

---

Stream: Internet Engineering Task Force (IETF)  
RFC: [9128](#)  
Category: Standards Track  
Published: September 2021  
ISSN: 2070-1721  
Authors:  
X. Liu P. McAllister A. Peter M. Sivakumar Y. Liu  
*Volta Networks Metaswitch Networks Individual Juniper Networks Huawei Technologies*  
F. Hu  
*ZTE Corporation*

# RFC 9128

## YANG Data Model for Protocol Independent Multicast (PIM)

---

### Abstract

This document defines a YANG data model that can be used to configure and manage devices supporting Protocol Independent Multicast (PIM). The model covers the PIM protocol configuration, operational state, and event notifications data.

### Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at <https://www.rfc-editor.org/info/rfc9128>.

### Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions

with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction
  - 1.1. Terminology
  - 1.2. Tree Diagrams
  - 1.3. Prefixes in Data Node Names
2. Design of Data Model
  - 2.1. Scope of Model
  - 2.2. Optional Capabilities
  - 2.3. Datastore Applicability
  - 2.4. Module and Hierarchy Organization
  - 2.5. Position of Address Family in Hierarchy
3. Module Structure
  - 3.1. PIM Base Module
    - 3.1.1. High-Level Structure
    - 3.1.2. Global Data
    - 3.1.3. Per Address Family Data
    - 3.1.4. PIM Interface Modeling
    - 3.1.5. Neighbor Modeling
    - 3.1.6. Notifications
  - 3.2. PIM RP Module
    - 3.2.1. Static RPs
    - 3.2.2. BSRs
    - 3.2.3. RP State Data
    - 3.2.4. RP-to-Group Mappings
    - 3.2.5. Notifications
  - 3.3. PIM-SM Module
  - 3.4. PIM-DM Module

- 3.5. PIM-BIDIR Module
  - 4. Complete Tree Structure
    - 4.1. PIM Base Module
    - 4.2. PIM RP Module
    - 4.3. PIM-SM Module
    - 4.4. PIM-DM Module
    - 4.5. PIM-BIDIR Module
  - 5. Relationship to the PIM-STD-MIB
    - 5.1. pimInterfaceTable
    - 5.2. pimNeighborTable
    - 5.3. pimStarGTable
    - 5.4. pimSGTable
    - 5.5. pimSGRptTable
    - 5.6. pimBidirDFElectionTable
    - 5.7. pimStaticRPTTable
    - 5.8. pimAnycastRPSetTable
    - 5.9. pimGroupMappingTable
  - 6. PIM YANG Modules
    - 6.1. PIM Base Module
    - 6.2. PIM RP Module
    - 6.3. PIM-SM Module
    - 6.4. PIM-DM Module
    - 6.5. PIM-BIDIR Module
  - 7. Security Considerations
  - 8. IANA Considerations
  - 9. References
    - 9.1. Normative References
    - 9.2. Informative References
- Appendix A. Data Tree Example
- Acknowledgments

[Authors' Addresses](#)

## 1. Introduction

YANG [RFC7950] is a data modeling language that was introduced to model the configuration and operational state of a device managed using network management protocols such as the Network Configuration Protocol (NETCONF) [RFC6241] or RESTCONF [RFC8040]. YANG is now also being used as a component of other management interfaces, such as CLIs.

This document defines a YANG data model that can be used to configure and manage devices supporting Protocol Independent Multicast (PIM). This model supports the core PIM protocol, as well as many other features; see [Section 2.1](#). Non-core features are defined as optional in the provided data model.

### 1.1. Terminology

The terminology for describing YANG data models is found in [RFC7950].

The following abbreviations are used in this document and the defined model:

ASM:	Any-Source Multicast service model [RFC3569] [RFC4607]
BFD:	Bidirectional Forwarding Detection [RFC5880]
BSR:	Bootstrap Router [RFC5059]
DF:	Designated Forwarder [RFC5015]
DR:	Designated Router [RFC7761]
IGMP:	Internet Group Management Protocol [RFC3376]
MLD:	Multicast Listener Discovery [RFC3810]
mLDP:	Multipoint extensions for LDP [RFC6388]
MRIB:	Multicast Routing Information Base [RFC3973] [RFC5015] [RFC7761]
MSDP:	Multicast Source Discovery Protocol [RFC3618]
mVPN:	Multicast VPN
PIM:	Protocol Independent Multicast [RFC3973] [RFC5015] [RFC7761]
PIM-BIDIR:	Protocol Independent Multicast - Bidirectional Mode [RFC5015]
PIM-DM:	Protocol Independent Multicast - Dense Mode [RFC3973]
PIM-SM:	Protocol Independent Multicast - Sparse Mode [RFC7761]

---

RP:	Rendezvous Point [ <a href="#">RFC7761</a> ]
RPA:	Rendezvous Point Address [ <a href="#">RFC5015</a> ]
RPF:	Reverse Path Forwarding [ <a href="#">RFC3973</a> ] [ <a href="#">RFC5015</a> ] [ <a href="#">RFC7761</a> ]
RPT:	Rendezvous Point Tree [ <a href="#">RFC7761</a> ]
SPT:	Shortest Path Tree [ <a href="#">RFC7761</a> ]
SSM:	Source-Specific Multicast service model [ <a href="#">RFC3569</a> ] [ <a href="#">RFC4607</a> ]
VRF:	Virtual Routing and Forwarding

## 1.2. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [[RFC8340](#)].

In addition, the following notation is used as a placeholder at the location of the name of a tree node, to represent a section of nodes:

<summary description of a section of nodes>

## 1.3. Prefixes in Data Node Names

In this document, names of data nodes, actions, and other data model objects are often used without a prefix, as long as it is clear from the context in which YANG module each name is defined. Otherwise, names are prefixed using the standard prefix associated with the corresponding YANG module, as shown in [Table 1](#).

Prefix	YANG Module	Reference
yang	ietf-yang-types	<a href="#">[RFC6991]</a>
inet	ietf-inet-types	<a href="#">[RFC6991]</a>
if	ietf-interfaces	<a href="#">[RFC8343]</a>
rt	ietf-routing	<a href="#">[RFC8349]</a>
rt-types	ietf-routing-types	<a href="#">[RFC8294]</a>
bfd-types	ietf-bfd-types	<a href="#">[RFC9127]</a>

*Table 1: Prefixes and Corresponding YANG Modules*

## 2. Design of Data Model

### 2.1. Scope of Model

The model covers PIM Sparse Mode [RFC7761] (including the Source-Specific subset [RFC3569] [RFC4607]), Dense Mode [RFC3973], and Bidirectional PIM [RFC5015].

The PIM extensions represented in the model include BSRs [RFC5059] and Anycast-RPs [RFC4610].

The data model can be used to configure and manage these protocol features. The operational state data and statistics can be retrieved by this model. The protocol-specific notifications are also defined in the model.

This model does not cover other multicast protocols such as IGMP/MLD, MSDP, mVPN, or mLDP in-band signaling. It does not cover any configuration required to generate the MRIB. These will be specified in separate documents.

### 2.2. Optional Capabilities

This model is designed to represent the capabilities of devices supporting PIM with various specifications, including some with basic subsets of the PIM protocol. The main design goals of this document are that any major currently existing implementation may be said to support the base model and that the configuration of all implementations meeting the specification is easy to express through some combination of the features in the base model and simple vendor augmentations.

There is also value in widely supported features being standardized, to save work for individual vendors, and so that mapping between different vendors' configurations is not needlessly complicated. Therefore, these modules declare a number of features representing capabilities that not all deployed devices support.

The extensive use of feature declarations should also substantially simplify the capability negotiation process for a vendor's PIM implementation.

On the other hand, operational state parameters are not so widely designated as features, as there are many cases where the defaulting of an operational state parameter would not cause any harm to the system, and it is much more likely that an implementation without native support for a piece of operational state would be able to derive a suitable value for a state variable that is not natively supported.

For the same reason, wide constant ranges (for example, timer maxima and minima) are used in the model. It is expected that vendors will augment the model with any specific extensions and restrictions needed to adapt it to their vendor-specific implementations.

### 2.3. Datastore Applicability

This model conforms to the Network Management Datastore Architecture (NMDA) [[RFC8342](#)]. The operational state data is combined with the associated configuration data in the same hierarchy [[RFC8407](#)].

### 2.4. Module and Hierarchy Organization

This model defines several separate modules for modeling PIM configuration. Again, this separation makes it easier to express the specific capabilities of a PIM device. The module organization, along with the usage of the YANG extensible features such as identity, allows the model to be easily augmented for new capabilities.

The hierarchy of PIM configuration is designed so that objects that are only relevant for one situation or feature are collected in a container for that feature. For example, a configuration for PIM-SM that is not relevant for an SSM-only implementation is collected in an ASM container.

Where fields are not genuinely essential to protocol operation, they are marked as optional. Some fields are essential but have a default specified, so they need not be explicitly configured.

This module structure also applies, where applicable, to the operational state and notifications of the model.

### 2.5. Position of Address Family in Hierarchy

This document contains "address-family" as a node in the hierarchy multiple times: under both the interface list and the PIM instance.

The reasoning for this is to make it easier for implementations in which configuration options are not supported for specific address families.

For these implementations, the restriction that interface configuration must be address-family independent may be expressed either (1) as a vendor augmentation of an address-family-independent parameter above the address-family level or (2) by a constraint on the base model objects of a form similar to the following:

```
deviation "/rt:routing/rt:control-plane-protocols/"
+ "pim-base:pim/pim-base:interfaces/pim-base:interface/"
+ "pim-base:address-family" {

  deviate add {
    must "(address-family = 'rt:ipv4' and dr-priority = "
    + "../address-family[address-family = 'rt:ipv6']/"
    + "dr-priority) or "
    + "(address-family = 'rt:ipv6' and dr-priority = "
    + "../address-family[address-family = 'rt:ipv4']/"
    + "dr-priority)" {
      error-message
        "Error: The IPv6 DR priority must match the "
        + "IPv4 DR priority.";
      error-app-tag "dr-priority-mismatch";
    }
  }
}
```

## 3. Module Structure

### 3.1. PIM Base Module

The PIM base module defines the base framework not specific to any PIM mode and is imported by the other modules. The base module by itself does not provide sufficient data for any PIM mode to operate. Other mode-specific and feature-specific modules need to be implemented in addition to this module, depending on the feature set required by the implementation.

This model augments the core routing data model "ietf-routing" specified in [\[RFC8349\]](#). The PIM base model augments "/rt:routing/rt:control-plane-protocols" as opposed to augmenting "/rt:routing/rt:control-plane-protocols/rt:control-plane-protocol", as the latter would allow multiple protocol instances, while the PIM protocol is designed to be enabled or disabled as a single protocol instance on a network instance or a logical network element.



### 3.1.1. High-Level Structure

The high-level structure of the model is shown below:

```
module: ietf-pim-base
  augment /rt:routing/rt:control-plane-protocols:
    +--rw pim!
      +--rw <global configuration>
      +--ro <global operational state>
      +--rw address-family* [address-family]
      | +--rw address-family      identityref
      | +--rw <per address family configuration>
      | +--ro <per address family operational state>
      +--rw interfaces
        +--rw interface* [name]
          +--rw name              if:interface-ref
          +--rw address-family* [address-family]
            +--rw address-family  identityref
            +--rw <per interface configuration>
            +--ro <per interface operational state>
          +--ro neighbors
            +--ro ipv4-neighbor* [address]
            | +--ro address        inet:ipv4-address
            | +--ro <IPv4 per neighbor operational state>
            +--ro ipv6-neighbor* [address]
            | +--ro address        inet:ipv6-address
            | +--ro <IPv6 per neighbor operational state>
```

The presence of the top-level container "pim" enables the PIM protocols.

### 3.1.2. Global Data

The global configuration data and operational state data cover support for graceful restart in the PIM base model. Additional features can be added by augmentation if required by an implementation.

### 3.1.3. Per Address Family Data

Support for per address family data is shown below:

```

+--rw pim!
  +--rw address-family* [address-family]
    | +--rw address-family      identityref
    | +--rw graceful-restart
    |   ...
    | +--ro statistics
    | | +--ro discontinuity-time? yang:date-and-time
    | | +--ro error
    | | | +--ro assert?          yang:counter32
    | |   ...
    | | +--ro queue
    | | | +--ro size?           uint32
    | | | +--ro overflow?      yang:counter32
    | | +--ro received
    | | | +--ro assert?        yang:counter32
    | |   ...
    | | +--ro sent
    | | | +--ro assert?        yang:counter32
    | |   ...
    | +--ro topology-tree-info
    | | +--ro ipv4-route* [group source-address is-rpt]
    | | | +--ro group
    | | | | rt-types:ipv4-multicast-group-address
    | | | +--ro source-address
    | | | | rt-types:ipv4-multicast-source-address
    | | | +--ro is-rpt          boolean
    | | +--ro ipv6-route* [group source-address is-rpt]
    | | | +--ro group
    | | | | rt-types:ipv6-multicast-group-address
    | | | +--ro source-address
    | | | | rt-types:ipv6-multicast-source-address
    | | | +--ro is-rpt          boolean
    | |   ...
    | | +--ro incoming-interface? if:interface-ref
    | |   ...
    | | +--ro outgoing-interface* [name]
    | | | +--ro name            if:interface-ref
    | | | +--ro expiration?    rt-types:timer-value-seconds16
    | | | +--ro up-time?       rt-types:timeticks64
    | | | +--ro jp-state?      enumeration

```

This is the location that most of the PIM RP module (ietf-pim-rp) augments. Each of the mode-specific modules also augments this schema tree.

### 3.1.4. PIM Interface Modeling

The configuration data and operational state data of PIM interfaces are modeled as shown below:

```

+--rw pim!
  +--rw interfaces
    +--rw interface* [name]
      +--rw name          if:interface-ref
      +--rw address-family* [address-family]
        +--rw address-family  identityref
        +--rw bfd {bfd}?
          ...
        +--rw dr-priority?      uint32 {intf-dr-priority}?
        +--rw hello-interval?   rt-types:timer-value-seconds16
          | {intf-hello-interval}?
        +--rw (hello-holdtime-or-multiplier)?
          | +--:(holdtime) {intf-hello-holdtime}?
          | | +--rw hello-holdtime?
          | |   rt-types:timer-value-seconds16
          | +--:(multiplier) {intf-hello-multiplier}?
          |   +--rw hello-multiplier?
          |     rt-types:timer-multiplier
        +--rw jp-interval?      rt-types:timer-value-seconds16
          | {intf-jp-interval}?
        +--rw (jp-holdtime-or-multiplier)?
          | +--:(holdtime) {intf-jp-holdtime}?
          | | +--rw jp-holdtime?
          | |   rt-types:timer-value-seconds16
          | +--:(multiplier) {intf-jp-multiplier}?
          |   +--rw jp-multiplier?
          |     rt-types:timer-multiplier
        +--rw override-interval? uint16
          | {intf-override-interval}?
        +--rw propagation-delay? uint16
          | {intf-propagation-delay}?
        +--ro oper-status?       enumeration
        +--ro gen-id?           uint32
        +--ro hello-expiration? rt-types:timer-value-seconds16
        +--ro ipv4
          | +--ro address*      inet:ipv4-address
          | +--ro dr-address?   inet:ipv4-address
        +--ro ipv6
          | +--ro address*      inet:ipv6-address
          | +--ro dr-address?   inet:ipv6-address

```

Support for BFD is achieved by using a grouping provided by an external module, "ietf-bfd-types", as defined in [\[RFC9127\]](#).

### 3.1.5. Neighbor Modeling

For each PIM interface, there can be a list of neighbors that contains operational state data for each neighbor. To model such data, the following structure is specified:

```

+--rw pim!
  +--rw interfaces
    +--rw interface* [name]
      +--rw address-family* [address-family]
        +--ro neighbors
          +--ro ipv4-neighbor* [address]
            +--ro address          inet:ipv4-address
            +--ro bfd-status?      enumeration
            +--ro expiration?
            |   rt-types:timer-value-seconds16
            +--ro dr-priority?     uint32
            +--ro gen-id?          uint32
            +--ro lan-prune-delay
            | +--ro present?        boolean
            | +--ro override-interval? uint16
            | +--ro propagation-delay? uint16
            | +--ro t-bit?          boolean
            +--ro up-time?          rt-types:timeticks64
          +--ro ipv6-neighbor* [address]
            +--ro address          inet:ipv6-address
            +--ro bfd-status?      enumeration
            +--ro expiration?
            |   rt-types:timer-value-seconds16
            +--ro dr-priority?     uint32
            +--ro gen-id?          uint32
            +--ro lan-prune-delay
            | +--ro present?        boolean
            | +--ro override-interval? uint16
            | +--ro propagation-delay? uint16
            | +--ro t-bit?          boolean
            +--ro up-time?          rt-types:timeticks64

```

### 3.1.6. Notifications

The PIM base module also defines the notifications for PIM interface and neighbor events, as shown below:

```

notifications:
  +---n pim-neighbor-event
  |   +--ro event-type?          neighbor-event-type
  |   +--ro interface-ref?      leafref
  |   +--ro interface-af-ref?   leafref
  |   +--ro neighbor-ipv4-ref?  leafref
  |   +--ro neighbor-ipv6-ref?  leafref
  |   +--ro up-time?            rt-types:timeticks64
  +---n pim-interface-event
  |   +--ro event-type?         interface-event-type
  |   +--ro interface-ref?     leafref
  |   +--ro ipv4
  |     | +--ro address*        inet:ipv4-address
  |     | +--ro dr-address?     inet:ipv4-address
  |   +--ro ipv6
  |     | +--ro address*        inet:ipv6-address
  |     | +--ro dr-address?     inet:ipv6-address

```

## 3.2. PIM RP Module

The PIM RP module augments the PIM base module to define the configuration and operational state information scoped to RP-related features:

```

module: ietf-pim-rp
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
  /pim-base:address-family:
    +--rw rp
    |   +--rw static-rp
    |   |   ...
    |   +--rw bsr {bsr}?
    |   |   ...
    |   +--ro rp-list
    |   |   ...
    |   +--ro rp-mappings
    |   |   ...

```

This module is shared by PIM-SM and PIM-BIDIR mode but is not shared by PIM-DM. The PIM-SM module and the PIM-BIDIR module augment this module to cover mode-specific data.

The following sections describe the features and capabilities covered in this module.

### 3.2.1. Static RPs

Static RPs can be configured by using the following portion of the module:

```
+--rw rp
  +--rw static-rp
    | +--rw ipv4-rp* [rp-address]
    | | +--rw rp-address inet:ipv4-address
    | +--rw ipv6-rp* [rp-address]
    | +--rw rp-address inet:ipv6-address
```

### 3.2.2. BSRs

Support for BSRs includes both configuration data and operational state data, as shown below:

```

+--rw rp
  +--rw bsr {bsr}?
    +--rw bsr-candidate!
      | +--rw (interface-or-address)?
      | | +--:(interface) {candidate-interface}?
      | | | +--rw interface if:interface-ref
      | | | +--:(ipv4-address) {candidate-ipv4}?
      | | | | +--rw ipv4-address inet:ipv4-address
      | | | | +--:(ipv6-address) {candidate-ipv6}?
      | | | | +--rw ipv6-address inet:ipv6-address
      | | +--rw hash-mask-length uint8
      | | +--rw priority? uint8
      +--rw rp-candidate
        | +--rw interface* [name] {candidate-interface}?
        | | +--rw name if:interface-ref
        | | +--rw policy-name? string
        | | +--rw mode? identityref
        | +--rw ipv4-address* [address] {candidate-ipv4}?
        | | +--rw address inet:ipv4-address
        | | +--rw policy-name? string
        | | +--rw mode? identityref
        | +--rw ipv6-address* [address] {candidate-ipv6}?
        | | +--rw address inet:ipv6-address
        | | +--rw policy-name? string
        | | +--rw mode? identityref
      +--ro bsr
        | +--ro address? inet:ip-address
        | +--ro hash-mask-length? uint8
        | +--ro priority? uint8
        | +--ro up-time? rt-types:timeticks64
      +--ro (election-state)? {bsr-election-state}?
        | +--:(candidate)
        | | +--ro candidate-bsr-state? enumeration
        | +--:(non-candidate)
        | | +--ro non-candidate-bsr-state? enumeration
      +--ro bsr-next-bootstrap? uint16
    +--ro rp
      | +--ro rp-address? inet:ip-address
      | +--ro policy-name? string
      | +--ro up-time? rt-types:timeticks64
    +--ro rp-candidate-next-advertisement? uint16

```

### 3.2.3. RP State Data

This portion of the model provides the operational state information for all RPs on the router, including the statically configured RPs and the BSR elected RPs.

```

+--rw rp
  +--ro rp-list
    | +--ro ipv4-rp* [rp-address mode]
    | | +--ro rp-address          inet:ipv4-address
    | | +--ro mode                 identityref
    | | +--ro info-source-address? inet:ipv4-address
    | | +--ro info-source-type?    identityref
    | | +--ro up-time?             rt-types:timeticks64
    | | +--ro expiration?         rt-types:timer-value-seconds16
    | +--ro ipv6-rp* [rp-address mode]
    | +--ro rp-address            inet:ipv6-address
    | +--ro mode                   identityref
    | +--ro info-source-address?  inet:ipv6-address
    | +--ro info-source-type?     identityref
    | +--ro up-time?              rt-types:timeticks64
    | +--ro expiration?           rt-types:timer-value-seconds16

```

### 3.2.4. RP-to-Group Mappings

The operational state data of the mappings between RPs and multicast groups is modeled as follows:

```

+--rw rp
  +--ro rp-mappings
    +--ro ipv4-rp* [group rp-address]
    | +--ro group          inet:ipv4-prefix
    | +--ro rp-address     inet:ipv4-address
    | +--ro up-time?       rt-types:timeticks64
    | +--ro expiration?   rt-types:timer-value-seconds16
    +--ro ipv6-rp* [group rp-address]
    +--ro group            inet:ipv6-prefix
    +--ro rp-address       inet:ipv6-address
    +--ro up-time?         rt-types:timeticks64
    +--ro expiration?     rt-types:timer-value-seconds16

```



### 3.2.5. Notifications

The PIM RP module also defines the notifications for RP-related events, as shown below:

```

notifications:
  +---n pim-rp-event
    +--ro event-type?      rp-event-type
    +--ro instance-af-ref? leafref
    +--ro group?          rt-types:ip-multicast-group-address
    +--ro rp-address?     inet:ip-address
    +--ro is-rpt?        boolean
    +--ro mode?          pim-base:pim-mode
    +--ro message-origin? inet:ip-address

```

### 3.3. PIM-SM Module

The PIM-SM module covers Sparse Mode modeling, including PIM Any-Source Multicast (PIM-ASM) and PIM Source-Specific Multicast (PIM-SSM). This module has dependencies on the PIM base module and the PIM RP module, both of which are augmented by this module.

The augmentation to the "address-family" branch of the PIM base module is shown below:

```

module: ietf-pim-sm
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
  /pim-base:address-family:
    +--rw sm
      +--rw asm
        | +--rw anycast-rp!
        | | +--rw ipv4-anycast-rp* [anycast-address rp-address]
        | | | +--rw anycast-address inet:ipv4-address
        | | | +--rw rp-address      inet:ipv4-address
        | | +--rw ipv6-anycast-rp* [anycast-address rp-address]
        | | | +--rw anycast-address inet:ipv6-address
        | | | +--rw rp-address      inet:ipv6-address
        | +--rw spt-switch
        | | +--rw infinity! {spt-switch-infinity}?
        | | +--rw policy-name? string {spt-switch-policy}?
    +--rw ssm!
      +--rw range-policy? string

```

To support PIM-SM on an interface, this module augments the "interface" branch of the PIM base module, as follows:

```

module: ietf-pim-sm
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
  /pim-base:interfaces/pim-base:interface/pim-base:address-family:
    +--rw sm!
      +--rw passive? empty

```

This module also augments the PIM RP module to allow an RP to be configured in PIM-SM:

```
module: ietf-pim-sm
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
  /pim-base:address-family/pim-rp:rp/pim-rp:static-rp/pim-rp:ipv4-rp:
    +--rw sm!
      +--rw policy-name?   string
      +--rw override?     boolean {static-rp-override}?
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
  /pim-base:address-family/pim-rp:rp/pim-rp:static-rp/pim-rp:ipv6-rp:
    +--rw sm!
      +--rw policy-name?   string
      +--rw override?     boolean {static-rp-override}?
```

### 3.4. PIM-DM Module

The PIM-DM module covers Dense Mode modeling. This module augments the PIM base module, but it has no dependency on the PIM RP module.

```
module: ietf-pim-dm
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
  /pim-base:address-family:
    +--rw dm!
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
  /pim-base:interfaces/pim-base:interface
  /pim-base:address-family:
    +--rw dm!
```

### 3.5. PIM-BIDIR Module

The PIM-BIDIR module covers Bidirectional PIM modeling. Like PIM-SM, this module augments both the PIM base module and the PIM RP module.

The augmentations to the PIM base module, on the "address-family", "interface", and "neighbor" branches, are as follows:

```
module: ietf-pim-bidir
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
/pim-base:address-family:
  +--rw bidir!

  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
/pim-base:interfaces/pim-base:interface/pim-base:address-family:
  +--rw bidir!
  +--rw df-election {intf-df-election}?
    +--rw offer-interval?      uint16
    +--rw backoff-interval?    uint16
    +--rw offer-multiplier?    uint8

  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
/pim-base:interfaces/pim-base:interface/pim-base:address-family
/pim-base:neighbors/pim-base:ipv4-neighbor:
  +--ro bidir-capable?  boolean

  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
/pim-base:interfaces/pim-base:interface/pim-base:address-family
/pim-base:neighbors/pim-base:ipv6-neighbor:
  +--ro bidir-capable?  boolean
```

This module also augments the PIM RP module to extend the capabilities of RPs for PIM-BIDIR mode:

```

module: ietf-pim-bidir
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
  /pim-base:address-family/pim-rp:rp/pim-rp:static-rp/pim-rp:ipv4-rp:
    +--rw bidir!
      +--rw policy-name?   string
      +--rw override?     boolean {static-rp-override}?

  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
  /pim-base:address-family/pim-rp:rp/pim-rp:static-rp/pim-rp:ipv6-rp:
    +--rw bidir!
      +--rw policy-name?   string
      +--rw override?     boolean {static-rp-override}?

  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
  /pim-base:address-family/pim-rp:rp:
    +--ro bidir
      +--ro df-election
      | +--ro ipv4-rp* [rp-address]
      | | +--ro rp-address   inet:ipv4-address
      | +--ro ipv6-rp* [rp-address]
      | | +--ro rp-address   inet:ipv6-address
      +--ro interface-df-election
      +--ro ipv4-rp* [rp-address interface-name]
      | +--ro rp-address           inet:ipv4-address
      | +--ro interface-name       if:interface-ref
      | +--ro df-address?         inet:ipv4-address
      | +--ro interface-state?    identityref
      | +--ro up-time?            rt-types:timeticks64
      | +--ro winner-metric?      uint32
      | +--ro winner-metric-preference? uint32
      +--ro ipv6-rp* [rp-address interface-name]
      +--ro rp-address           inet:ipv6-address
      +--ro interface-name       if:interface-ref
      +--ro df-address?         inet:ipv6-address
      +--ro interface-state?    identityref
      +--ro up-time?            rt-types:timeticks64
      +--ro winner-metric?      uint32
      +--ro winner-metric-preference? uint32

```

## 4. Complete Tree Structure

## 4.1. PIM Base Module

```

module: ietf-pim-base
  augment /rt:routing/rt:control-plane-protocols:
    +--rw pim!
      +--rw graceful-restart
        | +--rw enabled?    boolean
        | +--rw duration?  uint16
      +--rw address-family* [address-family]
        | +--rw address-family    identityref
        | +--rw graceful-restart
        | | +--rw enabled?    boolean
        | | +--rw duration?  uint16
      +--ro statistics
        | +--ro discontinuity-time?  yang:date-and-time
        | +--ro error
        | | +--ro assert?          yang:counter64
        | | +--ro bsr?             yang:counter64
        | | +--ro candidate-rp-advertisement? yang:counter64
        | | +--ro df-election?     yang:counter64
        | | +--ro graft?           yang:counter64
        | | +--ro graft-ack?       yang:counter64
        | | +--ro hello?           yang:counter64
        | | +--ro join-prune?      yang:counter64
        | | +--ro register?        yang:counter64
        | | +--ro register-stop?   yang:counter64
        | | +--ro state-refresh?   yang:counter64
        | | +--ro checksum?        yang:counter64
        | | +--ro format?          yang:counter64
        | +--ro queue
        | | +--ro size?            uint32
        | | +--ro overflow?        yang:counter32
        | +--ro received
        | | +--ro assert?          yang:counter64
        | | +--ro bsr?             yang:counter64
        | | +--ro candidate-rp-advertisement? yang:counter64
        | | +--ro df-election?     yang:counter64
        | | +--ro graft?           yang:counter64
        | | +--ro graft-ack?       yang:counter64
        | | +--ro hello?           yang:counter64
        | | +--ro join-prune?      yang:counter64
        | | +--ro register?        yang:counter64
        | | +--ro register-stop?   yang:counter64
        | | +--ro state-refresh?   yang:counter64
        | +--ro sent
        | | +--ro assert?          yang:counter64
        | | +--ro bsr?             yang:counter64
        | | +--ro candidate-rp-advertisement? yang:counter64
        | | +--ro df-election?     yang:counter64
        | | +--ro graft?           yang:counter64
        | | +--ro graft-ack?       yang:counter64
        | | +--ro hello?           yang:counter64
        | | +--ro join-prune?      yang:counter64
        | | +--ro register?        yang:counter64
        | | +--ro register-stop?   yang:counter64
        | | +--ro state-refresh?   yang:counter64
      +--ro topology-tree-info
        | +--ro ipv4-route* [group source-address is-rpt]
        | | +--ro group

```

```

|         rt-types:ipv4-multicast-group-address
+--ro source-address
|         rt-types:ipv4-multicast-source-address
+--ro is-rpt          boolean
+--ro expiration?
|         rt-types:timer-value-seconds16
+--ro incoming-interface?  if:interface-ref
+--ro is-spt?            boolean
+--ro mode?             identityref
+--ro msdp-learned?      boolean
+--ro rp-address?       inet:ip-address
+--ro rpf-neighbor?     inet:ip-address
+--ro up-time?         rt-types:timeticks64
+--ro outgoing-interface* [name]
|   +--ro name          if:interface-ref
|   +--ro expiration?  rt-types:timer-value-seconds16
|   +--ro up-time?    rt-types:timeticks64
|   +--ro jp-state?   enumeration
+--ro ipv6-route* [group source-address is-rpt]
+--ro group
|   rt-types:ipv6-multicast-group-address
+--ro source-address
|   rt-types:ipv6-multicast-source-address
+--ro is-rpt          boolean
+--ro expiration?
|   rt-types:timer-value-seconds16
+--ro incoming-interface?  if:interface-ref
+--ro is-spt?            boolean
+--ro mode?             identityref
+--ro msdp-learned?      boolean
+--ro rp-address?       inet:ip-address
+--ro rpf-neighbor?     inet:ip-address
+--ro up-time?         rt-types:timeticks64
+--ro outgoing-interface* [name]
|   +--ro name          if:interface-ref
|   +--ro expiration?  rt-types:timer-value-seconds16
|   +--ro up-time?    rt-types:timeticks64
|   +--ro jp-state?   enumeration
+--rw interfaces
+--rw interface* [name]
+--rw name          if:interface-ref
+--rw address-family* [address-family]
+--rw address-family  identityref
+--rw bfd {bfd}?
| +--rw enable?          boolean
| +--rw local-multiplier? multiplier
| +--rw (interval-config-type)?
| | +--:(tx-rx-intervals)
| | | +--rw desired-min-tx-interval  uint32
| | | +--rw required-min-rx-interval  uint32
| | +--:(single-interval)
| | +--rw min-interval              uint32
+--rw dr-priority?          uint32
| {intf-dr-priority}?
+--rw hello-interval?
|   rt-types:timer-value-seconds16
|   {intf-hello-interval}?
+--rw (hello-holdtime-or-multiplier)?

```



```

| +--:(holdtime) {intf-hello-holdtime}?
| | +--rw hello-holdtime?
| | | rt-types:timer-value-seconds16
| +--:(multiplier) {intf-hello-multiplier}?
| | +--rw hello-multiplier?
| | | rt-types:timer-multiplier
+--rw jp-interval?
| | rt-types:timer-value-seconds16
| | {intf-jp-interval}?
+--rw (jp-holdtime-or-multiplier)?
| +--:(holdtime) {intf-jp-holdtime}?
| | +--rw jp-holdtime?
| | | rt-types:timer-value-seconds16
| +--:(multiplier) {intf-jp-multiplier}?
| | +--rw jp-multiplier?
| | | rt-types:timer-multiplier
+--rw override-interval? uint16
| | {intf-override-interval}?
+--rw propagation-delay? uint16
| | {intf-propagation-delay}?
+--ro oper-status? enumeration
+--ro gen-id? uint32
+--ro hello-expiration?
| | rt-types:timer-value-seconds16
+--ro ipv4
| +--ro address* inet:ipv4-address
| +--ro dr-address? inet:ipv4-address
+--ro ipv6
| +--ro address* inet:ipv6-address
| +--ro dr-address? inet:ipv6-address
+--ro neighbors
+--ro ipv4-neighbor* [address]
| +--ro address inet:ipv4-address
| +--ro bfd-state? bfd-types:state
| +--ro expiration?
| | rt-types:timer-value-seconds16
| +--ro dr-priority? uint32
| +--ro gen-id? uint32
| +--ro lan-prune-delay
| | +--ro present? boolean
| | +--ro override-interval? uint16
| | +--ro propagation-delay? uint16
| | +--ro t-bit? boolean
| +--ro up-time? rt-types:timeticks64
+--ro ipv6-neighbor* [address]
+--ro address inet:ipv6-address
+--ro bfd-state? bfd-types:state
+--ro expiration?
| | rt-types:timer-value-seconds16
+--ro dr-priority? uint32
+--ro gen-id? uint32
+--ro lan-prune-delay
| +--ro present? boolean
| +--ro override-interval? uint16
| +--ro propagation-delay? uint16
| +--ro t-bit? boolean
+--ro up-time? rt-types:timeticks64

```

```
notifications:
  +---n pim-neighbor-event
  | +--ro event-type?      neighbor-event-type
  | +--ro interface-ref?   leafref
  | +--ro interface-af-ref? leafref
  | +--ro neighbor-ipv4-ref? leafref
  | +--ro neighbor-ipv6-ref? leafref
  | +--ro up-time?         rt-types:timeticks64
  +---n pim-interface-event
  +--ro event-type?      interface-event-type
  +--ro interface-ref?   leafref
  +--ro ipv4
  | +--ro address*        inet:ipv4-address
  | +--ro dr-address?     inet:ipv4-address
  +--ro ipv6
  +--ro address*         inet:ipv6-address
  +--ro dr-address?     inet:ipv6-address
```

## 4.2. PIM RP Module

```

module: ietf-pim-rp
augment /rt:routing/rt:control-plane-protocols/pim-base:pim
/pim-base:address-family:
+--rw rp
+--rw static-rp
| +--rw ipv4-rp* [rp-address]
| | +--rw rp-address inet:ipv4-address
| +--rw ipv6-rp* [rp-address]
| | +--rw rp-address inet:ipv6-address
+--rw bsr {bsr}?
| +--rw bsr-candidate!
| | +--rw (interface-or-address)?
| | | +--:(interface) {candidate-interface}?
| | | | +--rw interface if:interface-ref
| | | +--:(ipv4-address) {candidate-ipv4}?
| | | | +--rw ipv4-address inet:ipv4-address
| | | +--:(ipv6-address) {candidate-ipv6}?
| | | | +--rw ipv6-address inet:ipv6-address
| | +--rw hash-mask-length uint8
| | +--rw priority? uint8
+--rw rp-candidate
| +--rw interface* [name] {candidate-interface}?
| | +--rw name if:interface-ref
| | +--rw policy-name? string
| | +--rw mode? identityref
| +--rw ipv4-address* [address] {candidate-ipv4}?
| | +--rw address inet:ipv4-address
| | +--rw policy-name? string
| | +--rw mode? identityref
| +--rw ipv6-address* [address] {candidate-ipv6}?
| | +--rw address inet:ipv6-address
| | +--rw policy-name? string
| | +--rw mode? identityref
+--ro bsr
| +--ro address? inet:ip-address
| +--ro hash-mask-length? uint8
| +--ro priority? uint8
| +--ro up-time? rt-types:timeticks64
+--ro (election-state)? {bsr-election-state}?
| +--:(candidate)
| | +--ro candidate-bsr-state? enumeration
| +--:(non-candidate)
| | +--ro non-candidate-bsr-state? enumeration
+--ro bsr-next-bootstrap? uint16
+--ro rp
| +--ro rp-address? inet:ip-address
| +--ro policy-name? string
| +--ro up-time? rt-types:timeticks64
+--ro rp-candidate-next-advertisement? uint16
+--ro rp-list
| +--ro ipv4-rp* [rp-address mode]
| | +--ro rp-address inet:ipv4-address
| | +--ro mode identityref
| | +--ro info-source-address? inet:ipv4-address
| | +--ro info-source-type? identityref
| | +--ro up-time? rt-types:timeticks64
| | +--ro expiration?

```

```

| |         rt-types:timer-value-seconds16
| +--ro ipv6-rp* [rp-address mode]
|   +--ro rp-address          inet:ipv6-address
|   +--ro mode                identityref
|   +--ro info-source-address? inet:ipv6-address
|   +--ro info-source-type?   identityref
|   +--ro up-time?            rt-types:timeticks64
|   +--ro expiration?
|                               rt-types:timer-value-seconds16
+--ro rp-mappings
  +--ro ipv4-rp* [group-range rp-address]
  | +--ro group-range         inet:ipv4-prefix
  | +--ro rp-address          inet:ipv4-address
  | +--ro up-time?            rt-types:timeticks64
  | +--ro expiration?        rt-types:timer-value-seconds16
  +--ro ipv6-rp* [group-range rp-address]
  | +--ro group-range         inet:ipv6-prefix
  | +--ro rp-address          inet:ipv6-address
  | +--ro up-time?            rt-types:timeticks64
  | +--ro expiration?        rt-types:timer-value-seconds16

notifications:
+---n pim-rp-event
  +--ro event-type?           rp-event-type
  +--ro instance-af-ref?     leafref
  +--ro group?                rt-types:ip-multicast-group-address
  +--ro rp-address?           inet:ip-address
  +--ro is-rpt?               boolean
  +--ro mode?                 identityref
  +--ro message-origin?      inet:ip-address

```

### 4.3. PIM-SM Module

```

module: ietf-pim-sm
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
    /pim-base:address-family:
      +--rw sm
        +--rw asm
          | +--rw anycast-rp!
          | | +--rw ipv4-anycast-rp* [anycast-address rp-address]
          | | | +--rw anycast-address inet:ipv4-address
          | | | +--rw rp-address inet:ipv4-address
          | | +--rw ipv6-anycast-rp* [anycast-address rp-address]
          | | | +--rw anycast-address inet:ipv6-address
          | | | +--rw rp-address inet:ipv6-address
          | +--rw spt-switch
          | | +--rw infinity! {spt-switch-infinity}?
          | | +--rw policy-name? string {spt-switch-policy}?
        +--rw ssm!
          +--rw range-policy? string
        augment /rt:routing/rt:control-plane-protocols/pim-base:pim
          /pim-base:interfaces/pim-base:interface
            /pim-base:address-family:
              +--rw sm!
                +--rw passive? empty
            augment /rt:routing/rt:control-plane-protocols/pim-base:pim
              /pim-base:address-family/pim-rp:rp/pim-rp:static-rp
                /pim-rp:ipv4-rp:
                  +--rw sm!
                    +--rw policy-name? string
                    +--rw override? boolean {static-rp-override}?
            augment /rt:routing/rt:control-plane-protocols/pim-base:pim
              /pim-base:address-family/pim-rp:rp/pim-rp:static-rp
                /pim-rp:ipv6-rp:
                  +--rw sm!
                    +--rw policy-name? string
                    +--rw override? boolean {static-rp-override}?

```

### 4.4. PIM-DM Module

```

module: ietf-pim-dm
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
    /pim-base:address-family:
      +--rw dm!
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
    /pim-base:interfaces/pim-base:interface
      /pim-base:address-family:
        +--rw dm!

```

## 4.5. PIM-BIDIR Module

```

module: ietf-pim-bidir
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
    /pim-base:address-family:
      +--rw bidir!
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
    /pim-base:interfaces/pim-base:interface
      /pim-base:address-family:
        +--rw bidir!
          +--rw df-election {intf-df-election}?
            +--rw offer-interval?      uint16
            +--rw backoff-interval?    uint16
            +--rw offer-multiplier?    uint8
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
    /pim-base:address-family/pim-rp:rp/pim-rp:static-rp
      /pim-rp:ipv4-rp:
        +--rw bidir!
          +--rw policy-name?  string
          +--rw override?     boolean {static-rp-override}?
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
    /pim-base:address-family/pim-rp:rp/pim-rp:static-rp
      /pim-rp:ipv6-rp:
        +--rw bidir!
          +--rw policy-name?  string
          +--rw override?     boolean {static-rp-override}?
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
    /pim-base:address-family/pim-rp:rp:
      +--ro bidir
        +--ro df-election
          | +--ro ipv4-rp* [rp-address]
          | | +--ro rp-address      inet:ipv4-address
          | | +--ro rp-address      inet:ipv4-address
          | +--ro ipv6-rp* [rp-address]
          | | +--ro rp-address      inet:ipv6-address
          | | +--ro rp-address      inet:ipv6-address
        +--ro interface-df-election
          +--ro ipv4-rp* [rp-address interface-name]
            | +--ro rp-address      inet:ipv4-address
            | +--ro interface-name  if:interface-ref
            | +--ro df-address?     inet:ipv4-address
            | +--ro interface-state? identityref
            | +--ro up-time?        rt-types:timeticks64
            | +--ro winner-metric?  uint32
            | +--ro winner-metric-preference? uint32
          +--ro ipv6-rp* [rp-address interface-name]
            +--ro rp-address      inet:ipv6-address
            +--ro interface-name  if:interface-ref
            +--ro df-address?     inet:ipv6-address
            +--ro interface-state? identityref
            +--ro up-time?        rt-types:timeticks64
            +--ro winner-metric?  uint32
            +--ro winner-metric-preference? uint32
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
    /pim-base:interfaces/pim-base:interface
      /pim-base:address-family/pim-base:neighbors
        /pim-base:ipv4-neighbor:
          +--ro bidir-capable?  boolean
  augment /rt:routing/rt:control-plane-protocols/pim-base:pim
    /pim-base:interfaces/pim-base:interface
      /pim-base:address-family/pim-base:neighbors

```



```

    /pim-base:ipv6-neighbor:
    +--ro bidir-capable?  boolean

```

## 5. Relationship to the PIM-STD-MIB

The following sections describe the mappings between the objects in the PIM-STD-MIB defined in [RFC5060] and the YANG data nodes defined in this document.

### 5.1. pimInterfaceTable

pimInterfaceTable is mapped to pim/interfaces/interface. The key of pimInterfaceTable is pimInterfaceIfIndex and pimInterfaceIPVersion, while the key of the "interface" list in YANG is the node "name". For each value of pimInterfaceIPVersion, the "interface" list contains a corresponding sublist whose key is the node "address-family".

Table 2 lists the YANG data nodes with corresponding objects of pimInterfaceTable in the PIM-STD-MIB.

YANG Node	PIM-STD-MIB Object
address-family	pimInterfaceAddressType
ipv4/address	pimInterfaceAddress
ipv6/address	
gen-id	pimInterfaceGenerationIDValue
ipv4/dr-address	pimInterfaceDR
ipv6/dr-address	
dr-priority	pimInterfaceDRPriority
hello-interval	pimInterfaceHelloInterval
hello-holdtime	pimInterfaceHelloHoldtime
jp-interval	pimInterfaceJoinPruneInterval
jp-holdtime	pimInterfaceJoinPruneHoldtime
bidir/offer-multiplier	pimInterfaceDFElectionRobustness
propagation-delay	pimInterfacePropagationDelay
override-interval	pimInterfaceOverrideInterval

Table 2: YANG Nodes and pimInterfaceTable Objects

## 5.2. pimNeighborTable

pimNeighborTable is mapped to pim/interfaces/interface/neighbors/ipv4-neighbor and pim/interfaces/interface/neighbors/ipv6-neighbor.

Table 3 lists the YANG data nodes with corresponding objects of pimNeighborTable in the PIM-STD-MIB.

YANG Node	PIM-STD-MIB Object
ipv4-neighbor	pimNeighborAddressType
ipv6-neighbor	
address	pimNeighborAddress
gen-id	pimNeighborGenerationIDValue
up-time	pimNeighborUpTime
expiration	pimNeighborExpiryTime
dr-priority	pimNeighborDRPriority
lan-prune-delay/present	pimNeighborLanPruneDelayPresent
lan-prune-delay/t-bit	pimNeighborTBit
lan-prune-delay/propagation-delay	pimNeighborPropagationDelay
lan-prune-delay/override-interval	pimNeighborOverrideInterval
ietf-pim-bidir:bidir-capable	pimNeighborBidirCapable

Table 3: YANG Nodes and pimNeighborTable Objects

## 5.3. pimStarGTable

pimStarGTable is mapped to pim/address-family/topology-tree-info/ipv4-route and pim/address-family/topology-tree-info/ipv6-route, when the value of the "source-address" leaf is "ietf-routing-types:\*" and the value of the "is-rpt" leaf is "false".

Table 4 lists the YANG data nodes with corresponding objects of pimStarGTable in the PIM-STD-MIB.

YANG Node	PIM-STD-MIB Object
ipv4-route	pimStarGAddressType

YANG Node	PIM-STD-MIB Object
ipv6-route	
group	pimStarGGrpAddress
up-time	pimStarGUpTime
mode	pimStarGPimMode
rp-address	pimStarGRPAddressType
rp-address	pimStarGRPAddress
rpf-neighbor	pimStarGUpstreamNeighborType
rpf-neighbor	pimStarGUpstreamNeighbor
incoming-interface	pimStarGRPFIfIndex

Table 4: YANG Nodes and pimStarGTable Objects

In addition, the object "pimStarGPimModeOrigin" in pimStarGTable is mapped to the node "rp/rp-list/ipv4-rp/info-source-type" or the node "rp/rp-list/ipv6-rp/info-source-type" in the YANG module "ietf-pim-rp".

#### 5.4. pimSGTable

pimSGTable is mapped to pim/address-family/topology-tree-info/ipv4-route and pim/address-family/topology-tree-info/ipv6-route, when the value of the "source-address" leaf is not "ietf-routing-types:\*" and the value of the "is-rpt" leaf is "false".

Table 5 lists the YANG data nodes with corresponding objects of pimSGTable in the PIM-STD-MIB.

YANG Node	PIM-STD-MIB Object
ipv4-route	pimSGAddressType
ipv6-route	
group	pimSGGrpAddress
source-address	pimSGSrcAddress
up-time	pimSGUpTime
mode	pimSGPimMode
rpf-neighbor	pimStarGUpstreamNeighbor

YANG Node	PIM-STD-MIB Object
incoming-interface	pimStarGRPFifIndex
is-spt	pimSGSPTBit
expiration	pimSGKeepaliveTimer

Table 5: YANG Nodes and pimSGTable Objects

## 5.5. pimSGRptTable

pimSGRptTable is mapped to pim/address-family/topology-tree-info/ipv4-route and pim/address-family/topology-tree-info/ipv6-route, when the value of the "is-rpt" leaf is "true".

Table 6 lists the YANG data nodes with corresponding objects of pimSGRptTable in the PIM-STD-MIB.

YANG Node	PIM-STD-MIB Object
ipv4-route	pimStarGAddressType
ipv6-route	
group	pimStarGGrpAddress
source-address	pimSGRptSrcAddress
up-time	pimSGRptUpTime

Table 6: YANG Nodes and pimSGRptTable Objects

## 5.6. pimBidirDFElectionTable

pimBidirDFElectionTable is mapped to pim/address-family/rp/bidir/interface-df-election/ipv4-rp and pim/address-family/rp/bidir/interface-df-election/ipv6-rp. The key of pimBidirDFElectionTable includes pimBidirDFElectionIfIndex, whose type is InterfaceIndex, while the YANG lists use a node "name" with the type string instead.

Table 7 lists the YANG data nodes with corresponding objects of pimBidirDFElectionTable in the PIM-STD-MIB.

YANG Node	PIM-STD-MIB Object
ipv4-rp	pimBidirDFElectionAddressType
ipv6-rp	

YANG Node	PIM-STD-MIB Object
rp-address	pimBidirDFElectionRPAddress
df-address	pimBidirDFElectionWinnerAddressType
	pimBidirDFElectionWinnerAddress
up-time	pimBidirDFElectionWinnerUpTime
winner-metric-preference	pimBidirDFElectionWinnerMetricPref
winner-metric-preference	pimBidirDFElectionWinnerMetric
interface-state	pimBidirDFElectionState

Table 7: YANG Nodes and pimBidirDFElectionTable Objects

### 5.7. pimStaticRPTable

pimStaticRPTable is mapped to pim/address-family/rp/static-rp/ipv4-rp and pim/address-family/rp/static-rp/ipv6-rp.

Table 8 lists the YANG data nodes with corresponding objects of pimStaticRPTable in the PIM-STD-MIB.

YANG Node	PIM-STD-MIB Object
ipv4-rp	pimStaticRPAddressType
ipv6-rp	
rp-address	pimStaticRPRPAddress
bidir	pimStaticRPPimMode
sm	
bidir/override	pimStaticRPOverrideDynamic
sm/override	

Table 8: YANG Nodes and pimStaticRPTable Objects

### 5.8. pimAnycastRPSetTable

pimAnycastRPSetTable is mapped to pim/address-family/sm/asm/anycast-rp/ipv4-anycast-rp and pim/address-family/sm/asm/anycast-rp/ipv6-anycast-rp.

Table 9 lists the YANG data nodes with corresponding objects of pimAnycastRPSetTable in the PIM-STD-MIB.

YANG Node	PIM-STD-MIB Object
ipv4-anycast-rp	pimAnycastRPSetAddressType
ipv6-anycast-rp	
anycast-address	pimAnycastRPSetAnycastAddress
rp-address	pimAnycastRPSetRouterAddress

Table 9: YANG Nodes and pimAnycastRPSetTable Objects

### 5.9. pimGroupMappingTable

pimGroupMappingTable is mapped to pim/address-family/rp/rp-mappings/ipv4-rp and pim/address-family/rp/rp-mappings/ipv6-rp.

Table 10 lists the YANG data nodes with corresponding objects of pimGroupMappingTable in the PIM-STD-MIB.

YANG Node	PIM-STD-MIB Object
ipv4-rp	pimGroupMappingAddressType
ipv6-rp	
group	pimGroupMappingGrpAddress
	pimGroupMappingGrpPrefixLength
ipv4-rp	pimGroupMappingRPAddressType
ipv6-rp	
rp-address	pimGroupMappingRPAddress
	pimGroupMappingPimMode

Table 10: YANG Nodes and pimGroupMappingTable Objects

In addition, the object "pimGroupMappingPimMode" in pimGroupMappingTable is mapped to the node "rp/rp-list/ipv4-rp/mode" or the node "rp/rp-list/ipv6-rp/mode" in the YANG module "ietf-pim-rp".

## 6. PIM YANG Modules

## 6.1. PIM Base Module

This module references [\[RFC3973\]](#), [\[RFC5015\]](#), [\[RFC5880\]](#), [\[RFC6991\]](#), [\[RFC7761\]](#), [\[RFC8294\]](#), [\[RFC8343\]](#), [\[RFC8349\]](#), [\[RFC8706\]](#), and [\[RFC9127\]](#).



```
<CODE BEGINS> file "ietf-pim-base@2021-09-03.yang"

module ietf-pim-base {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-pim-base";
  prefix pim-base;

  import ietf-inet-types {
    prefix "inet";
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-yang-types {
    prefix "yang";
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-routing-types {
    prefix "rt-types";
    reference
      "RFC 8294: Common YANG Data Types for the Routing Area";
  }

  import ietf-interfaces {
    prefix "if";
    reference
      "RFC 8343: A YANG Data Model for Interface Management";
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC 8349: A YANG Data Model for Routing Management (NMDA
      Version)";
  }

  import ietf-bfd-types {
    prefix "bfd-types";
    reference
      "RFC 9127: YANG Data Model for Bidirectional Forwarding
      Detection (BFD)";
  }

  organization
    "IETF PIM Working Group";

  contact
    "WG Web: <https://datatracker.ietf.org/wg/pim/>
    WG List: <mailto:pim@ietf.org>

    Editor: Xufeng Liu
           <mailto:xufeng.liu.ietf@gmail.com>

    Editor: Pete McAllister
           <mailto:pete.mcallister@metaswitch.com>
```

```
Editor: Anish Peter
       <mailto:anish.ietf@gmail.com>

Editor: Mahesh Sivakumar
       <mailto:sivakumar.mahesh@gmail.com>

Editor: Yisong Liu
       <mailto:liuyisong@huawei.com>

Editor: Fangwei Hu
       <mailto:hu.fangwei@zte.com.cn>;
```

description

"This module defines a collection of YANG definitions common for all PIM (Protocol Independent Multicast) modes.

Copyright (c) 2021 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>).

This version of this YANG module is part of RFC 9128; see the RFC itself for full legal notices.";

```
revision 2021-09-03 {
  description
    "Initial revision.";
  reference
    "RFC 9128: A YANG Data Model for Protocol Independent
     Multicast (PIM)";
}

/*
 * Features
 */
feature bfd {
  description
    "Supports BFD (Bidirectional Forwarding Detection).";
  reference
    "RFC 5880: Bidirectional Forwarding Detection (BFD)";
}

feature global-graceful-restart {
  description
    "Global configuration for graceful restart support as per
     RFC 8706.";
  reference
    "RFC 8706: Restart Signaling for IS-IS";
}

feature intf-dr-priority {
  description
```

```
    "Supports configuration of an interface DR (Designated Router)
    priority.";
  reference
    "RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.3.2";
}

feature intf-hello-holdtime {
  description
    "Supports configuration of the interface Hello Holdtime.";
  reference
    "RFC 3973: Protocol Independent Multicast - Dense Mode
    (PIM-DM): Protocol Specification (Revised), Section 4.3.3
    RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.11";
}

feature intf-hello-interval {
  description
    "Supports configuration of the interface Hello interval.";
  reference
    "RFC 3973: Protocol Independent Multicast - Dense Mode
    (PIM-DM): Protocol Specification (Revised), Section 4.8
    RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.11";
}

feature intf-hello-multiplier {
  description
    "Supports configuration of the interface Hello multiplier.";
  reference
    "RFC 3973: Protocol Independent Multicast - Dense Mode
    (PIM-DM): Protocol Specification (Revised), Section 4.8
    RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.11";
}

feature intf-jp-interval {
  description
    "Supports configuration of the interface Join/Prune interval.";
  reference
    "RFC 3973: Protocol Independent Multicast - Dense Mode
    (PIM-DM): Protocol Specification (Revised), Section 4.8
    RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.11";
}

feature intf-jp-holdtime {
  description
    "Supports configuration of the interface Join/Prune Holdtime.";
  reference
    "RFC 3973: Protocol Independent Multicast - Dense Mode
    (PIM-DM): Protocol Specification (Revised), Section 4.8
    RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.11";
}

feature intf-jp-multiplier {
```

```
description
  "Supports configuration of the interface Join/Prune
  multiplier.";
reference
  "RFC 3973: Protocol Independent Multicast - Dense Mode
  (PIM-DM): Protocol Specification (Revised), Section 4.8
  RFC 7761: Protocol Independent Multicast - Sparse Mode
  (PIM-SM): Protocol Specification (Revised), Section 4.11";
}

feature intf-propagation-delay {
  description
    "Supports configuration of interface propagation delay.";
  reference
    "RFC 3973: Protocol Independent Multicast - Dense Mode
    (PIM-DM): Protocol Specification (Revised), Section 4.3.5
    RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.3.3";
}

feature intf-override-interval {
  description
    "Supports configuration of the interface override interval.";
  reference
    "RFC 3973: Protocol Independent Multicast - Dense Mode
    (PIM-DM): Protocol Specification (Revised), Section 4.3.3
    RFC 5015: Bidirectional Protocol Independent Multicast
    (BIDIR-PIM), Section 3.6
    RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.11";
}

feature per-af-graceful-restart {
  description
    "Per address family configuration for graceful restart support
    as per RFC 8706.";
  reference
    "RFC 8706: Restart Signaling for IS-IS";
}

/*
 * Typedefs
 */
typedef interface-event-type {
  type enumeration {
    enum up {
      description
        "Neighbor status changed to 'up'.";
    }
    enum down {
      description
        "Neighbor status changed to 'down'.";
    }
    enum new-dr {
      description
        "A new DR (Designated Router) was elected on the connected
        network.";
    }
  }
}
```

```
    enum new-df {
      description
        "A new DF (Designated Forwarder) was elected on the
        connected network.";
    }
  }
  description
    "Operational status event type for notifications.";
}

typedef neighbor-event-type {
  type enumeration {
    enum up {
      description
        "Neighbor status changed to 'up'.";
    }
    enum down {
      description
        "Neighbor status changed to 'down'.";
    }
  }
  description
    "Operational status event type for notifications.";
}

/*
 * Identities
 */
identity pim-mode {
  description
    "The PIM mode in which a group is operating.";
}
identity pim-none {
  base pim-mode;
  description
    "PIM is not operating.";
}
identity pim-bidir {
  base pim-mode;
  description
    "PIM is operating in Bidirectional Mode.";
}
identity pim-dm {
  base pim-mode;
  description
    "PIM is operating in Dense Mode (DM).";
}
identity pim-sm {
  base pim-mode;
  description
    "PIM is operating in Sparse Mode (SM).";
}
identity pim-asm {
  base pim-sm;
  description
    "PIM is operating in Sparse Mode with Any-Source Multicast
    (ASM).";
}
```

```
identity pim-ssm {
  base pim-sm;
  description
    "PIM is operating in Sparse Mode with Source-Specific
    Multicast (SSM).";
}

/*
 * Groupings
 */
grouping graceful-restart-container {
  description
    "A grouping defining a container of graceful restart
    attributes.";
  container graceful-restart {
    leaf enabled {
      type boolean;
      default false;
      description
        "Enables or disables graceful restart.";
    }
    leaf duration {
      type uint16;
      units seconds;
      default 60;
      description
        "Maximum time for graceful restart to finish.";
    }
  }
  description
    "Container of graceful restart attributes.";
} // graceful-restart-container

grouping multicast-route-attributes {
  description
    "A grouping defining multicast route attributes.";

  leaf expiration {
    type rt-types:timer-value-seconds16;
    description
      "When the route will expire.";
  }
  leaf incoming-interface {
    type if:interface-ref;
    description
      "Reference to an entry in the global interface list.";
  }
  leaf is-spt {
    type boolean;
    description
      "'true' if the SPTbit (Shortest Path Tree bit) is set to
      indicate that forwarding is taking place on the
      (S,G) SPT.";
    reference
      "RFC 7761: Protocol Independent Multicast - Sparse Mode
      (PIM-SM): Protocol Specification (Revised), Section 4.1.3";
  }
  leaf mode {
```

```
    type identityref {
      base pim-mode;
    }
    description
      "PIM mode.";
  }
  leaf msdp-learned {
    type boolean;
    description
      "'true' if the route is learned from MSDP (the Multicast
      Source Discovery Protocol).";
  }
  leaf rp-address {
    type inet:ip-address;
    description
      "RP (Rendezvous Point) address.";
  }
  leaf rpf-neighbor {
    type inet:ip-address;
    description
      "RPF (Reverse Path Forwarding) neighbor address.";
  }
  leaf up-time {
    type rt-types:timeticks64;
    description
      "The number of time ticks (hundredths of a second) since the
      route last transitioned into the active state.";
  }
  list outgoing-interface {
    key "name";
    description
      "A list of outgoing interfaces.";

    leaf name {
      type if:interface-ref;
      description
        "Interface name.";
    }

    leaf expiration {
      type rt-types:timer-value-seconds16;
      description
        "Expiration time.";
    }

    leaf up-time {
      type rt-types:timeticks64;
      description
        "The number of time ticks (hundredths of a second) since
        the 'oper-status' setting of the interface was last
        changed to 'up'.";
    }

    leaf jp-state {
      type enumeration {
        enum "no-info" {
          description
            "The interface has no (*,G) Join state and no timers";
        }
      }
    }
  }
}
```

```

        running.";
    }
    enum "join" {
        description
            "The interface has Join state.";
    }
    enum "prune-pending" {
        description
            "The router has received a Prune on this interface from
            a downstream neighbor and is waiting to see whether
            the Prune will be overridden by another downstream
            router. For forwarding purposes, the Prune-Pending
            state functions exactly like the Join state.";
    }
    }
    description
        "Join/Prune state.";
    }
} // multicast-route-attributes

grouping neighbor-state-af-attributes {
    description
        "A grouping defining neighbor per address family attributes.";
    leaf bfd-state {
        type bfd-types:state;
        description
            "BFD (Bidirectional Forwarding Detection) status.";
    }
    leaf expiration {
        type rt-types:timer-value-seconds16;
        description
            "Neighbor expiration time.";
    }
    leaf dr-priority {
        type uint32;
        description
            "DR (Designated Router) priority as the preference in the DR
            election process.";
    }
    leaf gen-id {
        type uint32;
        description
            "The value of the Generation ID in the last Hello message
            from the neighbor.";
    }
    container lan-prune-delay {
        description
            "The information of the LAN Prune Delay option in the Hello
            message from the neighbor.";
        leaf present {
            type boolean;
            description
                "'true' if the LAN Prune Delay option is present in the
                last Hello message from the neighbor.";
        }
        leaf override-interval {
            when "../present = 'true'" {

```



```

        description
            "Available only when 'leaf present' is 'true'.";
    }
    type uint16;
    units milliseconds;
    description
        "The value of the Override_Interval field of the LAN Prune
        Delay option in the last Hello message from the neighbor.
        The neighbor uses this value to indicate a short period
        after a Join or Prune to allow other routers on the LAN
        to override the Join or Prune.";
    }
    leaf propagation-delay {
        when "../present = 'true'" {
            description
                "Available only when 'leaf present' is 'true'.";
        }
        type uint16;
        units milliseconds;
        description
            "The value of the Propagation_Delay field of the LAN Prune
            Delay option in the last Hello message from the neighbor.
            The value is the propagation delay over the local link
            expected by the neighbor.";
    }
    leaf t-bit {
        when "../present = 'true'" {
            description
                "Available only when 'leaf present' is 'true'.";
        }
        type boolean;
        description
            "'true' if the T bit is set in the LAN Prune Delay option
            in the last Hello message from the neighbor. This flag
            indicates the neighbor's ability to disable Join
            message suppression.";
    }
    }
    leaf up-time {
        type rt-types:timeticks64;
        description
            "The number of time ticks (hundredths of a second) since
            the neighbor relationship has been formed as reachable
            without being timed out.";
    }
} // neighbor-state-af-attributes

grouping pim-instance-af-state-ref {
    description
        "An absolute reference to a PIM instance address family.";
    leaf instance-af-ref {
        type leafref {
            path "/rt:routing/rt:control-plane-protocols/"
                + "pim-base:pim/pim-base:address-family/"
                + "pim-base:address-family";
        }
        description
            "Reference to a PIM instance address family.";
    }
}

```

```
    }
  } // pim-instance-af-state-ref

  grouping pim-interface-state-ref {
    description
      "An absolute reference to a PIM interface state.";
    leaf interface-ref {
      type leafref {
        path "/rt:routing/rt:control-plane-protocols/"
          + "pim-base:pim/pim-base:interfaces/pim-base:interface/"
          + "pim-base:name";
      }
      description
        "Reference to a PIM interface.";
    }
  } // pim-interface-state-ref

  grouping statistics-sent-received {
    description
      "A grouping defining sent and received statistics
      on PIM messages.";
    reference
      "RFC 3973: Protocol Independent Multicast - Dense Mode
      (PIM-DM): Protocol Specification (Revised), Section 4.7.1
      RFC 5015: Bidirectional Protocol Independent Multicast
      (BIDIR-PIM), Section 3.7
      RFC 7761: Protocol Independent Multicast - Sparse Mode
      (PIM-SM): Protocol Specification (Revised), Section 4.9";
    leaf assert {
      type yang:counter64;
      description
        "The number of Assert messages, with the message Type
        of 5 (RFCs 3973 and 7761).";
      reference
        "RFC 3973: Protocol Independent Multicast - Dense Mode
        (PIM-DM): Protocol Specification (Revised)
        RFC 7761: Protocol Independent Multicast - Sparse Mode
        (PIM-SM): Protocol Specification (Revised)";
    }
    leaf bsr {
      type yang:counter64;
      description
        "The number of Bootstrap messages, with the message Type
        of 4 (RFCs 3973 and 7761).";
    }
    leaf candidate-rp-advertisement {
      type yang:counter64;
      description
        "The number of Candidate RP Advertisement messages, with the
        message Type of 8 (RFCs 3973 and 7761).";
    }
    leaf df-election {
      type yang:counter64;
      description
        "The number of DF (Designated Forwarder) election messages,
        with the message Type of 10 (RFC 5015).";
      reference
        "RFC 5015: Bidirectional Protocol Independent Multicast
```

```
    (BIDIR-PIM)";
  }
  leaf graft {
    type yang:counter64;
    description
      "The number of Graft messages, with the message Type
       of 6 (RFCs 3973 and 7761).";
  }
  leaf graft-ack {
    type yang:counter64;
    description
      "The number of Graft-Ack messages, with the message Type
       of 7 (RFCs 3973 and 7761).";
  }
  leaf hello {
    type yang:counter64;
    description
      "The number of Hello messages, with the message Type
       of 0 (RFCs 3973 and 7761).";
  }
  leaf join-prune {
    type yang:counter64;
    description
      "The number of Join/Prune messages, with the message Type
       of 3 (RFCs 3973 and 7761).";
  }
  leaf register {
    type yang:counter64;
    description
      "The number of Register messages, with the message Type
       of 1 (RFCs 3973 and 7761).";
  }
  leaf register-stop {
    type yang:counter64;
    description
      "The number of Register-Stop messages, with the message Type
       of 2 (RFCs 3973 and 7761).";
  }
  leaf state-refresh {
    type yang:counter64;
    description
      "The number of State Refresh messages, with the message Type
       of 9 (RFC 3973).";
  }
} // statistics-sent-received

/*
 * Data nodes
 */

augment "/rt:routing/rt:control-plane-protocols" {
  description
    "PIM augmentation to the routing instance model.";

  container pim {
    presence
      "Enables the PIM protocol.";
    description

```

```
"PIM configuration data and operational state data.";

uses graceful-restart-container {
  if-feature global-graceful-restart;
}

list address-family {
  key "address-family";
  description
    "Each list entry for one address family.";
  uses rt:address-family;
  uses graceful-restart-container {
    if-feature per-af-graceful-restart;
  }

  container statistics {
    config false;
    description
      "A container defining statistics attributes.";
    leaf discontinuity-time {
      type yang:date-and-time;
      description
        "The time of the most recent occasion at which any one
        or more of the statistics counters suffered a
        discontinuity.  If no such discontinuities have
        occurred since the last reinitialization of the local
        management subsystem, then this node contains the time
        the local management subsystem reinitialized
        itself.";
    }
    container error {
      description
        "Contains error statistics.";
      uses statistics-sent-received {
        description
          "Statistics counters on the PIM messages per PIM
          message Type.  Each leaf attribute counts the number
          of PIM messages that were of a particular Type (such
          as Hello) and contained errors preventing them from
          being processed by PIM.

          Such messages are also counted by the corresponding
          counter of the same Type (such as Hello) in the
          'received' container.";
      }
    }
    leaf checksum {
      type yang:counter64;
      description
        "The number of PIM messages that were passed to PIM
        and contained checksum errors.";
    }
    leaf format {
      type yang:counter64;
      description
        "The number of PIM messages that passed checksum
        validation but contained format errors, including
        errors related to PIM Version, Type, and message
        length.";
    }
  }
}
```

```
    }
  }
  container queue {
    description
      "Contains queue statistics.";
    leaf size {
      type uint32;
      description
        "The size of the input queue.";
    }
    leaf overflow {
      type yang:counter32;
      description
        "The number of input queue overflows.";
    }
  }
  container received {
    description
      "Contains statistics of received messages.";
    uses statistics-sent-received;
  }
  container sent {
    description
      "Contains statistics of sent messages.";
    uses statistics-sent-received;
  }
}

container topology-tree-info {
  config false;
  description
    "Contains topology tree information.";
  list ipv4-route {
    when "../..address-family = 'rt:ipv4'" {
      description
        "Only applicable to an IPv4 address family.";
    }
    key "group source-address is-rpt";
    description
      "A list of IPv4 routes.";
    leaf group {
      type rt-types:ipv4-multicast-group-address;
      description
        "Group address.";
    }
    leaf source-address {
      type rt-types:ipv4-multicast-source-address;
      description
        "Source address.";
    }
    leaf is-rpt {
      type boolean;
      description
        "'true' if the tree is an RPT
        (Rendezvous Point Tree).";
    }
  }
  uses multicast-route-attributes;
}
```

```

    } // ipv4-route

    list ipv6-route {
      when "../..address-family = 'rt:ipv6'" {
        description
          "Only applicable to an IPv6 address family.";
      }
      key "group source-address is-rpt";
      description
        "A list of IPv6 routes.";
      leaf group {
        type rt-types:ipv6-multicast-group-address;
        description
          "Group address.";
      }
      leaf source-address {
        type rt-types:ipv6-multicast-source-address;
        description
          "Source address.";
      }
      leaf is-rpt {
        type boolean;
        description
          "'true' if the tree is an RPT.";
      }
      uses multicast-route-attributes;
    } // ipv6-route
  } // topology-tree-info
} // address-family

container interfaces {
  description
    "Contains a list of interfaces.";
  list interface {
    key "name";
    description
      "List of PIM interfaces.";
    leaf name {
      type if:interface-ref;
      description
        "Reference to an entry in the global interface list.";
    }
  }
  list address-family {
    key "address-family";
    description
      "Each list entry for one address family.";
    uses rt:address-family;

    container bfd {
      if-feature bfd;
      description
        "BFD (Bidirectional Forwarding Detection)
        operation.";
      uses bfd-types:client-cfg-parms;
    }
    leaf dr-priority {
      if-feature intf-dr-priority;
    }
  }
}

```

```
    type uint32;
    default 1;
    description
      "DR (Designated Router) priority as the preference in
       the DR election process.";
  }
  leaf hello-interval {
    if-feature intf-hello-interval;
    type rt-types:timer-value-seconds16;
    default 30;
    description
      "Periodic interval for Hello messages.
       If 'infinity' or 'not-set' is used, no periodic
       Hello messages are sent.";
    reference
      "RFC 3973: Protocol Independent Multicast -
       Dense Mode (PIM-DM): Protocol Specification
       (Revised), Section 4.8
       RFC 7761: Protocol Independent Multicast - Sparse
       Mode (PIM-SM): Protocol Specification (Revised),
       Section 4.11";
  }
  choice hello-holdtime-or-multiplier {
    description
      "The Holdtime is the timer value to time out the
       neighbor state when the timer expires.
       The Holdtime value can be specified by either
       (1) the given Holdtime value or (2) the calculation
       of the Hello interval multiplied by the given value
       of the multiplier.";
    case holdtime {
      if-feature intf-hello-holdtime;
      leaf hello-holdtime {
        type rt-types:timer-value-seconds16;
        default 105;
        description
          "The Hello Holdtime is the amount of time to
           keep the neighbor reachable until a new
           Hello message is received.";
      }
    }
    case multiplier {
      if-feature intf-hello-multiplier;
      leaf hello-multiplier {
        type rt-types:timer-multiplier;
        default 3;
        description
          "The Hello multiplier is the number by which the
           Hello interval is multiplied to obtain the
           Hello Holdtime.
           The value of the Hello Holdtime is calculated
           as:
           hello-holdtime =
           (multiplier + 0.5) * (hello-interval).";
      }
    }
  }
  leaf jp-interval {
```

```
    if-feature intf-jp-interval;
    type rt-types:timer-value-seconds16;
    default 60;
    description
      "Periodic interval between Join/Prune messages.
       If 'infinity' or 'not-set' is used, no periodic
       Join/Prune messages are sent.";
  }
  choice jp-holdtime-or-multiplier {
    description
      "The Join/Prune Holdtime is the amount of time a
       receiver must keep the Join/Prune state alive.
       The Holdtime value can be specified by either
       (1) the given Holdtime value or (2) the calculation
       of 'jp-interval' multiplied by the given value of
       the multiplier.";
    case holdtime {
      if-feature intf-jp-holdtime;
      leaf jp-holdtime {
        type rt-types:timer-value-seconds16;
        default 210;
        description
          "The Join/Prune Holdtime is the amount of time a
           receiver must keep the Join/Prune state alive.";
      }
    }
    case multiplier {
      if-feature intf-jp-multiplier;
      leaf jp-multiplier {
        type rt-types:timer-multiplier;
        default 3;
        description
          "The Join/Prune multiplier is the number
           by which the Join/Prune interval is multiplied
           to obtain the Join/Prune Holdtime.
           The value of the Join/Prune Holdtime is
           calculated as:
           jp-holdtime =
           (multiplier + 0.5) * (jp-interval).";
      }
    }
  }
  leaf override-interval {
    if-feature intf-override-interval;
    type uint16;
    units milliseconds;
    default 2500;
    description
      "A short period after a Join or Prune to allow other
       routers on the LAN to override the Join or Prune.";
  }
  leaf propagation-delay {
    if-feature intf-propagation-delay;
    type uint16;
    units milliseconds;
    default 500;
    description
      "Expected propagation delay over the local link.";
  }
}
```



```
}  
  
// Interface state attributes  
leaf oper-status {  
  type enumeration {  
    enum up {  
      description  
        "The interface is ready to pass PIM messages.";  
    }  
    enum down {  
      description  
        "The interface does not pass PIM messages.";  
    }  
  }  
  config false;  
  description  
    "PIM operational status on the interface.  
    This status is PIM specific and separate from the  
    operational status of the underlying interface.";  
}  
leaf gen-id {  
  type uint32;  
  config false;  
  description  
    "The value of the Generation ID this router uses to  
    insert into the PIM Hello message sent on this  
    interface.";  
}  
leaf hello-expiration {  
  type rt-types:timer-value-seconds16;  
  config false;  
  description  
    "Hello interval expiration time.";  
}  
container ipv4 {  
  when "../address-family = 'rt:ipv4'" {  
    description  
      "Only applicable to an IPv4 address family.";  
  }  
  config false;  
  description  
    "Interface state attributes for IPv4.";  
  leaf-list address {  
    type inet:ipv4-address;  
    description  
      "List of addresses on which PIM is operating.";  
  }  
  leaf dr-address {  
    type inet:ipv4-address;  
    description  
      "DR (Designated Router) address.";  
  }  
}  
container ipv6 {  
  when "../address-family = 'rt:ipv6'" {  
    description  
      "Only applicable to an IPv6 address family.";  
  }  
}
```

```

    config false;
    description
      "Interface state attributes for IPv6.";
    leaf-list address {
      type inet:ipv6-address;
      description
        "List of addresses on which PIM is operating.";
    }
    leaf dr-address {
      type inet:ipv6-address;
      description
        "DR address.";
    }
  }
  container neighbors {
    config false;
    description
      "Information learned from neighbors through this
      interface.";
    list ipv4-neighbor {
      when "../..address-family = 'rt:ipv4'" {
        description
          "Only applicable to an IPv4 address family.";
      }
      key "address";
      description
        "Neighbor state information.";
      leaf address {
        type inet:ipv4-address;
        description
          "Neighbor address.";
      }
      uses neighbor-state-af-attributes;
    } // list ipv4-neighbor
    list ipv6-neighbor {
      when "../..address-family = 'rt:ipv6'" {
        description
          "Only applicable to an IPv6 address family.";
      }
      key "address";
      description
        "Neighbor state information.";
      leaf address {
        type inet:ipv6-address;
        description
          "Neighbor address.";
      }
      uses neighbor-state-af-attributes;
    } // list ipv6-neighbor
  } // neighbors
} // address-family
} // interface
} // interfaces
} // pim
} // augment

/*
 * Notifications

```

```

*/
notification pim-neighbor-event {
  description
    "Notification event for a neighbor.";
  leaf event-type {
    type neighbor-event-type;
    description
      "Event type.";
  }
  uses pim-interface-state-ref;
  leaf interface-af-ref {
    type leafref {
      path "/rt:routing/rt:control-plane-protocols/"
        + "pim-base:pim/pim-base:interfaces/pim-base:interface"
        + "[pim-base:name = current()/../interface-ref]/"
        + "pim-base:address-family/pim-base:address-family";
    }
    description
      "Reference to a PIM interface address family.";
  }
  leaf neighbor-ipv4-ref {
    when "../interface-af-ref = 'rt:ipv4'" {
      description
        "Only applicable to an IPv4 address family.";
    }
    type leafref {
      path "/rt:routing/rt:control-plane-protocols/"
        + "pim-base:pim/pim-base:interfaces/pim-base:interface"
        + "[pim-base:name = current()/../interface-ref]/"
        + "pim-base:address-family"
        + "[pim-base:address-family = "
        + "current()/../interface-af-ref]/"
        + "pim-base:neighbors/pim-base:ipv4-neighbor/"
        + "pim-base:address";
    }
    description
      "Reference to a PIM IPv4 neighbor.";
  }
  leaf neighbor-ipv6-ref {
    when "../interface-af-ref = 'rt:ipv6'" {
      description
        "Only applicable to an IPv6 address family.";
    }
    type leafref {
      path "/rt:routing/rt:control-plane-protocols/"
        + "pim-base:pim/pim-base:interfaces/pim-base:interface"
        + "[pim-base:name = current()/../interface-ref]/"
        + "pim-base:address-family"
        + "[pim-base:address-family = "
        + "current()/../interface-af-ref]/"
        + "pim-base:neighbors/pim-base:ipv6-neighbor/"
        + "pim-base:address";
    }
    description
      "Reference to a PIM IPv6 neighbor.";
  }
  leaf up-time {
    type rt-types:timeticks64;
  }
}

```

```
    description
      "The number of time ticks (hundredths of a second) since
       the neighbor relationship has been formed as reachable
       without being timed out.";
  }
}
notification pim-interface-event {
  description
    "Notification event for an interface.";
  leaf event-type {
    type interface-event-type;
    description
      "Event type.";
  }
  uses pim-interface-state-ref;
  container ipv4 {
    description
      "Contains IPv4 information.";
    leaf-list address {
      type inet:ipv4-address;
      description
        "List of addresses.";
    }
    leaf dr-address {
      type inet:ipv4-address;
      description
        "DR (Designated Router) address.";
    }
  }
  container ipv6 {
    description
      "Contains IPv6 information.";
    leaf-list address {
      type inet:ipv6-address;
      description
        "List of addresses.";
    }
    leaf dr-address {
      type inet:ipv6-address;
      description
        "DR address.";
    }
  }
}
}
```

<CODE ENDS>

## 6.2. PIM RP Module

This module references [\[RFC5059\]](#), [\[RFC6991\]](#), [\[RFC7761\]](#), [\[RFC8294\]](#), [\[RFC8343\]](#), and [\[RFC8349\]](#).

```
<CODE BEGINS> file "ietf-pim-rp@2021-09-03.yang"

module ietf-pim-rp {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-pim-rp";
  prefix pim-rp;

  import ietf-inet-types {
    prefix "inet";
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-routing-types {
    prefix "rt-types";
    reference
      "RFC 8294: Common YANG Data Types for the Routing Area";
  }

  import ietf-interfaces {
    prefix "if";
    reference
      "RFC 8343: A YANG Data Model for Interface Management";
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC 8349: A YANG Data Model for Routing Management (NMDA
      Version)";
  }

  import ietf-pim-base {
    prefix "pim-base";
    reference
      "RFC 9128: A YANG Data Model for Protocol Independent
      Multicast (PIM)";
  }

  organization
    "IETF PIM Working Group";

  contact
    "WG Web: <https://datatracker.ietf.org/wg/pim/>
    WG List: <mailto:pim@ietf.org>

    Editor: Xufeng Liu
           <mailto:xufeng.liu.ietf@gmail.com>

    Editor: Pete McAllister
           <mailto:pete.mcallister@metaswitch.com>

    Editor: Anish Peter
           <mailto:anish.ietf@gmail.com>

    Editor: Mahesh Sivakumar
           <mailto:sivakumar.mahesh@gmail.com>
```

```
Editor: Yisong Liu
       <mailto:liuyisong@huawei.com>

Editor: Fangwei Hu
       <mailto:hu.fangwei@zte.com.cn>;

description
"This YANG module defines a PIM (Protocol Independent Multicast)
RP (Rendezvous Point) model.

Copyright (c) 2021 IETF Trust and the persons identified as
authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or
without modification, is permitted pursuant to, and subject to
the license terms contained in, the Simplified BSD License set
forth in Section 4.c of the IETF Trust's Legal Provisions
Relating to IETF Documents
(https://trustee.ietf.org/license-info).

This version of this YANG module is part of RFC 9128; see the
RFC itself for full legal notices.";

revision 2021-09-03 {
  description
    "Initial revision.";
  reference
    "RFC 9128: A YANG Data Model for Protocol Independent
    Multicast (PIM)";
}

/*
 * Features
 */
feature bsr {
  description
    "This feature indicates that the system supports BSRs
    (Bootstrap Routers).";
  reference
    "RFC 5059: Bootstrap Router (BSR) Mechanism for Protocol
    Independent Multicast (PIM)";
}

feature bsr-election-state {
  if-feature bsr;
  description
    "This feature indicates that the system supports providing
    BSR election state.";
  reference
    "RFC 5059: Bootstrap Router (BSR) Mechanism for Protocol
    Independent Multicast (PIM)";
}

feature static-rp-override {
  description
    "This feature indicates that the system supports configuration
    of static RP (Rendezvous Point) override.";
```

```
reference
  "RFC 7761: Protocol Independent Multicast - Sparse Mode
  (PIM-SM): Protocol Specification (Revised), Section 3.7";
}

feature candidate-interface {
  description
    "This feature indicates that the system supports using
    an interface to configure a BSR or RP candidate.";
}

feature candidate-ipv4 {
  description
    "This feature indicates that the system supports using
    an IPv4 address to configure a BSR or RP candidate.";
}

feature candidate-ipv6 {
  description
    "This feature indicates that the system supports using
    an IPv6 address to configure a BSR or RP candidate.";
}

/*
 * Typedefs
 */
typedef rp-event-type {
  type enumeration {
    enum invalid-join {
      description
        "An invalid Join/Prune message has been received.";
    }
    enum invalid-register {
      description
        "An invalid Register message has been received.";
    }
    enum mapping-created {
      description
        "A new mapping has been created.";
    }
    enum mapping-deleted {
      description
        "A mapping has been deleted.";
    }
  }
  description
    "Operational status event type for notifications.";
}

/*
 * Identities
 */
identity rp-mode {
  description
    "The mode of an RP, which can be SM (Sparse Mode) or
    BIDIR (Bidirectional).";
}
```



```
identity rp-info-source-type {
  description
    "The information source of an RP.";
}
identity static {
  base rp-info-source-type;
  description
    "The RP is statically configured.";
}
identity bootstrap {
  base rp-info-source-type;
  description
    "The RP is learned from a Bootstrap.";
}

/*
 * Groupings
 */
grouping rp-mapping-state-attributes {
  description
    "Grouping of RP mapping attributes.";
  leaf up-time {
    type rt-types:timeticks64;
    description
      "The number of time ticks (hundredths of a second) since
       the RP mapping or the RP became actively available.";
  }
  leaf expiration {
    type rt-types:timer-value-seconds16;
    description
      "Expiration time.";
  }
} // rp-mapping-state-attributes

grouping rp-state-attributes {
  description
    "Grouping of RP state attributes.";
  leaf info-source-type {
    type identityref {
      base rp-info-source-type;
    }
    description
      "The information source of an RP.";
  } // info-source-type
  leaf up-time {
    type rt-types:timeticks64;
    description
      "The number of time ticks (hundredths of a second) since
       the RP became actively available.";
  }
  leaf expiration {
    type rt-types:timer-value-seconds16;
    description
      "Expiration time.";
  }
} // rp-state-attributes
```

```
grouping static-rp-attributes {
  description
    "Grouping of static RP attributes, used in augmenting
    modules.";
  leaf policy-name {
    type string;
    description
      "The string value is the name to uniquely identify a
      policy that contains one or more policy rules used to
      determine which multicast group addresses are mapped
      to this statically configured RP address.
      If a policy is not specified, the entire multicast address
      space is mapped.
      The definition of such a policy is outside the scope
      of this document.";
  }
  leaf override {
    if-feature static-rp-override;
    type boolean;
    default false;
    description
      "When there is a conflict between static RPs and dynamic RPs,
      setting this attribute to 'true' will ask the system to use
      static RPs.";
  }
} // static-rp-attributes

grouping rp-candidate-attributes {
  description
    "Grouping of RP candidate attributes.";
  leaf policy-name {
    type string;
    description
      "The string value is the name to uniquely identify a
      policy that contains one or more policy rules used to
      accept or reject certain multicast groups.
      If a policy is not specified, the entire multicast address
      space is accepted.
      The definition of such a policy is outside the scope
      of this document.";
  }
  leaf mode {
    type identityref {
      base rp-mode;
    }
    description
      "The mode of an RP, which can be SM (Sparse Mode) or BIDIR
      (Bidirectional). Each of these modes is defined in a
      separate YANG module. If a system supports an RP mode,
      the corresponding YANG module is implemented.
      When the value of this leaf is not specified, the default
      value is the supported mode if only one mode is implemented,
      or the default value is SM if both SM and BIDIR are
      implemented.";
  }
} // rp-candidate-attributes

/*
```

```
* Configuration data nodes
*/

augment "/rt:routing/rt:control-plane-protocols/pim-base:pim/"
+ "pim-base:address-family" {
  description
    "PIM RP augmentation.";

  container rp {
    description
      "PIM RP configuration data.";

    container static-rp {
      description
        "Contains static RP attributes.";
      list ipv4-rp {
        when "../..../pim-base:address-family = 'rt:ipv4'" {
          description
            "Only applicable to an IPv4 address family.";
        }
        key "rp-address";
        description
          "A list of IPv4 RP addresses.";
        leaf rp-address {
          type inet:ipv4-address;
          description
            "Specifies a static RP address.";
        }
      }

      list ipv6-rp {
        when "../..../pim-base:address-family = 'rt:ipv6'" {
          description
            "Only applicable to an IPv6 address family.";
        }
        key "rp-address";
        description
          "A list of IPv6 RP addresses.";
        leaf rp-address {
          type inet:ipv6-address;
          description
            "Specifies a static RP address.";
        }
      }
    } // static-rp

    container bsr {
      if-feature bsr;
      description
        "Contains BSR (Bootstrap Router) attributes.";
      container bsr-candidate {
        presence
          "Present to serve as a BSR candidate";
        description
          "BSR candidate attributes.";

        choice interface-or-address {
          description

```

```
    "Use either an interface or an IP address.";
  case interface {
    if-feature candidate-interface;
    leaf interface {
      type if:interface-ref;
      mandatory true;
      description
        "Interface to be used by a BSR.";
    }
  }
  case ipv4-address {
    when "../..../pim-base:address-family = 'rt:ipv4'" {
      description
        "Only applicable to an IPv4 address family.";
    }
    if-feature candidate-ipv4;
    leaf ipv4-address {
      type inet:ipv4-address;
      mandatory true;
      description
        "IP address to be used by a BSR.";
    }
  }
  case ipv6-address {
    when "../..../pim-base:address-family = 'rt:ipv6'" {
      description
        "Only applicable to an IPv6 address family.";
    }
    if-feature candidate-ipv6;
    leaf ipv6-address {
      type inet:ipv6-address;
      mandatory true;
      description
        "IP address to be used by a BSR.";
    }
  }
}

leaf hash-mask-length{
  type uint8 {
    range "0..128";
  }
  mandatory true;
  description
    "Value contained in BSR messages used by all routers to
    hash (map) to an RP.";
}

leaf priority {
  type uint8 {
    range "0..255";
  }
  default 64;
  description
    "BSR election priority among different candidate BSRs.
    A larger value has a higher priority over a smaller
    value.";
}
```

```
    } // bsr-candidate

    container rp-candidate {
      description
        "Contains RP candidate attributes.";
      list interface {
        if-feature candidate-interface;
        key "name";
        description
          "A list of RP candidates.";
        leaf name {
          type if:interface-ref;
          description
            "Interface that the RP candidate uses.";
        }
        uses rp-candidate-attributes;
      }

      list ipv4-address {
        when "../..../pim-base:address-family = 'rt:ipv4'" {
          description
            "Only applicable to an IPv4 address family.";
        }
        if-feature candidate-ipv4;
        key "address";
        description
          "A list of RP candidate addresses.";
        leaf address {
          type inet:ipv4-address;
          description
            "IPv4 address that the RP candidate uses.";
        }
        uses rp-candidate-attributes;
      }

      list ipv6-address {
        when "../..../pim-base:address-family = 'rt:ipv6'" {
          description
            "Only applicable to an IPv6 address family.";
        }
        if-feature candidate-ipv6;
        key "address";
        description
          "A list of RP candidate addresses.";
        leaf address {
          type inet:ipv6-address;
          description
            "IPv6 address that the RP candidate uses.";
        }
        uses rp-candidate-attributes;
      }
    }

    // BSR state attributes
    container bsr {
      config false;
      description
        "BSR information.";
```

```
leaf address {
  type inet:ip-address;
  description
    "BSR address.";
}
leaf hash-mask-length {
  type uint8 {
    range "0..128";
  }
  description
    "Hash mask length.";
}
leaf priority {
  type uint8 {
    range "0..255";
  }
  description
    "Priority.";
}
leaf up-time {
  type rt-types:timeticks64;
  description
    "The number of time ticks (hundredths of a second)
    since the BSR came up.";
}
}
choice election-state {
  if-feature bsr-election-state;
  config false;
  description
    "BSR election state.";
  case candidate {
    leaf candidate-bsr-state {
      type enumeration {
        enum "candidate" {
          description
            "The router is a candidate to be the BSR for the
            scope zone, but currently another router is the
            preferred BSR.";
        }
        enum "pending" {
          description
            "The router is a candidate to be the BSR for the
            scope zone. Currently, no other router is the
            preferred BSR, but this router is not yet the
            elected BSR. This is a temporary state that
            prevents rapid thrashing of the choice of BSR
            during BSR election.";
        }
        enum "elected" {
          description
            "The router is the elected BSR for the
            scope zone, and it must perform all of the
            BSR functions.";
        }
      }
    }
  }
  description
    "Candidate-BSR (C-BSR) state.";
```

```

        reference
            "RFC 5059: Bootstrap Router (BSR) Mechanism for
            Protocol Independent Multicast (PIM),
            Section 3.1.1";
    }
}
case "non-candidate" {
    leaf non-candidate-bsr-state {
        type enumeration {
            enum "no-info" {
                description
                    "The router has no information about this scope
                    zone.";
            }
            enum "accept-any" {
                description
                    "The router does not know of an active BSR and
                    will accept the first Bootstrap message it sees
                    that provides the new BSR's identity and the
                    RP-Set.";
            }
            enum "accept" {
                description
                    "The router knows the identity of the current
                    BSR and is using the RP-Set provided by that
                    BSR. Only Bootstrap messages from that BSR or
                    from a Candidate-BSR (C-BSR) with higher weight
                    than the current BSR will be accepted.";
            }
        }
        description
            "Non-Candidate-BSR state.";
        reference
            "RFC 5059: Bootstrap Router (BSR) Mechanism for
            Protocol Independent Multicast (PIM),
            Section 3.1.2";
    }
} // election-state
leaf bsr-next-bootstrap {
    type uint16;
    units seconds;
    config false;
    description
        "The remaining time interval in seconds until the next
        Bootstrap will be sent.";
}

container rp {
    config false;
    description
        "State information of the RP.";
    leaf rp-address {
        type inet:ip-address;
        description
            "RP address.";
    }
    leaf policy-name {

```

```
    type string;
    description
      "The string value is the name to uniquely identify a
      policy that contains one or more policy rules used to
      accept or reject certain multicast groups.
      If a policy is not specified, the entire multicast
      address space is accepted.
      The definition of such a policy is outside the scope
      of this document.";
  }
  leaf up-time {
    type rt-types:timeticks64;
    description
      "The number of time ticks (hundredths of a second)
      since the RP became actively available.";
  }
}
leaf rp-candidate-next-advertisement {
  type uint16;
  units seconds;
  config false;
  description
    "The remaining time interval in seconds until the next
    RP candidate advertisement will be sent.";
}
} // bsr

container rp-list {
  config false;
  description
    "Contains a list of RPs.";
  list ipv4-rp {
    when "../.../pim-base:address-family = 'rt:ipv4'" {
      description
        "Only applicable to an IPv4 address family.";
    }
    key "rp-address mode";
    description
      "A list of IPv4 RP addresses.";
    leaf rp-address {
      type inet:ipv4-address;
      description
        "RP address.";
    }
    leaf mode {
      type identityref {
        base rp-mode;
      }
      description
        "RP mode.";
    }
    leaf info-source-address {
      type inet:ipv4-address;
      description
        "The address where RP information is learned.";
    }
  }
  uses rp-state-attributes;
}
```



```
list ipv6-rp {
  when "../..../pim-base:address-family = 'rt:ipv6'" {
    description
      "Only applicable to an IPv6 address family.";
  }
  key "rp-address mode";
  description
    "A list of IPv6 RP addresses.";
  leaf rp-address {
    type inet:ipv6-address;
    description
      "RP address.";
  }
  leaf mode {
    type identityref {
      base rp-mode;
    }
    description
      "RP mode.";
  }
  leaf info-source-address {
    type inet:ipv6-address;
    description
      "The address where RP information is learned.";
  }
  uses rp-state-attributes;
}
} // rp-list

container rp-mappings {
  config false;
  description
    "Contains a list of group-to-RP mappings.";
  list ipv4-rp {
    when "../..../pim-base:address-family = 'rt:ipv4'" {
      description
        "Only applicable to an IPv4 address family.";
    }
    key "group-range rp-address";
    description
      "A list of group-to-RP mappings.";
    leaf group-range {
      type inet:ipv4-prefix;
      description
        "Group range presented in the format of a prefix.";
    }
    leaf rp-address {
      type inet:ipv4-address;
      description
        "RP address.";
    }
    uses rp-mapping-state-attributes;
  }

  list ipv6-rp {
    when "../..../pim-base:address-family = 'rt:ipv6'" {
      description
```

```

        "Only applicable to an IPv6 address family.";
    }
    key "group-range rp-address";
    description
        "A list of IPv6 RP addresses.";
    leaf group-range {
        type inet:ipv6-prefix;
        description
            "Group range presented in the format of a prefix.";
    }
    leaf rp-address {
        type inet:ipv6-address;
        description
            "RP address.";
    }
    uses rp-mapping-state-attributes;
} // rp-mappings
} // rp
} // augment

/*
 * Notifications
 */
notification pim-rp-event {
    description
        "Notification event for an RP.";
    leaf event-type {
        type rp-event-type;
        description
            "Event type.";
    }
    uses pim-base:pim-instance-af-state-ref;
    leaf group {
        type rt-types:ip-multicast-group-address;
        description
            "Group address.";
    }
    leaf rp-address {
        type inet:ip-address;
        description
            "RP address.";
    }
    leaf is-rpt {
        type boolean;
        description
            "'true' if the tree is an RPT (Rendezvous Point Tree).";
    }
    leaf mode {
        type identityref {
            base pim-base:pim-mode;
        }
        description
            "PIM mode.";
    }
    leaf message-origin {
        type inet:ip-address;
        description

```

```
    "Where the message originated.";  
  }  
}  
<CODE ENDS>
```

### 6.3. PIM-SM Module

This module references [\[RFC4607\]](#), [\[RFC6991\]](#), [\[RFC7761\]](#), and [\[RFC8349\]](#).

```
<CODE BEGINS> file "ietf-pim-sm@2021-09-03.yang"

module ietf-pim-sm {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-pim-sm";
  prefix pim-sm;

  import ietf-inet-types {
    prefix "inet";
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC 8349: A YANG Data Model for Routing Management (NMDA
      Version)";
  }

  import ietf-pim-base {
    prefix "pim-base";
    reference
      "RFC 9128: A YANG Data Model for Protocol Independent
      Multicast (PIM)";
  }

  import ietf-pim-rp {
    prefix "pim-rp";
    reference
      "RFC 9128: A YANG Data Model for Protocol Independent
      Multicast (PIM)";
  }

  organization
    "IETF PIM Working Group";

  contact
    "WG Web: <https://datatracker.ietf.org/wg/pim/>
    WG List: <mailto:pim@ietf.org>

    Editor: Xufeng Liu
           <mailto:xufeng.liu.ietf@gmail.com>

    Editor: Pete McAllister
           <mailto:pete.mcallister@metaswitch.com>

    Editor: Anish Peter
           <mailto:anish.ietf@gmail.com>

    Editor: Mahesh Sivakumar
           <mailto:sivakumar.mahesh@gmail.com>

    Editor: Yisong Liu
           <mailto:liuyisong@huawei.com>

    Editor: Fangwei Hu
```

```
        <mailto:hu.fangwei@zte.com.cn>;
description
  "This YANG module defines a PIM (Protocol Independent Multicast)
  SM (Sparse Mode) model.

  Copyright (c) 2021 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject to
  the license terms contained in, the Simplified BSD License set
  forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (https://trustee.ietf.org/license-info).

  This version of this YANG module is part of RFC 9128; see the
  RFC itself for full legal notices."

revision 2021-09-03 {
  description
    "Initial revision.";
  reference
    "RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.2
    RFC 9128: A YANG Data Model for Protocol Independent
    Multicast (PIM)";
}

/*
 * Features
 */
feature spt-switch-infinity {
  description
    "This feature indicates that the system supports the
    configuration choice of whether to trigger switchover from
    the RPT (Rendezvous Point Tree) to the SPT (Shortest Path
    Tree).";
  reference
    "RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.2";
}

feature spt-switch-policy {
  description
    "This feature indicates that the system supports configuring
    the policy for switchover from the RPT to the SPT.";
  reference
    "RFC 7761: Protocol Independent Multicast - Sparse Mode
    (PIM-SM): Protocol Specification (Revised), Section 4.2";
}

/*
 * Identities
 */
identity rp-sm {
  base pim-rp:rp-mode;
  description
```

```

    "SM (Sparse Mode).";
  }

  /*
   * Groupings
   */
  grouping static-rp-sm-container {
    description
      "Grouping that contains SM attributes for static RPs.";
    container sm {
      presence
        "Indicates support for PIM-SM.";
      description
        "PIM-SM configuration data.";

      uses pim-rp:static-rp-attributes;
    } // sm
  } // static-rp-sm-container

  /*
   * Configuration data nodes
   */
  augment "/rt:routing/rt:control-plane-protocols/pim-base:pim/"
    + "pim-base:address-family" {
    description
      "PIM-SM augmentation.";

    container sm {
      description
        "PIM-SM configuration data.";

      container asm {
        description
          "ASM (Any-Source Multicast) attributes.";

        container anycast-rp {
          presence
            "Present to enable an Anycast-RP (Rendezvous Point).";
          description
            "Anycast-RP attributes.";

          list ipv4-anycast-rp {
            when "../..../pim-base:address-family = 'rt:ipv4'" {
              description
                "Only applicable to an IPv4 address family.";
            }
            key "anycast-address rp-address";
            description
              "A list of IPv4 Anycast-RP settings. Only applicable
               when 'pim-base:address-family' is IPv4.";
            leaf anycast-address {
              type inet:ipv4-address;
              description
                "IP address of the Anycast-RP set. This IP address
                 is used by the multicast groups or sources to join
                 or register.";
            }
          }
        }
      }
    }
  }

```

```
    leaf rp-address {
      type inet:ipv4-address;
      description
        "IP address of the router configured with an
        Anycast-RP. This is the IP address where the
        Register messages are forwarded.";
    }
  }
  list ipv6-anycast-rp {
    when "../..../pim-base:address-family = 'rt:ipv6'" {
      description
        "Only applicable to an IPv6 address family.";
    }
    key "anycast-address rp-address";
    description
      "A list of IPv6 Anycast-RP settings. Only applicable
      when 'pim-base:address-family' is IPv6.";
    leaf anycast-address {
      type inet:ipv6-address;
      description
        "IP address of the Anycast-RP set. This IP address
        is used by the multicast groups or sources to join
        or register.";
    }
    leaf rp-address {
      type inet:ipv6-address;
      description
        "IP address of the router configured with an
        Anycast-RP. This is the IP address where the
        Register messages are forwarded.";
    }
  }
}

container spt-switch {
  description
    "SPT (Shortest Path Tree) switching attributes.";
  container infinity {
    if-feature spt-switch-infinity;
    presence
      "Present if the SPT switchover threshold is set to
      infinity, according to the policy specified below.";
    description
      "The receiver's DR (Designated Router) never triggers
      switchover from the RPT to the SPT.";
    leaf policy-name {
      if-feature spt-switch-policy;
      type string;
      description
        "The string value is the name to uniquely identify a
        policy that contains one or more policy rules used
        to accept or reject certain multicast groups.
        The groups accepted by this policy have the SPT
        switchover threshold set to infinity, meaning that
        they will stay on the shared tree forever.
        If a policy is not specified, the entire multicast
        address space is accepted."
    }
  }
}
```



```

        The definition of such a policy is outside the scope
        of this document.";
    }
  } // infinity
} // asm

container ssm {
  presence
    "Present to enable SSM (Source-Specific Multicast).";
  description
    "SSM attributes.";

  leaf range-policy {
    type string;
    description
      "The string value is the name to uniquely identify a
      policy that contains one or more policy rules used
      to accept or reject certain multicast groups.
      The groups accepted by this policy define the multicast
      group range used by SSM.
      If a policy is not specified, the default SSM multicast
      group range is used.
      The default SSM multicast group range is 232.0.0.0/8
      for IPv4 and ff3x::/96 for IPv6, where x represents any
      valid scope identifier.
      The definition of such a policy is outside the scope
      of this document.";
    reference
      "RFC 4607: Source-Specific Multicast for IP";
  }
} // ssm
} // sm
} // augment

augment "/rt:routing/rt:control-plane-protocols/pim-base:pim/"
+ "pim-base:interfaces/pim-base:interface/"
+ "pim-base:address-family" {
  description
    "PIM-SM augmentation.";

  container sm {
    presence "Present to enable PIM-SM.";
    description
      "PIM-SM configuration data.";

    leaf passive {
      type empty;
      description
        "Specifies that no PIM messages are sent or accepted on
        this PIM interface, but the interface can be included in a
        multicast forwarding entry.";
    }
  } // sm
} // augment

augment "/rt:routing/rt:control-plane-protocols/pim-base:pim/"
+ "pim-base:address-family/pim-rp:rp/"

```

```
+ "pim-rp:static-rp/pim-rp:ipv4-rp" {
  description
    "PIM-SM augmentation.";

  uses static-rp-sm-container;
} // augment

augment "/rt:routing/rt:control-plane-protocols/pim-base:pim/"
+ "pim-base:address-family/pim-rp:rp/"
+ "pim-rp:static-rp/pim-rp:ipv6-rp" {
  description
    "PIM-SM augmentation.";

  uses static-rp-sm-container;
} // augment
}

<CODE ENDS>
```

## 6.4. PIM-DM Module

This module references [\[RFC3973\]](#) and [\[RFC8349\]](#).

```
<CODE BEGINS> file "ietf-pim-dm@2021-09-03.yang"

module ietf-pim-dm {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-pim-dm";
  prefix pim-dm;

  import ietf-routing {
    prefix "rt";
    reference
      "RFC 8349: A YANG Data Model for Routing Management (NMDA
      Version)";
  }

  import ietf-pim-base {
    prefix "pim-base";
    reference
      "RFC 9128: A YANG Data Model for Protocol Independent
      Multicast (PIM)";
  }

  organization
    "IETF PIM Working Group";

  contact
    "WG Web: <https://datatracker.ietf.org/wg/pim/>
    WG List: <mailto:pim@ietf.org>

    Editor: Xufeng Liu
           <mailto:xufeng.liu.ietf@gmail.com>

    Editor: Pete McAllister
           <mailto:pete.mcallister@metaswitch.com>

    Editor: Anish Peter
           <mailto:anish.ietf@gmail.com>

    Editor: Mahesh Sivakumar
           <mailto:sivakumar.mahesh@gmail.com>

    Editor: Yisong Liu
           <mailto:liuyisong@huawei.com>

    Editor: Fangwei Hu
           <mailto:hu.fangwei@zte.com.cn>";

  description
    "This YANG module defines a PIM (Protocol Independent Multicast)
    DM (Dense Mode) model.

    Copyright (c) 2021 IETF Trust and the persons identified as
    authors of the code. All rights reserved.

    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject to
    the license terms contained in, the Simplified BSD License set
    forth in Section 4.c of the IETF Trust's Legal Provisions
```

```
Relating to IETF Documents
(https://trustee.ietf.org/license-info).
```

This version of this YANG module is part of RFC 9128; see the RFC itself for full legal notices.";

```
revision 2021-09-03 {
  description
    "Initial revision.";
  reference
    "RFC 3973: Protocol Independent Multicast - Dense Mode
    (PIM-DM): Protocol Specification (Revised)
    RFC 9128: A YANG Data Model for Protocol Independent
    Multicast (PIM)";
}

/*
 * Configuration data nodes
 */

augment "/rt:routing/rt:control-plane-protocols/"
+ "pim-base:pim/pim-base:address-family" {
  description
    "PIM-DM augmentation.";

  container dm {
    presence "Present to enable PIM-DM.";
    description
      "PIM-DM configuration data.";
  } // dm
} // augment

augment "/rt:routing/rt:control-plane-protocols/"
+ "pim-base:pim/pim-base:interfaces/pim-base:interface/"
+ "pim-base:address-family" {
  description
    "PIM-DM augmentation to 'pim-base:interface.'";

  container dm {
    presence "Present to enable PIM-DM.";
    description
      "PIM-DM configuration data.";
  } // dm
} // augment
}

<CODE ENDS>
```

## 6.5. PIM-BIDIR Module

This module references [\[RFC5015\]](#), [\[RFC6991\]](#), [\[RFC8294\]](#), [\[RFC8343\]](#), and [\[RFC8349\]](#).

```
<CODE BEGINS> file "ietf-pim-bidir@2021-09-03.yang"

module ietf-pim-bidir {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-pim-bidir";
  prefix pim-bidir;

  import ietf-inet-types {
    prefix "inet";
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-routing-types {
    prefix "rt-types";
    reference
      "RFC 8294: Common YANG Data Types for the Routing Area";
  }

  import ietf-interfaces {
    prefix "if";
    reference
      "RFC 8343: A YANG Data Model for Interface Management";
  }

  import ietf-routing {
    prefix "rt";
    reference
      "RFC 8349: A YANG Data Model for Routing Management (NMDA
      Version)";
  }

  import ietf-pim-base {
    prefix "pim-base";
    reference
      "RFC 9128: A YANG Data Model for Protocol Independent
      Multicast (PIM)";
  }

  import ietf-pim-rp {
    prefix "pim-rp";
    reference
      "RFC 9128: A YANG Data Model for Protocol Independent
      Multicast (PIM)";
  }

  organization
    "IETF PIM Working Group";

  contact
    "WG Web: <https://datatracker.ietf.org/wg/pim/>
    WG List: <mailto:pim@ietf.org>

    Editor: Xufeng Liu
           <mailto:xufeng.liu.ietf@gmail.com>

    Editor: Pete McAllister
```

```

        <mailto:pete.mcallister@metaswitch.com>

Editor:  Anish Peter
        <mailto:anish.ietf@gmail.com>

Editor:  Mahesh Sivakumar
        <mailto:sivakumar.mahesh@gmail.com>

Editor:  Yisong Liu
        <mailto:liuyisong@huawei.com>

Editor:  Fangwei Hu
        <mailto:hu.fangwei@zte.com.cn>;

description
  "This YANG module defines a PIM (Protocol Independent Multicast)
  BIDIR (Bidirectional) mode model.

  Copyright (c) 2021 IETF Trust and the persons identified as
  authors of the code.  All rights reserved.

  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject to
  the license terms contained in, the Simplified BSD License set
  forth in Section 4.c of the IETF Trust's Legal Provisions
  Relating to IETF Documents
  (https://trustee.ietf.org/license-info).

```



```
    "BIDIR mode.";
  }

  identity df-state {
    description
      "DF (Designated Forwarder) election state type.";
    reference
      "RFC 5015: Bidirectional Protocol Independent Multicast
      (BIDIR-PIM)";
  }

  identity df-state-offer {
    base df-state;
    description
      "Initial election state.  When in the Offer state, a router
      thinks it can eventually become the winner and periodically
      generates Offer messages.";
  }

  identity df-state-lose {
    base df-state;
    description
      "Either (1) there is a different election winner or
      (2) no router on the link has a path to the RPA
      (Rendezvous Point Address).";
  }

  identity df-state-win {
    base df-state;
    description
      "The router is the acting DF without any contest.";
  }

  identity df-state-backoff {
    base df-state;
    description
      "The router is the acting DF, but another router has made a
      bid to take over.";
  }

  /*
  * Groupings
  */
  grouping static-rp-bidir-container {
    description
      "Grouping that contains BIDIR attributes for a static RP
      (Rendezvous Point).";
    container bidir {
      presence
        "Indicates support for BIDIR mode.";
      description
        "PIM-BIDIR configuration data.";

      uses pim-rp:static-rp-attributes;
    } // bidir
  } // static-rp-bidir-container

  grouping interface-df-election-state-attributes {
```

```

description
  "Grouping that contains the state attributes of a DF election
  on an interface.";
leaf interface-state {
  type identityref {
    base df-state;
  }
  description
    "Interface state with respect to the DF election.";
}
leaf up-time {
  type rt-types:timeticks64;
  description
    "The number of time ticks (hundredths of a second) since the
    current DF has been elected as the winner.";
}
leaf winner-metric {
  type uint32;
  description
    "The unicast routing metric used by the DF to reach the RP.
    The value is announced by the DF.";
}
leaf winner-metric-preference {
  type uint32;
  description
    "The preference value assigned to the unicast routing
    protocol that the DF used to obtain the route to the RP.
    The value is announced by the DF.";
}
} // interface-df-election-state-attributes

/*
 * Configuration data and operational state data nodes
 */

augment "/rt:routing/rt:control-plane-protocols/"
+ "pim-base:pim/pim-base:address-family" {
  description
    "PIM-BIDIR augmentation.";

  container bidir {
    presence "Present to enable BIDIR mode.";
    description
      "PIM-BIDIR configuration data.";
  } // bidir
} // augment

augment "/rt:routing/rt:control-plane-protocols/"
+ "pim-base:pim/pim-base:interfaces/pim-base:interface/"
+ "pim-base:address-family" {
  description
    "PIM-BIDIR augmentation.";

  container bidir {
    presence "Present to enable BIDIR mode.";
    description
      "PIM-BIDIR configuration data.";
    container df-election {

```

```

    if-feature intf-df-election;
    description
      "DF election attributes.";
    leaf offer-interval {
      type uint16;
      units milliseconds;
      default 100;
      description
        "Offer interval. Specifies the interval between
         repeated DF election messages.";
    }
    leaf backoff-interval {
      type uint16;
      units milliseconds;
      default 1000;
      description
        "This is the interval that the acting DF waits between
         receiving a better DF Offer and sending the Pass message
         to transfer DF responsibility.";
    }
    leaf offer-multiplier {
      type uint8;
      default 3;
      description
        "This is the number of transmission attempts for
         DF election messages.
         When a DF election Offer or Winner message fails to be
         received, the message is retransmitted.
         'offer-multiplier' sets the minimum number of DF
         election messages that must fail to be received for DF
         election to fail.
         If a router receives from a neighbor a better offer than
         its own, the router stops participating in the election
         for a period of 'offer-multiplier' * 'offer-interval'.
         Eventually, all routers except the best candidate stop
         sending Offer messages.";
    }
  } // df-election
} // bidir
} // augment

augment "/rt:routing/rt:control-plane-protocols/"
+ "pim-base:pim/pim-base:address-family/pim-rp:rp/"
+ "pim-rp:static-rp/pim-rp:ipv4-rp" {
  description
    "PIM-BIDIR augmentation.";

  uses static-rp-bidir-container;
} // augment

augment "/rt:routing/rt:control-plane-protocols/"
+ "pim-base:pim/pim-base:address-family/pim-rp:rp/"
+ "pim-rp:static-rp/pim-rp:ipv6-rp" {
  description
    "PIM-BIDIR augmentation.";

  uses static-rp-bidir-container;
} // augment

```

```

/*
 * Operational state data nodes
 */

augment "/rt:routing/rt:control-plane-protocols/"
+ "pim-base:pim/pim-base:address-family/pim-rp:rp" {
  description
    "PIM-BIDIR augmentation to RP state data.";

  container bidir {
    config false;
    description
      "PIM-BIDIR state data.";
    container df-election {
      description
        "DF election data.";
      list ipv4-rp {
        when "../..../pim-base:address-family = 'rt:ipv4'" {
          description
            "Only applicable to an IPv4 address family.";
        }
        key "rp-address";
        description
          "A list of IPv4 RP addresses.";
        leaf rp-address {
          type inet:ipv4-address;
          description
            "The address of the RP.";
        }
      } // ipv4-rp
      list ipv6-rp {
        when "../..../pim-base:address-family = 'rt:ipv6'" {
          description
            "Only applicable to an IPv6 address family.";
        }
        key "rp-address";
        description
          "A list of IPv6 RP addresses.";
        leaf rp-address {
          type inet:ipv6-address;
          description
            "The address of the RP.";
        }
      } // ipv6-rp
    } // df-election

    container interface-df-election {
      description
        "Interface DF election data.";
      list ipv4-rp {
        when "../..../pim-base:address-family = 'rt:ipv4'" {
          description
            "Only applicable to an IPv4 address family.";
        }
        key "rp-address interface-name";
        description
          "A list of IPv4 RP addresses.";
      }
    }
  }
}

```

```

    leaf rp-address {
      type inet:ipv4-address;
      description
        "The address of the RP.";
    }
    leaf interface-name {
      type if:interface-ref;
      description
        "The name of the interface for which the DF state is
        being maintained.";
    }
    leaf df-address {
      type inet:ipv4-address;
      description
        "The address of the elected DF, which is the winner of
        the DF election process.";
    }
    uses interface-df-election-state-attributes;
  } // ipv4-rp
  list ipv6-rp {
    when "../..../pim-base:address-family = 'rt:ipv6'" {
      description
        "Only applicable to an IPv6 address family.";
    }
    key "rp-address interface-name";
    description
      "A list of IPv6 RP addresses.";
    leaf rp-address {
      type inet:ipv6-address;
      description
        "The address of the RP.";
    }
    leaf interface-name {
      type if:interface-ref;
      description
        "The address of the RP.";
    }
    leaf df-address {
      type inet:ipv6-address;
      description
        "DF address.";
    }
    uses interface-df-election-state-attributes;
  } // ipv6-rp
} // interface-df-election
}
} // augment

augment "/rt:routing/rt:control-plane-protocols/"
+ "pim-base:pim/pim-base:interfaces/pim-base:interface/"
+ "pim-base:address-family/pim-base:neighbors/"
+ "pim-base:ipv4-neighbor" {
  description
    "PIM-BIDIR augmentation to the IPv4 neighbor state data.";
  leaf bidir-capable {
    type boolean;
    description
      "'true' if the neighbor is using the Bidirectional Capable

```

```
        option in the last Hello message.";
    }
} // augment

augment "/rt:routing/rt:control-plane-protocols/"
+ "pim-base:pim/pim-base:interfaces/pim-base:interface/"
+ "pim-base:address-family/pim-base:neighbors/"
+ "pim-base:ipv6-neighbor" {
    description
        "PIM-BIDIR augmentation to the IPv6 neighbor state data.";
    leaf bidir-capable {
        type boolean;
        description
            "'true' if the neighbor is using the Bidirectional Capable
            option in the last Hello message.";
    }
} // augment
}

<CODE ENDS>
```

## 7. Security Considerations

The YANG modules specified in this document define a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in these YANG modules that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

### pim-base:graceful-restart

This subtree specifies the configuration for PIM graceful restart at the global level on a device. Modifying the configuration can cause temporary interruption to the multicast routing during restart.

### pim-base:address-family/pim-base:graceful-restart

This subtree specifies the per address family configuration for PIM graceful restart on a device. Modifying the configuration can cause temporary interruption to the multicast routing during restart.

pim-base:address-family/pim-rp:pim-rp:rp

This subtree specifies the configuration for the PIM Rendezvous Point (RP) on a device. Modifying the configuration can cause RP malfunctions.

pim-base:address-family/pim-sm:sm

This subtree specifies the configuration for PIM Sparse Mode (PIM-SM) on a device. Modifying the configuration can cause multicast traffic to be disabled or rerouted in PIM-SM.

pim-base:address-family/pim-dm:dm

This subtree specifies the configuration for PIM Dense Mode (PIM-DM) on a device. Modifying the configuration can cause multicast traffic to be disabled or rerouted in PIM-DM.

pim-base:address-family/pim-bidir:bidir

This subtree specifies the configuration for PIM Bidirectional Mode (PIM-BIDIR) on a device. Modifying the configuration can cause multicast traffic to be disabled or rerouted in PIM-BIDIR.

pim-base:interfaces

This subtree specifies the configuration for the PIM interfaces on a device. Modifying the configuration can cause the PIM protocol to get insufficient or incorrect information.

These subtrees are all under "/rt:routing/rt:control-plane-protocols/pim-base:pim".

Unauthorized access to any data node of these subtrees can adversely affect the multicast routing subsystem of both the local device and the network. This may lead to network malfunctions, delivery of packets to inappropriate destinations, and other problems.

Some of the readable data nodes in these YANG modules may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

/rt:routing/rt:control-plane-protocols/pim-base:pim

Unauthorized access to any data node of the above subtree can disclose the operational state information of PIM on this device.

## 8. IANA Considerations

IANA has registered the following namespace URIs in the "IETF XML Registry" [[RFC3688](#)]:

URI: urn:ietf:params:xml:ns:yang:ietf-pim-base

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-pim-bidir

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-pim-dm

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-pim-rp

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-pim-sm

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

IANA has registered the following YANG modules in the "YANG Module Names" registry [[RFC6020](#)]:

Name: ietf-pim-base

Namespace: urn:ietf:params:xml:ns:yang:ietf-pim-base

Prefix: pim-base

Reference: RFC 9128

Name: ietf-pim-bidir

Namespace: urn:ietf:params:xml:ns:yang:ietf-pim-bidir

Prefix: pim-bidir

Reference: RFC 9128

Name: ietf-pim-dm

Namespace: urn:ietf:params:xml:ns:yang:ietf-pim-dm

Prefix: pim-dm

Reference: RFC 9128

Name: ietf-pim-rp

Namespace: urn:ietf:params:xml:ns:yang:ietf-pim-rp

Prefix: pim-rp

Reference: RFC 9128

Name: ietf-pim-sm

Namespace: urn:ietf:params:xml:ns:yang:ietf-pim-sm

Prefix: pim-sm

Reference: RFC 9128



## 9. References

### 9.1. Normative References

- [RFC3569] Bhattacharyya, S., Ed., "An Overview of Source-Specific Multicast (SSM)", RFC 3569, DOI 10.17487/RFC3569, July 2003, <<https://www.rfc-editor.org/info/rfc3569>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC3973] Adams, A., Nicholas, J., and W. Siadak, "Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)", RFC 3973, DOI 10.17487/RFC3973, January 2005, <<https://www.rfc-editor.org/info/rfc3973>>.
- [RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, DOI 10.17487/RFC4607, August 2006, <<https://www.rfc-editor.org/info/rfc4607>>.
- [RFC4610] Farinacci, D. and Y. Cai, "Anycast-RP Using Protocol Independent Multicast (PIM)", RFC 4610, DOI 10.17487/RFC4610, August 2006, <<https://www.rfc-editor.org/info/rfc4610>>.
- [RFC5015] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, DOI 10.17487/RFC5015, October 2007, <<https://www.rfc-editor.org/info/rfc5015>>.
- [RFC5059] Bhaskar, N., Gall, A., Lingard, J., and S. Venaas, "Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM)", RFC 5059, DOI 10.17487/RFC5059, January 2008, <<https://www.rfc-editor.org/info/rfc5059>>.
- [RFC5060] Sivaramu, R., Lingard, J., McWalter, D., Joshi, B., and A. Kessler, "Protocol Independent Multicast MIB", RFC 5060, DOI 10.17487/RFC5060, January 2008, <<https://www.rfc-editor.org/info/rfc5060>>.
- [RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <<https://www.rfc-editor.org/info/rfc6020>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.

- 
- [RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <<https://www.rfc-editor.org/info/rfc7761>>.
  - [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
  - [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
  - [RFC8294] Liu, X., Qu, Y., Lindem, A., Hopps, C., and L. Berger, "Common YANG Data Types for the Routing Area", RFC 8294, DOI 10.17487/RFC8294, December 2017, <<https://www.rfc-editor.org/info/rfc8294>>.
  - [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <<https://www.rfc-editor.org/info/rfc8341>>.
  - [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.
  - [RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <<https://www.rfc-editor.org/info/rfc8343>>.
  - [RFC8349] Lhotka, L., Lindem, A., and Y. Qu, "A YANG Data Model for Routing Management (NMDA Version)", RFC 8349, DOI 10.17487/RFC8349, March 2018, <<https://www.rfc-editor.org/info/rfc8349>>.
  - [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
  - [RFC9127] Rahman, R., Ed., Zheng, L., Ed., Jethanandani, M., Ed., Pallagatti, S., and G. Mirsky, "YANG Data Model for Bidirectional Forwarding Detection (BFD)", RFC 9127, DOI 10.17487/RFC9127, September 2021, <<https://www.rfc-editor.org/info/rfc9127>>.

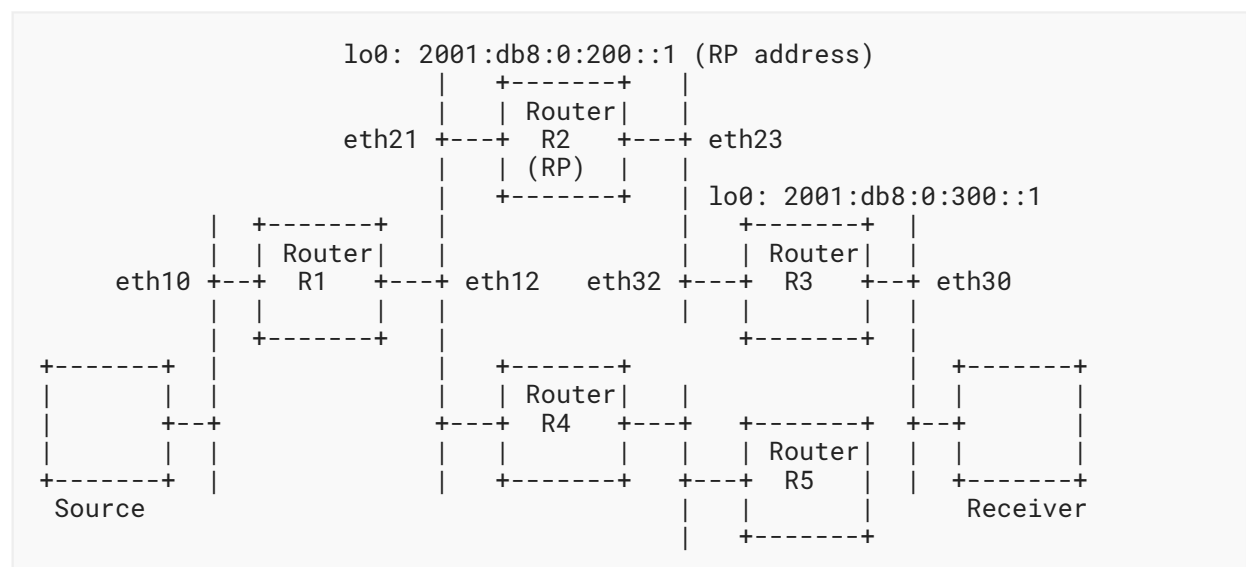
## 9.2. Informative References

- [RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, DOI 10.17487/RFC3376, October 2002, <<https://www.rfc-editor.org/info/rfc3376>>.
- [RFC3618] Fenner, B., Ed. and D. Meyer, Ed., "Multicast Source Discovery Protocol (MSDP)", RFC 3618, DOI 10.17487/RFC3618, October 2003, <<https://www.rfc-editor.org/info/rfc3618>>.

- [RFC3810] Vida, R., Ed. and L. Costa, Ed., "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, DOI 10.17487/RFC3810, June 2004, <<https://www.rfc-editor.org/info/rfc3810>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://www.rfc-editor.org/info/rfc5880>>.
- [RFC6388] Wijnands, IJ., Ed., Minei, I., Ed., Kompella, K., and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths", RFC 6388, DOI 10.17487/RFC6388, November 2011, <<https://www.rfc-editor.org/info/rfc6388>>.
- [RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <<https://www.rfc-editor.org/info/rfc7951>>.
- [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <<https://www.rfc-editor.org/info/rfc8340>>.
- [RFC8407] Bierman, A., "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", BCP 216, RFC 8407, DOI 10.17487/RFC8407, October 2018, <<https://www.rfc-editor.org/info/rfc8407>>.
- [RFC8706] Ginsberg, L. and P. Wells, "Restart Signaling for IS-IS", RFC 8706, DOI 10.17487/RFC8706, February 2020, <<https://www.rfc-editor.org/info/rfc8706>>.

## Appendix A. Data Tree Example

This appendix contains an example of an instance data tree, in JSON encoding [RFC7951], containing both configuration data and state data.



The configuration instance data tree for Router R3 in the above figure could be as follows:

```

{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "lo0",
        "description": "R3 loopback interface.",
        "type": "iana-if-type:softwareLoopback",
        "ietf-ip:ipv6": {
          "address": [
            {
              "ip": "2001:db8:0:300::1",
              "prefix-length": 64
            }
          ]
        }
      },
      {
        "name": "eth30",
        "description": "An interface connected to the receiver.",
        "type": "iana-if-type:ethernetCsmacd",
        "ietf-ip:ipv6": {
          "forwarding": true
        }
      },
      {
        "name": "eth32",
        "description": "An interface connected to the RP (R2).",
        "type": "iana-if-type:ethernetCsmacd",
        "ietf-ip:ipv6": {
          "forwarding": true
        }
      }
    ]
  },
  "ietf-routing:routing": {
    "router-id": "203.0.113.3",
    "control-plane-protocols": {
      "ietf-pim-base:pim": {
        "address-family": [
          {
            "address-family": "ietf-routing:ipv6",
            "ietf-pim-rp:rp": {
              "static-rp": {
                "ipv6-rp": [
                  {
                    "rp-address": "2001:db8:0:200::1",
                    "ietf-pim-sm:sm": {
                      }
                  }
                ]
              }
            }
          }
        ]
      }
    }
  },
  "interfaces": {
    "interface": [
      {

```

```
    "name": "lo0",
    "address-family": [
      {
        "address-family": "ietf-routing:ipv6",
        "hello-interval": "infinity",
        "ietf-pim-sm:sm": {
        }
      }
    ]
  },
  {
    "name": "eth30",
    "address-family": [
      {
        "address-family": "ietf-routing:ipv6",
        "ietf-pim-sm:sm": {
        }
      }
    ]
  },
  {
    "name": "eth32",
    "address-family": [
      {
        "address-family": "ietf-routing:ipv6",
        "ietf-pim-sm:sm": {
        }
      }
    ]
  }
]
}
}
}
}
}
```

The corresponding operational state data for Router R3 could be as follows:

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "lo0",
        "description": "R3 loopback interface.",
        "type": "iana-if-type:softwareLoopback",
        "phys-address": "00:00:5e:00:53:03",
        "oper-status": "up",
        "statistics": {
          "discontinuity-time": "2018-01-23T12:34:56-05:00"
        },
        "ietf-ip:ipv6": {
          "mtu": 1500,
          "address": [
            {
              "ip": "2001:db8:0:300::1",
              "prefix-length": 64,
              "origin": "static",
              "status": "preferred"
            },
            {
              "ip": "fe80::200:5eff:fe00:5303",
              "prefix-length": 64,
              "origin": "link-layer",
              "status": "preferred"
            }
          ],
          "neighbor": [
            ]
        }
      },
      {
        "name": "eth30",
        "description": "An interface connected to the receiver.",
        "type": "iana-if-type:ethernetCsmacd",
        "phys-address": "00:00:5e:00:53:30",
        "oper-status": "up",
        "statistics": {
          "discontinuity-time": "2018-01-23T12:34:56-05:00"
        },
        "ietf-ip:ipv6": {
          "forwarding": true,
          "mtu": 1500,
          "address": [
            {
              "ip": "fe80::200:5eff:fe00:5330",
              "prefix-length": 64,
              "origin": "link-layer",
              "status": "preferred"
            }
          ],
          "neighbor": [
            ]
        }
      }
    ]
  }
}
```



```

    "name": "eth32",
    "description": "An interface connected to the RP (R2).",
    "type": "iana-if-type:ethernetCsmacd",
    "phys-address": "00:00:5e:00:53:32",
    "oper-status": "up",
    "statistics": {
      "discontinuity-time": "2018-01-23T12:34:56-05:00"
    },
    "ietf-ip:ipv6": {
      "forwarding": true,
      "mtu": 1500,
      "address": [
        {
          "ip": "fe80::200:5eff:fe00:5332",
          "prefix-length": 64,
          "origin": "link-layer",
          "status": "preferred"
        }
      ],
      "neighbor": [
        {
          "ip": "fe80::200:5eff:fe00:5323",
          "link-layer-address": "00:00:5e:00:53:23",
          "origin": "dynamic",
          "is-router": [null],
          "state": "reachable"
        }
      ]
    }
  ],
},
"ietf-routing:routing": {
  "router-id": "203.0.113.1",
  "interfaces": {
    "interface": [
      "lo0",
      "eth30",
      "eth32"
    ]
  }
},
"control-plane-protocols": {
  "ietf-pim-base:pim": {
    "address-family": [
      {
        "address-family": "ietf-routing:ipv6",
        "statistics": {
          "discontinuity-time": "2018-01-23T12:34:56-05:00"
        },
        "topology-tree-info": {
          "ipv6-route": [
            {
              "group": "ff06::1",
              "source-address": "*",
              "is-rpt": true,
              "expiration": 16,
              "incoming-interface": "eth32",
              "is-spt": false,
            }
          ]
        }
      }
    ]
  }
}

```

```

    "mode": "pim-asm",
    "msdp-learned": false,
    "rp-address": "2001:db8:0:200::1",
    "rpf-neighbor": "fe80::200:5eff:fe00:5323",
    "up-time": 123400,
    "outgoing-interface": [
      {
        "name": "eth30",
        "expiration": 36,
        "up-time": 223400,
        "jp-state": "join"
      }
    ]
  },
  {
    "group": "ff06::1",
    "source-address": "2001:db8:1:1::100",
    "is-rpt": false,
    "expiration": 8,
    "incoming-interface": "eth32",
    "is-spt": true,
    "mode": "pim-asm",
    "msdp-learned": false,
    "rp-address": "2001:db8:0:200::1",
    "rpf-neighbor": "fe80::200:5eff:fe00:5323",
    "up-time": 5200,
    "outgoing-interface": [
      {
        "name": "eth30",
        "expiration": 6,
        "up-time": 5600,
        "jp-state": "join"
      }
    ]
  }
],
"ietf-pim-rp:rp": {
  "static-rp": {
    "ipv6-rp": [
      {
        "rp-address": "2001:db8:0:200::1",
        "ietf-pim-sm:sm": {
        }
      }
    ]
  }
},
"rp-list": {
  "ipv6-rp": [
    {
      "rp-address": "2001:db8:0:200::1",
      "mode": "ietf-pim-sm:rp-sm",
      "info-source-type": "static",
      "up-time": 323400,
      "expiration": "not-set"
    }
  ]
},
},

```

```

        "rp-mappings": {
          "ipv6-rp": [
            {
              "group-range": "ff06::1/128",
              "rp-address": "2001:db8:0:200::1",
              "up-time": 123400,
              "expiration": "36"
            }
          ]
        }
      ],
    },
    "interfaces": {
      "interface": [
        {
          "name": "lo0",
          "address-family": [
            {
              "address-family": "ietf-routing:ipv6",
              "hello-interval": "infinity",
              "ietf-pim-sm:sm": {
                "oper-status": "up",
                "gen-id": 103689,
                "hello-expiration": "infinity",
                "ipv6": {
                  "address": [
                    "fe80::200:5eff:fe00:5303"
                  ],
                  "dr-address": "fe80::200:5eff:fe00:5303"
                }
              },
              "neighbors": {
                "ipv6-neighbor": [
                ]
              }
            }
          ]
        },
        {
          "name": "eth30",
          "address-family": [
            {
              "address-family": "ietf-routing:ipv6",
              "ietf-pim-sm:sm": {
                "oper-status": "up",
                "gen-id": 203689,
                "hello-expiration": 18,
                "ipv6": {
                  "address": [
                    "fe80::200:5eff:fe00:5330"
                  ],
                  "dr-address": "fe80::200:5eff:fe00:5330"
                }
              },
              "neighbors": {
                "ipv6-neighbor": [
                ]
              }
            }
          ]
        }
      ]
    }
  }
}

```

```
    }
  }
]
},
{
  "name": "eth32",
  "address-family": [
    {
      "address-family": "ietf-routing:ipv6",
      "ietf-pim-sm:sm": {
      },
      "oper-status": "up",
      "gen-id": 303689,
      "hello-expiration": 21,
      "ipv6": {
        "address": [
          "fe80::200:5eff:fe00:5332"
        ],
        "dr-address": "fe80::200:5eff:fe00:5332"
      },
      "neighbors": {
        "ipv6-neighbor": [
          {
            "address": "fe80::200:5eff:fe00:5323",
            "expiration": 28,
            "dr-priority": 1,
            "gen-id": 102,
            "lan-prune-delay": {
              "present": false
            },
            "up-time": 323500
          }
        ]
      }
    }
  ]
}
}
}
}
}
}
}
}
}
}
```

## Acknowledgments

The authors would like to thank Steve Baillargeon, Guo Feng, Robert Kebler, Tanmoy Kundu, and Stig Venaas for their valuable contributions.

## Authors' Addresses

**Xufeng Liu**

Volta Networks

Email: [xufeng.liu.ietf@gmail.com](mailto:xufeng.liu.ietf@gmail.com)**Pete McAllister**

Metaswitch Networks

100 Church Street

Enfield

EN2 6BQ

United Kingdom

Email: [pete.mcallister@metaswitch.com](mailto:pete.mcallister@metaswitch.com)**Anish Peter**

Individual

Email: [anish.ietf@gmail.com](mailto:anish.ietf@gmail.com)**Mahesh Sivakumar**

Juniper Networks

1133 Innovation Way

Sunnyvale, California

United States of America

Email: [sivakumar.mahesh@gmail.com](mailto:sivakumar.mahesh@gmail.com)**Yisong Liu**

Huawei Technologies

Huawei Administration Building

Longgang

Guangdong, 518129

China

Email: [liuyisong@huawei.com](mailto:liuyisong@huawei.com)**Fangwei Hu**

ZTE Corporation

889 Bibo Road

Shanghai

Shanghai, 201203

China

Email: [hu.fangwei@zte.com.cn](mailto:hu.fangwei@zte.com.cn)