {
       pim6 {
          log {
            timer {
               bsr {
                 all
                 bst
                 crp
                    }
                   }
                 }
               }
              }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM)6 BSR timer logs.

Use the `delete` form of this command to remove PIM6 timer BSR timer logs.

Use the `show` form of this command to view PIM6 timer BSR timer logs.

protocols pim6 log timer hello

Enables debugging for PIM6 hello timers.

Syntax:
`set protocols pim6 log timer hello { all | ht | nlt | tht }`

Syntax:
`delete protocols pim6 log timer hello { all | ht | nlt | tht }`

Syntax:
`show protocols pim6 log timer hello { all | ht | nlt | tht }`

None

`all`
: Enables debugging of all PIM6 hello timers.

`ht`
: Enables debugging of only PIM6 hello timers.

`nlt`
: Enables debugging of only PIM6 neighbor liveliness timers.

`tht`
: Enables debugging of only PIM6 triggerred hello timers.

Configuration mode

```
protocols {
        pim6 {
          log {
           timer {
            hello {
               all
               ht
               nlt
               tht
                   }
                  }
                 }
                }
               }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM6) timer hello logs.

Use the `delete` form of this command to remove PIM6 timer hello logs.

Use the `show` form of this command to view PIM6 hello logs.

protocols pim6 log timer joinprune

Enables PIM6 debugging for PIM join-prune timers.

Syntax:

`set protocols pim6 log timer joinprune` { **all** | **et** | **jt** | **kat** | **ot** | **ppt** }

Syntax:

`show protocols pim6 log timer joinprune` { **all** | **et** | **jt** | **kat** | **ot** | **ppt** }

Syntax:

`delete protocols pim6 log timer joinprune` { **all** | **et** | **jt** | **kat** | **ot** | **ppt** }

None

all

Enables debugging of all PIM join-prune timers.

et

Enables debugging of PIM join-prune expiry timers.

jt

Enables debugging of PIM join-prune upstream join timers.

kat

Enables debugging of PIM join-prune keep-alive timers.

ot

Enables debugging of PIM join-prune over-ride timers.

ppt

Enables debugging of PIM joinprune prune-pending timers.

Configuration mode

```
protocols {
        pim6 {
          log {
            timer {
             joinprune {
               all
               et
               jt
               kat
               ot
               ppt
                       }
                    }
                  }
                }
               }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM)6 join-prune timer logs.

Use the `delete` form of this command to remove PIM6 join-prune timer logs.

Use the `show` form of this command to view PIM6 join-prune timer logs.

protocols pim6 log timer register

Enables PIM6 debugging for PIM register timers.

Syntax:

`set protocols pim6 log timer register { all | rst }`

Syntax:

`delete protocols pim6 log timer register { all | rst }`

Syntax:

`show protocols pim6 log timer register { all | rst }`

None

`all`

Enables PIM6 debugging of all PIM register timers.

`rst`

Enables PIM6 debugging of only PIM register-stop timers.

Configuration mode

```
protocols {
       pim6 {
          log {
            timer {
              register {
                 all
                 rst
                       }
                     }
                   }
                 }
               }
```

Use the `set` form of this command to enable Protocol Independent Multicast (PIM)6 register timer logs.

Use the `delete` form of this command to remove PIM6 register timer logs.

Use the `show` form of this command to view PIM6 register timer logs.

protocols pim6 register-kat <timer>

Specifies the Keepalive Timer (KAT) for (S,G) states at the Rendezvous Point (RP).

Syntax:

`set protocols pim6 register-kat` *timer*

Syntax:

`delete protocols pim6 register-kat`

Syntax:

`show protocols pim6 register-kat`

The KAT for (S,G) states is 210 seconds.

timer

The KAT, in seconds. The range is 1 to 65535. The default is 210.

Configuration mode

```
protocols {
    pim6 {
        register-kat timer
    }
}
```

Use this command to specify the KAT for (S,G) states at the RP to monitor Register messages.

Use the `set` form of this command to specify the KAT for (S,G) states at the RP to monitor Register messages.

Use the `delete` form of this command to restore the KAT to 210 seconds.

Use the `show` form of this command to display register KAT configuration.

protocols pim6 register-rate-limit <rate>

Specifies the rate at which Register messages are sent by this designated router (DR).

Syntax:

`set protocols pim6 register-rate-limit` *rate*

Syntax:

`delete protocols pim6 register-rate-limit`

Syntax:

`show protocols pim6 register-rate-limit`

There is no limit to the rate at which Register messages are sent by the DR.

rate

The rate at which Register messages are sent by the DR, in packets per second. The range is 0 to 65535. The default is 0, meaning “no limit.”

Configuration mode

```
protocols {
    pim6 {
        register-rate-limit rate
    }
}
```

Use this command to specify the rate of Register messages sent by this DR. This rate is for each (S,G) state. The rate is not system wide.

Use the `set` form of this command to specify the rate of Register messages sent by this DR.

Use the `delete` form of this command to restore the rate of Register messages sent by this DR to its default value.

Use the `show` form of this command to display Register rate limit configuration.

protocols pim6 register-rp-reachability

Enables Rendezvous Point (RP) reachability checking for PIM Registers at the designated router (DR).

Syntax:

```
set protocols pim6 register-rp-reachability
```

Syntax:

```
delete protocols pim6 register-rp-reachability
```

Syntax:

```
show protocols pim6
```

RP reachability is not checked.

Configuration mode

```
protocols {
    pim6 {
         register-rp-reachability
    }
}
```

Use this command to enable RP reachability checking for PIM Registers at the DR.

Use the `set` form of this command to enable RP reachability checking for PIM Registers at the DR.

Use the `delete` form of this command to restore no checking for reachability.

Use the `show` form of this command to display the configuration of Register RP reachability.

protocols pim6 register-source

Specifies the source of Register messages sent by this designated router (DR).

Syntax:

```
set protocols pim6 register-source { address source | interface interface }
```

Syntax:

```
delete protocols pim6 register-source { address | interface }
```

Syntax:

```
show protocols pim6 register-source [ address | interface ]
```

The IPv6 address of the Reverse Path Forwarding (RPF) interface that faces the source host.

source

An IPv6 address to use as the source of Register messages.

interface

An interface to use as the source of Register messages. Note that it is not necessary for PIM to be enabled on this interface.

Configuration mode

```
protocols {
    pim6 {
        register-source {
            [address source | interface interface]
        }
    }
}
```

Use this command to specify the source of Register messages sent by this DR. The specified address must be reachable so that the Rendezvous Point (RP) router can send Register-Stop messages in response. The Register source address is usually the address of the loopback interface, though it can be another physical address. The specified address must be advertised by unicast routing protocols on the DR.

Use the `set` form of this command to specify the source of Register messages sent by this DR.

Use the `delete` form of this command to restore the default source of Register messages sent by the DR.

Use the `show` form of this command to display Register source configuration.

protocols pim6 register-suppression-timer <timer>

Specifies the register-suspension time.

Syntax:

```
set protocols pim6 register-suppression-timer timer
```

Syntax:

```
delete protocols pim6 register-suppression-timer
```

Syntax:

```
show protocols pim6 register-suppression-timer
```

The register-suppression time is 60 seconds.

timer

The register-suppression time, in seconds. The range is 1 to 65535. The default is 60.

Configuration mode

```
protocols {
    pim6 {
        register-suppression-timer timer
    }
}
```

Use this command to specify the register-suppression time. On a designated router (DR), this configuration modifies the register-suppression time. On a Rendezvous Point (RP) router, this configuration modifies the RPkeepalive-period if protocols pim6 register-kat <timer> *(page 65)* is not used.

Use the `set` form of this command to specify the register-suppression time.

Use the `delete` form of this command to restore the register-suppression time to 60 seconds.

Use the `show` form of this command to display the configuration of the register-suppression time.

protocols pim6 rp-address <rp-addr>

Specifies a static rendezvous point (RP) address for multicast groups.

Syntax:

```
set protocols pim6 rp-address rp-addr [ list acl6 | override ]
```

Syntax:

```
delete protocols pim6 rp-address rp-addr [ list | override ]
```

Syntax:

```
show protocols pim6 rp-address rp-addr [ list | override ]
```

rp-addr

The unicast IPv6 address of the RP set.

acl6

An IPv6 access list name. An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

`override`

Specifies that static RP addresses take precedence over dynamically learned RP addresses.

Configuration mode

```
protocols {
    pim6 {
        rp-address rp-addr {
            list acl6
            override
        }
    }
}
```

The PIM implementation supports multiple statically configured rendezvous points (RPs). It also supports the use of both statically configured RPs and those selected through the bootstrap router (BSR) mechanism simultaneously. Note the following:

- If multiple static RP addresses are available for a group range, then the one with the highest IP address is chosen.
- RP addresses configured for a multicast group through the BSR mechanism take precedence over those configured statically unless the **`override`** keyword is used. In those cases, a statically configured RP address takes precedence.
- Configuring multiple static RPs with the same RP address is not allowed.
- One static RP address can be configured for multiple group ranges by using access lists. The static RP address can either be configured for the whole multicast group range (that is, FF00::/8) or for specific group ranges if an access list is specified. When an access list is specified, the static RP address is configured for all the group ranges represented by Permit filters in the access list.
- Only Permit filters in access lists are considered as valid group ranges. The default Permit filter ::/0 is converted to the default multicast filter FF00::/8.
- After configuration, the RP address is inserted into a static RP group tree based on the configured group ranges. For each group range, multiple static RPs are maintained in a list. This list is sorted in descending order of IP addresses. When selecting static RPs for a group range, the first element of the list (the statically configured RP with the highest IP address) is selected.
- When an RP address is deleted, the static RP is removed from all the existing group ranges and RPs are recomputed for existing Tree Information Base (TIB) states if required.

Use the `set` form of this command to specify a static RP address for multicast groups.

Use the `delete` form of this command to remove the configuration of static RP addresses.

Use the `show` form of this command to display the configuration of static RP addresses.

protocols pim6 rp-candidate <interface>

Specifies that the router is a candidate Rendezvous Point (RP).

Syntax:
`set protocols pim6 rp-candidate` *interface* [**`group-list`** *acl6* | **`interval`** *interval* | **`priority`** *priority*]

Syntax:
`delete protocols pim6 rp-candidate` *interface* [**`group-list`** | **`interval`** | **`priority`**]

Syntax:
`show protocols pim6 rp-candidate` *interface* [**`group-list`** | **`interval`** | **`priority`**]

interface
The interface to set with candidate RP status.

acl6
An IPv6 access list name. An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

interval
The candidate RP advertisement interval, in seconds. The range is 1 to 16383. The default is 60.

priority

The candidate RP priority. The range is 0 to 255. The default is 192.

Configuration mode

```
protocols {
    pim6 {
        rp-candidate interface {
            group-list acl6
            interval interval
            priority priority
        }
    }
}
```

Use this command to specify that the router is a candidate RP by using the IP address of the specified interface.

Use the `set` form of this command to specify that this router is a candidate RP.

Use the `delete` form of this command to remove the router as a candidate RP.

Use the `show` form of this command to display the candidate RP configuration.

protocols pim6 rp-embedded

Enables the embedded-Rendezvous Point (RP) feature.

Syntax:

`set protocols pim6 rp-embedded`

Syntax:

`delete protocols pim6 rp-embedded`

Syntax:

`show protocols pim6 embedded`

Configuration mode

```
protocols {
    pim6 {
        rp-embedded
    }
}
```

Use this command to enable the embedded-RP feature.

Use the `set` form of this command to enable the RP-embedded feature.

Use the `delete` form of this command to disable the RP-embedded feature.

Use the `show` form of this command to display the RP-embedded configuration.

protocols pim6 spt-threshold

Enables the last-hop PIM router to switch to shortest-path tree (SPT).

Syntax:

`set protocols pim6 spt-threshold` [**`infinity`**] [**`group-list`** *acl6*]

Syntax:

`delete protocols pim6 spt-threshold` [**`infinity`**] [**`group-list`**]

Syntax:

```
show protocols pim6 spt-threshold [ infinity ] [ group-list ]
```

`infinity`

Sets the spt threshold to infinity, which disables the ability of the last-hop PIM router to switch to SPT.

acl6

An IPv6 access list name. An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

Configuration mode

```
protocols {
    pim6 {
        spt-threshold
            infinity
                group-list acl6
    }
}
```

Use this command to enable the last-hop PIM router to switch to SPT. To set the SPT threshold to infinity and disable the ability of the last-hop PIM router to switch to SPT, use the **`infinity`** option with this command.

Use the `set` form of this command to enable the last-hop PIM router to switch to SPT.

Use the `delete` form of this command to remove SPT threshold configuration. To remove the **`infinity`** option with this command, use the `delete` form of this command with this option, which enables the ability of the last-hop PIM router to switch to SPT.

Use the `show` form of this command to display SPT threshold configuration.

protocols pim6 ssm default

Enables Source Specific Multicast (SSM) and uses a default range of IP multicast addresses.

Syntax:

```
set protocols pim6 ssm default
```

Syntax:

```
delete protocols pim6 ssm default
```

Syntax:

```
show protocols pim6 ssm default
```

Configuration mode

```
protocols {
    pim6 {
        ssm {
            default
        }
    }
}
```

Use this command to configure SSM and use the default range of IP multicast addresses (that is, FF3x::/32). Use protocols pim6 ssm range <acl6> *(page 72)* to define the SSM range to be other than the default. When an SSM range of multicast addresses is defined, the no (*,G) or (S,G,rpt) state is initiated for groups in the SSM range. Messages corresponding to these states are neither accepted nor originated in the SSM range.

Use the `set` form of this command to enable SSM and use a default range of IP multicast addresses.

Use the `delete` form of this command to disable SSM.

Use the `show` form of this command to display SSM default configuration.

protocols pim6 ssm range <acl6>

Enables Source Specific Multicast (SSM) and defines a range of IPv6 multicast addresses based on an access list.

Syntax:

```
set protocols pim6 ssm range acl6
```

Syntax:

```
delete protocols pim6 ssm range
```

Syntax:

```
show protocols pim6 ssm range
```

acl6

An IPv6 access list name. An access control list is a type of routing policy; see AT&T Vyatta Network Operating System Routing Policies Configuration Guide for information on creating them.

Configuration mode

```
protocols {
    pim6 {
        ssm {
            range acl6
        }
    }
}
```

Use this command to enable SSM and define a range of IPv6 multicast addresses based on an access list.

Use the `set` form of this command to enable SSM and define a range of IPv6 multicast addresses based on an access list.

Use the `delete` form of this command to disable SSM.

Use the `show` form of this command to display SSM range configuration.

reset ipv6 mroute

Deletes multicast route table entries and multicast routes at the PIM protocol level.

Syntax:

```
reset ipv6 mroute [ group group [ source source [ pim { dense-mode | sparse-mode } ] ] | pim { dense-mode | sparse-mode } ]
```

group

A multicast group to delete. The format is an IPv6 multicast address.

source

A multicast source to delete. The format is an IPv6 host address.

`dense-mode`

Deletes the multicast route table for PIM dense-mode.

`sparse-mode`

Deletes the multicast route table for PIM sparse-mode.

Operational mode

Use this command to delete multicast route table entries and multicast routes at the PIM protocol level. Used with no options, the command deletes all multicast route table entries and multicast routes. Used with options, the command deletes a subset of the multicast route table entries and multicast routes.

The following example shows how to delete all multicast route table entries and multicast routes at the PIM protocol level.

```
vyatta@vyatta:~$reset ipv6 mroute
```

reset ipv6 pim sparse-mode bsr rp-set

Deletes all rendezvous point (RP) sets learned from the PIM Bootstrap Router (BSR).

Syntax:

`reset ipv6 pim sparse-mode bsr rp-set`

Operational mode

Use this command to delete all RP sets learned through the PIM BSR.

The following example shows how to delete all RP sets learned through the PIM BSR.

```
vyatta@vyatta:~$reset ipv6 pim sparse-mode bsr rp-set
```

show ipv6 pim bsr-router

Displays the bootstrap router (BSR) PIM Version 2 (PIMv2) address.

Syntax:

`show ipv6 pim bsr-router`

Operational mode

Displays the BSR PIMv2 address.

The following example shows how to display the BSR PIMv2 address.

```
vyatta@vyatta:~$show ipv6 pim bsr-router
PIMv2 Bootstrap information
  BSR address: 10.10.11.35 (?)
  Uptime:      00:00:38, BSR Priority: 0, Hash mask length: 10
  Expires:     00:01:32
  Role: Non-candidate BSR
  State: Accept Preferred
```

show ipv6 pim interface

Displays PIM interface information.

Syntax:

`show ipv6 pim interface [ detail ]`

detail

Displays detailed information about the PIM interface.

Operational mode

Use this command to display PIM interface information.

Example: Examples:

In the following example, the Ver/Mode column displays "SD" when configured in Sparse-Dense Mode (SDM).

```
vyatta@vrfR3:~$ show ipv6 pim interface
Interface        VIFindex Ver/   Nbr    DR
                          Mode   Count  Prior
dp0s4            0        v2/SD   1      1
    Address       : fe80::5054:ff:feca:1c10
    Global Address: 2001::7
    DR            : this system
dp0s6            2        v2/SD   0      1
    Address       : fe80::5054:ff:fed8:5278
    Global Address: 2003::7
    DR            : this system
dp0s7            1        v2/SD   1      1
    Address       : fe80::5054:ff:fe22:b4fb
    Global Address: 2023::7
    DR            : fe80::5054:ff:fea7:c25b
```

In this example, the Mode field shows "Sparse-Dense" when configured as SDM.

```
sh ipv6 pim interface detail
dp0s4 (vif 0):
  Address fe80::5054:ff:feca:1c10, Mode: Sparse-Dense
  DR fe80::5054:ff:feca:1c10, DR's priority: 1
  Propagation delay is 1000 milli-seconds
  Router-ID:192.168.122.215 Local-ID 9
  Secondary addresses:
   2001::7
  Neighbors:
   fe80::5054:ff:fea9:b339
  PIM neighbor count: 1
  PIM neighbor holdtime: 105
  PIM neighbor hello interval: 30
  PIM configured DR priority: 1
  PIM border interface: no
  PIM Neighbor policy: not configured
```

show ipv6 pim local-members

Displays local membership information for a PIM interface.

Syntax:

`show ipv6 pim local-members` [*interface*]

interface

An interface for which to display local membership information.

Operational mode

Use this command to display local membership information for a PIM interface.

show ipv6 pim mroute

Displays the IPv6 PIM multicast routing table.

Syntax:

`show ipv6 pim mroute` [[**group** *group* [[**source** *source*] **detail**] | **detail**] | [**rfc** [**group** *group* [[**source** *source*] [**detail**]] | **detail**] | **source** *source* [**detail**] | **detail** | **summary**]

group

The multicast group entries to display. The format is an IPv6 multicast address.

source

The multicast source entries to display. The format is an IPv6 host address.

rfc

Displays information for a PIM multicast routing table (RFC style).

detail

Displays detailed information for a PIM multicast routing table.

summary

Displays summarized information for a PIM multicast routing table.

Operational mode

Use this command to display the IPv6 PIM multicast routing table. Used with no options, the command displays all entries for an IPv6 PIM multicast routing table. Used with the **group** option, **source** option, or both options, the command displays a subset of entries for the IPv6 PIM multicast routing table. Used with the **rfc** option, the command displays the IPv6 PIM multicast routing table entries in an RFC style.

show ipv6 pim neighbor

Displays PIM neighbor information.

Syntax:

`show ipv6 pim neighbor` [*interface* [*addr* [**detail**] | **detail**] | **detail**]

interface

An interface for which to display PIM neighbor information.

addr

The IPv6 address of a neighbor interface.

detail

Displays detailed PIM neighbor information.

Operational mode

Use this command to display PIM neighbor information. Used with no options, the command displays all PIM neighbor information. Used with the **interface** option, **addr** option, or both options, the command displays information for a subset of PIM neighbors.

show ipv6 pim nexthop

Displays next-hop information used by PIM.

Syntax:

`show ipv6 pim nexthop`

Operational mode

Displays next-hop information used by PIM.

show ipv6 pim rp-hash <group>

Displays the rendezvous point (RP) to select based on the group specified.

Syntax:

`show ipv6 pim rp-hash` *group*

group

A multicast group for which to determine the RP. The format is an IPv6 multicast address.

Operational mode

Displays the RP to select based on the group specified.

show ipv6 pim rp-mapping

Displays the group-to-Rendezvous Point (RP) mappings and the RP set.

Syntax:

```
show ipv6 pim rp-mapping
```

Operational mode

Displays the group-to-RP mappings and the RP set.

show monitoring protocols multicast pim6

Displays IPv6 multicast debugging information.

Syntax:

```
show monitoring protocols multicast pim6
```

Operational mode

Use this command to display IPv6 multicast debugging information.

The following example shows how to display IPv6 multicast debugging information.

```
vyatta@vyatta:~$show monitoring protocols multicast pim6
Debugging status:
PIMv6 event debugging is on
PIMv6 MFC debugging is on
PIMv6 state debugging is on
PIMv6 packet debugging is on
PIMv6 Hello HT timer debugging is on
PIMv6 Hello NLT timer debugging is on
PIMv6 Hello THT timer debugging is on
PIMv6 Join/Prune JT timer debugging is on
PIMv6 Join/Prune ET timer debugging is on
PIMv6 Join/Prune PPT timer debugging is on
PIMv6 Join/Prune KAT timer debugging is on
PIMv6 Join/Prune OT timer debugging is on
PIMv6 Assert AT timer debugging is on
PIMv6 Register RST timer debugging is on
PIMv6 Bootstrap BST timer debugging is on
PIMv6 Bootstrap CRP timer debugging is on
PIMv6 mib debugging is on
PIMv6 nexthop debugging is on
PIMv6 mtrace debugging is on
PIMv6 NSM debugging is on
```

Supported Interface Types

The following table shows the syntax and parameters of supported interface types. Depending on the command, some of these types may not apply.

Interface Type	Syntax	Parameters
Bridge	`bridge` *brx*	*brx*: The name of a bridge group. The name ranges from br0 through br999.

Interface Type	Syntax	Parameters
Data plane	`dataplane` *interface-name*	*interface-name*: The name of a data plane interface. Following are the supported formats of the interface name: • `dp`*x*`p`*y*`p`*z*—The name of a data plane interface, where — `dp`*x* specifies the data plane identifier (ID). Currently, only dp0 is supported. — `p`*y* specifies a physical or virtual PCI slot index (for example, p129). — `p`*z* specifies a port index (for example, p1). For example, dp0p1p2, dp0p160p1, and dp0p192p1. • `dp`*x*`em`*y* —The name of a data plane interface on a LAN-on-motherboard (LOM) device that does not have a PCI slot, where `em`*y* specifies an embedded network interface number (typically, a small number). For example, dp0em3. • `dp`*x*`s`*y*—The name of a data plane interface in a system in which the BIOS identifies the network interface card to reside in a particular physical or virtual slot *y*, where *y* is typically a small number. For example, for the dp0s2 interface, the BIOS identifies slot 2 in the system to contain this interface. • `dp`*x*`P`*n*`p`*y*`p`*z* —The name of a data plane interface on a device that is installed on a secondary PCI bus, where `P`*n* specifies the bus number. You can use this format to name data plane interfaces on large physical devices with multiple PCI buses. For these devices, it is possible to have network interface cards installed on different buses with these cards having the same slot ID. The value of *n* must be an integer greater than 0. For example, dp0P1p162p1 and dp0P2p162p1.

Interface Type	Syntax	Parameters
Data plane vif	`dataplane` *interface-name* `vif` *vif-id* [`vlan` *vlan-id*]	*interface-name*: Refer to the preceding description. *vif-id*: A virtual interface ID. The ID ranges from 1 through 4094. *vlan-id*: The VLAN ID of a virtual interface. The ID ranges from 1 through 4094.
Loopback	`loopback lo` or `loopback lo`*n*	*n*: The name of a loopback interface, where *n* ranges from 1 through 99999.
OpenVPN	`openvpn` *vtunx*	*vtunx*: The identifier of an OpenVPN interface. The identifier ranges from vtun0 through vtun*x*, where *x* is a nonnegative integer.
Tunnel	`tunnel` *tunx* or `tunnel` *tunx* `parameters`	*tunx*: The identifier of a tunnel interface you are defining. The identifier ranges from tun0 through tun*x*, where *x* is a nonnegative integer.
Virtual tunnel	`vti` *vtix*	*vtix*: The identifier of a virtual tunnel interface you are defining. The identifier ranges from vti0 through vti*x*, where *x* is a nonnegative integer. **Note:** Before you can configure a vti interface, you must configure a corresponding vpn. **Note:** This interface does not support IPv6.
VRRP	*parent-interface* `vrrp` `vrrp-group` *group*	*parent-interface*: The type and identifier of a parent interface; for example, data plane dp0p1p2 or bridge br999. *group*: A VRRP group identifier. The name of a VRRP interface is not specified. The system internally constructs the interface name from the parent interface identifier plus the VRRP group number; for example, dp0p1p2v99. Note that VRRP interfaces support the same feature set as does the parent interface.

List of Acronyms

Acronym	Description
ACL	access control list
ADSL	Asymmetric Digital Subscriber Line
AH	Authentication Header
AMI	Amazon Machine Image
API	Application Programming Interface
AS	autonomous system
ARP	Address Resolution Protocol
AWS	Amazon Web Services
BGP	Border Gateway Protocol
BIOS	Basic Input Output System
BPDU	Bridge Protocol Data Unit
CA	certificate authority
CCMP	AES in counter mode with CBC-MAC
CHAP	Challenge Handshake Authentication Protocol
CLI	command-line interface
DDNS	dynamic DNS
DHCP	Dynamic Host Configuration Protocol
DHCPv6	Dynamic Host Configuration Protocol version 6
DLCI	data-link connection identifier
DMI	desktop management interface
DMVPN	dynamic multipoint VPN
DMZ	demilitarized zone
DN	distinguished name
DNS	Domain Name System
DSCP	Differentiated Services Code Point
DSL	Digital Subscriber Line
eBGP	external BGP
EBS	Amazon Elastic Block Storage
EC2	Amazon Elastic Compute Cloud
EGP	Exterior Gateway Protocol
ECMP	equal-cost multipath
ESP	Encapsulating Security Payload
FIB	Forwarding Information Base
FTP	File Transfer Protocol
GRE	Generic Routing Encapsulation
HDLC	High-Level Data Link Control
I/O	Input/Output
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers

Acronym	Description
IGMP	Internet Group Management Protocol
IGP	Interior Gateway Protocol
IPS	Intrusion Protection System
IKE	Internet Key Exchange
IP	Internet Protocol
IPOA	IP over ATM
IPsec	IP Security
IPv4	IP Version 4
IPv6	IP Version 6
ISAKMP	Internet Security Association and Key Management Protocol
ISM	Internet Standard Multicast
ISP	Internet Service Provider
KVM	Kernel-Based Virtual Machine
L2TP	Layer 2 Tunneling Protocol
LACP	Link Aggregation Control Protocol
LAN	local area network
LDAP	Lightweight Directory Access Protocol
LLDP	Link Layer Discovery Protocol
MAC	medium access control
mGRE	multipoint GRE
MIB	Management Information Base
MLD	Multicast Listener Discovery
MLPPP	multilink PPP
MRRU	maximum received reconstructed unit
MTU	maximum transmission unit
NAT	Network Address Translation
NBMA	Non-Broadcast Multi-Access
ND	Neighbor Discovery
NHRP	Next Hop Resolution Protocol
NIC	network interface card
NTP	Network Time Protocol
OSPF	Open Shortest Path First
OSPFv2	OSPF Version 2
OSPFv3	OSPF Version 3
PAM	Pluggable Authentication Module
PAP	Password Authentication Protocol
PAT	Port Address Translation
PCI	peripheral component interconnect
PIM	Protocol Independent Multicast
PIM-DM	PIM Dense Mode
PIM-SM	PIM Sparse Mode
PKI	Public Key Infrastructure
PPP	Point-to-Point Protocol
PPPoA	PPP over ATM

Acronym	Description
PPPoE	PPP over Ethernet
PPTP	Point-to-Point Tunneling Protocol
PTMU	Path Maximum Transfer Unit
PVC	permanent virtual circuit
QoS	quality of service
RADIUS	Remote Authentication Dial-In User Service
RHEL	Red Hat Enterprise Linux
RIB	Routing Information Base
RIP	Routing Information Protocol
RIPng	RIP next generation
RP	Rendezvous Point
RPF	Reverse Path Forwarding
RSA	Rivest, Shamir, and Adleman
Rx	receive
S3	Amazon Simple Storage Service
SLAAC	Stateless Address Auto-Configuration
SNMP	Simple Network Management Protocol
SMTP	Simple Mail Transfer Protocol
SONET	Synchronous Optical Network
SPT	Shortest Path Tree
SSH	Secure Shell
SSID	Service Set Identifier
SSM	Source-Specific Multicast
STP	Spanning Tree Protocol
TACACS+	Terminal Access Controller Access Control System Plus
TBF	Token Bucket Filter
TCP	Transmission Control Protocol
TKIP	Temporal Key Integrity Protocol
ToS	Type of Service
TSS	TCP Maximum Segment Size
Tx	transmit
UDP	User Datagram Protocol
VHD	virtual hard disk
vif	virtual interface
VLAN	virtual LAN
VPC	Amazon virtual private cloud
VPN	virtual private network
VRRP	Virtual Router Redundancy Protocol
WAN	wide area network
WAP	wireless access point
WPA	Wired Protected Access