PIM Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 4, 2015

H. Asaeda
NICT

August 31, 2014

IGMP/MLD-Based Explicit Membership Tracking Function for Multicast Routers
draft-ietf-pim-explicit-tracking-10

Abstract

This document describes the IGMP/MLD-based explicit membership tracking function for multicast routers and IGMP/MLD proxy devices supporting IGMPv3/MLDv2. The explicit membership tracking function contributes to saving network resources and shortening leave latency.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 4, 2015.

Table of Contents

1. Introduction

The Internet Group Management Protocol (IGMP) version 3 [2] for IPv4 and the Multicast Listener Discovery Protocol (MLD) version 2 [3] for IPv6 are the standard protocols used by member hosts and multicast routers. Lightweight IGMPv3 and Lightweight MLDv2 (or LW-IGMPv3 and LW-MLDv2) [4] are subsets of the standard IGMPv3 and MLDv2.

When a host starts/finishes listening to particular multicast channels, it sends IGMP/MLD State-Change Report messages specifying the corresponding channel information as the join/leave request to its upstream router (i.e., an adjacent multicast router or IGMP/MLD proxy device [8]). The "unsolicited" report messages are sent only when the host joins/leaves the channels. Since IGMP/MLD are non-reliable protocols, unsolicited report messages may be lost or may not reach upstream routers. To alleviate this problem, unsolicited report messages are retransmitted a number of times according to the value of the [Robustness Variable] defined in [2][3].

In addition, a querier router periodically sends IGMP/MLD General Query messages every General Query timer interval (i.e. [Query Interval] value defined in [2][3]). Upon receiving the query messages, the member hosts reply with "solicited" report messages. Routers then keep their membership state information up to date. However, this approach still does not guarantee that the membership state is always perfectly synchronized. To minimize the possibility of having outdated membership information, routers may shorten the periodic General Query timer interval. Unfortunately, this increases the number of transmitted solicited report messages and induces

network congestion. And the greater the amount of network congestion, the greater the potential for IGMP/MLD report messages being lost and the membership state information being outdated in the router.

IGMPv3 [2], MLDv2 [3], and these lightweight protocols [4] can provide the ability to keep track of the downstream (adjacent) multicast membership state in multicast routers, yet the specifications are not clearly given. This document describes the "IGMP/MLD-based explicit member tracking function" for multicast routers and a way for routers to implement the function. By enabling this explicit tracking function, routers can keep track of the downstream multicast membership state.

The explicit tracking function is important for the scalability of multicast networks, and might be widely implemented in modern multicast routers. However, it could seriously break IGMP/MLD communications if not implemented or configured correctly. For example, the explicit tracking function is useful for shortening leave latency, while wrong implementations or configurations in routers on a LAN may not work properly that the operators expect.

This document aims to get the explicit tracking function correctly. Regarding the way for shortening leave latency, it specifies the way to do it by tuning the values used by IGMPv3 [2], MLDv2 [3], and their lightweight version protocols [4], or by enabling the mechanism called "specific query suppression" with a robust link state. The latter mechanism does not make the router send any specific query message(s) and immediately leave the group or sources when the sole member has left according to its membership state information.

This document describes the risk of having wrong membership state as well. The explicit tracking function does not change the reliability of the message transmission. The list of tracked member hosts may be outdated in the router because of host departure from the network without sending State-Change Report messages or loss of such messages due to network congestion. This document guides for setting up appropriate values or mechanisms used with the explicit tracking function in routers.

The explicit tracking function potentially requires a large amount of memory so that routers keep all membership states. Particularly when a router needs to maintain a large number of member hosts, this resource requirement might be sensitive. As the security consideration, this document describes that operators may decide to disable this function when their routers have insufficient memory resources, despite the benefits.

The explicit tracking function does not change message formats used by IGMPv3 [2], MLDv2 [3], and their lightweight version protocols [4]; nor does it change a multicast data sender's and receiver's behavior.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [1].

3. Membership State Information

A router enabling the explicit tracking function maintains the "membership state information". When a multicast router receives a Current-State or State-Change Report message, it creates or modifies this membership state information to maintain the membership state up to date.

The membership state information consists of the following information:

(S, G, number of receivers, (receiver records))

where "S" denotes source address, "G" denotes group or multicast address, and each receiver record is of the form:

(IGMP/MLD membership/listener report sender's address)

In the state information, each S and G indicates a single IPv4/IPv6 address. S is set to "All sources" for Any-Source Multicast (ASM) communication (i.e., (*,G) join reception). In order to simplify the implementation, Lightweight-IGMPv3/MLDv2 [4] do not keep the state of (S,G) joined with EXCLUDE filter mode; if a router receives an (S,G) join/leave request with EXCLUDE filter mode from the downstream hosts, the router translates the request to a (*,G) join state/leave request and records the state and the receivers' addresses in the maintained membership state information.

The membership state information must be identified properly even though a receiver (i.e., IGMP/MLD Report sender) sends the identical report messages multiple times. And the maintained membership state information will be flushed when the router reboots or restarts the multicast routing processes.

4. Specific Query Suppression

In accordance with [2] and [3], when a router receives the State-Change Report and needs to confirm whether any hosts are still interested in a channel or not, the router sends the corresponding Group-Specific or Group-and-Source Specific Query messages as defined in Section 6.4.2 of [2] and Section 7.4.2 of [3]. The queries sent by actions defined in these sections need to be transmitted [Last Member Query Count] (LMQC) or [Last Listener Query Count] (LLQC) times, once every [Last Member Query Interval] (LMQI) or [Last Listener Query Interval] (LLQI), in order to confirm the sole member. (The default values for LMQI/LLQI defined in [2][3] are 1 second. The default values for LMQC/LLQC are the [Robustness Variable] value whose default value is 2.) All member hosts joining the identical channel then reply with their own states after acquiring these query messages. However, transmitting a large number of IGMP/MLD Report messages consumes network resources, and this may pose a particular problem especially when many hosts joining the identical channel send these reports simultaneously.

The explicit tracking function provides a mechanism called "specific query suppression". With the specific query suppression, regardless of the LMQC/LLQC values, if the router receives one or more replies from the downstream member(s), it SHOULD stop (i.e., cancel) retransmitting the specific query message(s) for the specified source and/or group. It reduces the number of Group-Specific or Group-and-Source Specific Query messages transmitted from a router, and in turn reduces the number of Current-State Report messages transmitted from member hosts. This contributes to saving network resources.

The specific query suppression MAY define an option called "robust link state". A router enabling the specific query suppression with a robust link state does not send any specific query message(s) and immediately leave the group or sources when the sole member has left according to its membership state information. The specific query suppression with a robust link state hence does not rely on LMQC/LLQC and LMQI/LLQI values. This contributes to shortening leave latency described in Section 5. However, this behavior requires that the router perfectly tracks all member hosts. (See a risk of wrong membership expectation described in Section 6.)

Note that the default behavior of the router that supports the explicit tracking function SHOULD disable this specific query suppression, in order to avoid the risk caused by the wrong membership expectation or by the case in which multiple multicast routers exist on a LAN and the querier router is not the forwarder router. The former case is described in Section 6. For the latter case, when the querier suppresses the specific query message

transmission, and expects that the State-Change Report sender is not the sole member of the channel, it does not send the specific query. Then the routers (including the forwarder) on the same LAN do not receive a Current-State Report message from the corresponding member hosts. The forwarder in this case may prune the routing path, although there are other member hosts subscribing to the channel on the LAN.

5. Shortening Leave Latency

A router enabling the explicit tracking function can shorten leave latencies by tuning the following values; [Last Member Query Count] (LMQC), [Last Listener Query Count] (LLQC), [Last Member Query Interval] (LMQI), [Last Listener Query Interval] (LLQI), and [Robustness Variable] values.

The [Last Member Query Interval] (LMQI) and [Last Listener Query Interval] (LLQI) values defined in the standard specifications [2][3] specify the maximum time allowed for a member host to send a responding Report. The [Last Member Query Count] (LMQC) and [Last Listener Query Count] (LLQC) are the number of Group-Specific Queries or Group-and-Source Specific Queries sent before the router assumes there are no local members. The [Last Member Query Time] (LMQT) and [Last Listener Query Time] (LLQT) values are the total time the router should wait for a report after the Querier has sent the first query.

The default values for LMQI/LLQI defined in [2][3] are 1 second, yet, for a router enabling the explicit tracking function, the LMQI/LLQI may be set to 1 second or shorter. As well, the default values for LMQC/LLQC are the [Robustness Variable] value whose default value is 2, yet the LMQC/LLQC may be set to 1 for the router. Smaller LMQC/ LLQC values give shorter LMQT/LLQT, which shorten the leave latencies.

Furthermore, if operators are confident that their link is fairly robust (e.g., the [Robustness Variable] value is appropriately configured so that the chances of unsolicited messages being lost are sufficiently low), and if the querier router always acts as the forwarder router for all multicast channels in the LAN, they will set smaller LMQC/LLQC and shorter LMQI/LLQI (and hence shorter LMQT/LLQT) with the specific query suppression, or enable the specific query suppression with a robust link state (Section 4) for their routers.

Note that setting smaller LMQC/LLQC and shorter LMQI/LLQI values or adopting the specific query suppression with a robust link state poses the risk of wrong membership state described in Section 6.

Operators setting these values or enabling that mechanism must recognize this tradeoff.

6. Risk of Wrong Membership State

There are possibilities that a router's membership expectation is inconsistent due to an outdated membership state. For example, (1) a router expects that more than one corresponding member host exists on its LAN, but in fact no member host exists for that multicast channel, or (2) a router expects that no corresponding member host exists on its LAN, but in fact one or more than one member host exists for that multicast channel.

The first case may occur in an environment where the sole member host departs the network without sending a State-Change Report message. The router later detects that there is no member host for the corresponding channels when it does not receive a Current-State Report within the timeout of the response for the periodic General Query (and then the group or source timers are expired). However, this situation prolongs leave latency and wastes network resources since the router forwards unneeded traffic for a while.

The second case occurs when a router sends a specific query but does not receive a Current-State Report from a downstream host within an LMQT or LLQT period. It recognizes that no member host exists on the LAN and might prune the routing path. The router reestablishes the routing path when it receives the solicited report message for the channels. However, the downstream hosts may loose the data packets until the routing path is reestablished and the data forwarding is restarted.

If operators do not believe that their link is fairly robust or that they can configure the [Robustness Variable] value appropriately, they may configure the LMQC/LLQC value to 2 (the default value of the [Robustness Variable] value) or bigger value for their routers. In this case, the routers would enable the explicit tracking function but may want to disable the specific query suppression specified in Section 4. Such configurations will not contribute to saving network resources, but reduce the risk of the incorrect membership expectation.

7. All-Zero and Unspecified Source Addresses

The IGMPv3 specification [2] mentions that an IGMPv3 report is usually sent with a valid IP source address, yet it permits a host to use the 0.0.0.0 source address (since the host has not yet acquired an IP address), and routers must accept a report with this source address.

When a router enabling the explicit tracking function receives IGMP report messages with an all-zero source address, it deals with the IGMP report messages correctly as defined in [2] and continuously keeps track of the membership state. However, the router SHOULD NOT maintain the host specifying all-zero source address in its membership state information. The router will maintain its membership state information by checking Current-State reports as ordinary routers do.

On the other hand, the MLDv2 specification [3] mentions that routers silently discard a message that is sent with an invalid link-local address or sent with the unspecified address (::), without taking any action, because of security considerations. According to this specification, whether the explicit tracking function is used or not, a router does not deal with a member hosts sending an MLD report message with the unspecified source address.

8. Compatibility with Older Version Protocols

The explicit tracking function does not work with older versions of IGMP or MLD, IGMPv1 [5], IGMPv2 [6], or MLDv1 [7], because a member host using these protocols enables "membership report suppression" by which the host will cancel sending pending membership reports if a similar report is observed from another member on the network.

To preserve compatibility with older versions of IGMP/MLD, routers supporting IGMPv3/MLDv2 enable the host compatibility mode defined in [2][3]. The host compatibility mode of an interface changes the operational protocol version on the LAN whenever an older version query (than the current compatibility mode) is heard or when certain timer conditions occur. The routers can hence support downstream hosts that are not upgraded to the latest versions and run membership report suppression.

Therefore, if a multicast router supporting IGMPv3/MLDv2 and enabling the explicit tracking function changes its compatibility mode to the older versions, the router SHOULD disable the explicit tracking function while it acts as the older version router.

9. Interoperability

There might be various ways to implement the explicit tracking function. Some existing implementations may not implement the mechanisms such as specific query suppression described in this document. Yet, the explicit tracking function does not change on-wire behavior, and the function or mechanisms described in this document do not break the interoperability between the existing implementations and the implementation based on this specification.

On the other hand, for the future implementation for the explicit tracking function, since this document specifies the minimum but effective sets of the explicit tracking function, it is RECOMMENDED to refer and follow this specification as the standard implementation for that function.

10. IANA Considerations

This document has no actions for IANA.

11. Security Considerations

The explicit tracking function potentially requires a large amount of memory so that routers keep all membership states. It gives some impact in the cases where (1) a router attaches to a link or an IGMP/MLD proxy device [8] that has a large number of member hosts, and a router has insufficient memory resources to maintain a large number of member hosts, or (2) a malicious host sends a large number of invalid IGMP/MLD State-Change Report messages without any intent to join the specified channels.

For the first case, operators may disable the explicit tracking function, despite the benefits mentioned above. For the second case, some serious threats may be induced. For instance;

1. Transmitting a large number of invalid IGMP/MLD report messages consumes network resources.

2. Keeping a large number of invalid membership states on a router consumes the router’s memory resources.

3. Dealing with a large number of invalid membership states on a router consumes the router’s CPU resources.

In order to mitigate such threats, a router enabling the explicit tracking function may limit a total amount of membership information the router can store, or may rate-limit State-Change Report messages per host. When the router enables rate-limiting per host, the router MAY ignore the received State-Change Report messages to minimize the processing overhead or prevent DoS attacks. The rate limit is left to the router’s implementation.

12. Acknowledgements

Luis M. Contreras, Toerless Eckert, Adrian Farrel, Sergio Figueiredo, Bharat Joshi, Nicolai Leymann, Magnus Nystrom, Stig Venaas, and others provided many constructive and insightful comments.

13. References

13.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to indicate requirement levels", RFC 2119, March 1997.

[2] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

[3] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

[4] Liu, H., Cao, W., and H. Asaeda, "Lightweight Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Version 2 (MLDv2) Protocols", RFC 5790, February 2010.

13.2. Informative References

[5] Deering, S., "Host Extensions for IP Multicasting", RFC 1112, August 1989.

[6] Fenner, W., "Internet Group Management Protocol, Version 2", RFC 2236, November 1997.

[7] Deering, S., Fenner, W., and B. Haberman, "Multicast Listener Discovery (MLD) for IPv6", RFC 2710, October 1999.

[8] Fenner, B., He, H., Haberman, B., and H. Sandick, "Internet Group Management Protocol (IGMP) / Multicast Listener Discovery (MLD)-Based Multicast Forwarding ("IGMP/MLD Proxying")", RFC 4605, August 2006.

Author’s Address

Hitoshi Asaeda
National Institute of Information and Communications Technology
4-2-1 Nukui-Kitamachi
Koganei, Tokyo 184-8795
Japan

Email: asaeda@nict.go.jp

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2015

Y. Liu
F. Guo
Huawei

October 22, 2014

Yang Model for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD)
draft-liu-pim-igmp-mld-yang-00

Abstract

This document defines a YANG data model that can be used to configure and manage IGMP and MLD.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 25, 2015.

Copyright Notice

Table of Contents

1. Introduction

YANG[RFC6020] is a data definition language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF[RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces(e.g. REST) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interface, such as CLI and Programmatic APIs.

This document defines a YANG data model that can be used to configure and manage IGMP and MLD. It includes IGMPv1[RFC1112], IGMPv2[RFC2236], IGMPv3[RFC3376] and MLDv1[RFC2710], MLDv2[RFC3810]. In addition, features described in IGMP and MLD standards other than mentioned above RFC are also supported. For convenience, IGMP and MLD are wholly called GMP below.

2. Terminology

- IGMP: Internet Group Management Protocol
- MLD: Multicast Listener Discovery

- o GMP: Group Management Protocol
- o SSM: Source-Specific Multicast

3. Design of Data Model

3.1. Overview

The GMP Yang module has only one main container ::

- o gmpInstances : that contains per-instance writable configuration objects.

The figure below describes the overall structure of the GMP Yang module :

```
module: gmp
+--rw gmp
   +--rw gmp-Instances
      +--rw gmp-Instance * [vrfName]
         |  ...
         +--rw gmp-SSM-Mappings
            ...
         |  ...
         +--rw gmp-Interfaces
            +--rw gmp-Interface * [ifName]
               |  ...
               +--rw gmp-Static-Groups
                  ...
```

Figure 1 The overview of GMP YANG data model

3.2. GMP Per-instance Configuration

GMP per-instance configuration container includes parameters of the public GMP instance or the GMP instance binding a specific VRF. GMP per-instance configuration container is divided into:

- o Per-instance parameters
- o Per-SSM-Mapping configuration of the GMP instance
- o Per-interface configuration of the GMP instance

3.2.1. Per-instance Parameters

The per-instance parameter includes the name of the VRF bound by the GMP instance, and timer parameters such as query interval etc.

```
      +--rw gmp-Instances
         +--rw gmp-Instance* [vrfName]
            +--rw vrfName                    string
            +--rw addrFamily                 enumeration
            +--rw queryInterval?             uint32
            +--rw queryRspInterval?          uint32
            +--rw robustness?                uint32
            +--rw lastMemberQueryInterval?   uint32
            +--rw reqRouterAlert?            boolean
            +--rw sendRouterAlert?           boolean
            +--rw othQuerierPstTime?         uint32
            +--ro gmpEntryLimit?             uint32
            +--rw ipsecName?                 string
            +--rw ipsecType?                 enumeration
```

Figure 2 The YANG data model of GMP instance

3.2.2. Per-SSM-Mapping Configuration of GMP Instance

Per-SSM-Mapping configuration of the GMP instance includes the SSM Mapping rules.IGMPv1/v2 and MLDv1 reports can use these rules to map SG state for PIM SSM[RFC4607].IGMPv3 and MLDv2 can default use PIM SSM , which is described in [RFC4604].

```
          +--rw gmp-SSM-Mappings
           |  +--rw gmp-SSM-Mapping
           |     +--rw IPV4-ssmmapingGrp     inet:ipv4-address
           |     +--rw IPV6-ssmmapingGrp     inet:ipv6-address
           |     +--rw isSSMapMask           boolean
           |     +--rw IPV4-ssmmapingMask?   inet:ipv4-address
           |     +--rw IPV6-ssmmapingMask?   uint32
           |     +--rw isMaskLen             boolean
           |     +--rw maskLen?              uint32
           |     +--rw IPV4-srcAddr          inet:ipv4-address
           |     +--rw IPV6-srcAddr          inet:ipv6-address
```

Figure 3 The YANG data model of GMP SSM-Mapping

3.2.3. Per-interface Configuration of GMP Instance

Per-interface configuration of the GMP instance includes the interface name, timer parameters, policies, static groups etc.GMP per-instance configuration container is divided into two containers:

o Per-interface parameters

o Per-static-group configuration of the GMP interface

3.2.3.1. Per-interface Parameters

The per-interface parameter includes the name of the interface, and the VRF name bound by the interface, and time parameters, policies etc.

```
           +--rw gmp-Interfaces
               +--rw gmp-Interface* [ifName]
                  +--rw vrfName                    string
                  +--rw ifName                     ifName
                  +--rw addrFamily                 enumeration
                  +--rw gmpEnable                  boolean
                  +--rw ipSourcePly?               boolean
                  +--rw ipSrcAclName?              string
                  +--rw ipSrcAclIpv6?              string
                  +--rw queryInterval?             uint32
                  +--rw queryRspInterval?          uint32
                  +--rw robustness?                uint32
                  +--rw version?                   uint32
                  +--rw lastMemberQueryInterval?   uint32
                  +--rw requireRouterAlert?        boolean
                  +--rw sendRouterAlert?           boolean
                  +--rw othQuerierPresentTime?     uint32
                  +--rw immediatelyLeave?          boolean
                  +--rw immLeaveAclName?           string
                  +--rw immLeaveAclIpv6?           string
                  +--rw gmpEntryLimit?             uint32
                  +--rw exceptAclName?             string
                  +--rw exceptAclIpv6?             string
                  +--rw ssmapEnable?               boolean
                  +--rw groupAclName?              string
                  +--rw groupAclIpv6?              string
                  +--rw groupAclGMPVer?            uint32
                  +--rw queryAclName?              string
                  +--rw queryAclIpv6?              string
                  +--rw ipsecName?                 string
                  +--rw ipsecType?                 enumeration
```

Figure 4 The YANG data model of GMP interface

3.2.3.2. Per-static-group Configuration of GMP interface

Per-static-group configuration of the GMP interface includes the static group address, and as a option also includes source address, every static group step, and group numbers on the interface.

```
                +--rw gmp-Static-Groups
                   +--rw gmp-Static-Group
                      +--rw vrfName                string
                      +--rw addrFamily             enumeration
                      +--rw ifName                 ifName
                      +--rw IPV4-staticGrp         inet:ipv4-address
                      +--rw IPV6-staticGrp         inet:ipv6-address
                      +--rw isSourceAddr           boolean
                      +--rw IPV4-sourceAddr?       inet:ipv4-address
                      +--rw IPV6-sourceAddr?       inet:ipv6-address
                      +--rw isStepGrpMask          boolean
                      +--rw IPV4-incStepGrpMask?   inet:ipv4-address
                      +--rw IPV6-incStepGrpMask?   inet:ipv6-address
                      +--rw isMaskLen              boolean
                      +--rw maskLen?               uint32
                      +--rw totalNum?              uint32
```

Figure 5 The YANG data model of GMP static group

4. GMP Yang Module

```
module gmp {
    namespace "urn:huawei:params:xml:ns:yang:gmp";
    // replace with IANA namespace when assigned - urn:ietf:params:xml:ns:yang:1
    prefix "gmp";
  import ietf-inet-types {
    prefix inet;
  }
    organization
      "Huawei Technologies Co., Ltd.";
    contact
      "liuyisong@huawei.com
       guofeng@huawei.com ";
    description
      "This YANG module defines the generic configuration
     data for GMP, i.e. IGMP and MLD, which is common across all of the vendor
     implementations of the protocol. It is intended that the module
     will be extended by vendors to define vendor-specific
     GMP configuration parameters.";
  revision 2014-10-21 {
    description
      "Initial revision.";
```

```
  }

  typedef ifName {
        description "ifName is like ethernet1/1/1/1";
        type string {
            length "1..63";
        }
    }


    container gmp {

        container gmp-Instances {

            list gmp-Instance {

                key "vrfName";
                max-elements "unbounded";
                min-elements "0";
                description "Specifies a list of gmp instances.";

                leaf vrfName {
                    description "Name of an gmp instance.If the name string is e
mpty the instance means a public instance whose name is _public_.";
                    config "true";
                    //default "_public_";
                    type string {
                        length "0..32";
                    }
                }
                leaf addrFamily {
                    description "Specify an address family, which determines whe
ther an address is an IPv4 or IPv6 address.";
                    config "true";
                    mandatory "true";
                    type enumeration {
                        enum ipv4unicast {
                            value "0";
                            description "Specify an address family, which determ
ines whether an address is an IPv4 or IPv6 address.";
                        }
                        enum ipv6unicast {
                            value "1";
                            description "Specify an address family, which determ
ines whether an address is an IPv4 or IPv6 address.";
                        }
                    }
                }
                leaf queryInterval {
                    description "Specify the interval at which the router sends
general query messages. The value is an integer ranging from 1 to 18000, in seco
nds. The default value for IPv4 is 60, and 125 for IPv6.";
                    config "true";
                    default "60";
```

```
                    type uint32 {
                        range "1..18000";
                    }
                }
                leaf queryRspInterval {
                    description "Specify the maximum response time for a query m
essage. The value is an integer ranging from 1 to 25, in seconds. The default va
lue is 10.";
                    config "true";
                    default "10";
                    type uint32 {
                        range "1..25";
                    }
                }
                leaf robustness {
                    description "Specify the number of times for retransmitting
a message to avoid the packet loss. The value is an integer ranging from 2 to 5.
 The default value is 2.";
                    config "true";
                    default "2";
                    type uint32 {
                        range "2..5";
                    }
                }
                leaf lastMemberQueryInterval {
                    description "Specify the interval at which the querier sends
 last-member query messages. The value is an integer ranging from 1 to 5, in sec
onds. The default value is 1. This parameter makes sense only when the current q
uerier runs IGMPv2, IGMPv3 or MLD.";
                    config "true";
                    default "1";
                    type uint32 {
                        range "1..5";
                    }
                }
                leaf reqRouterAlert {
                    description "Configure the router to process only the messag
es whose IP headers contain Router-Alert options globally. By default, the route
r does not check whether the received IGMP messages contain Router-Alert options
.";
                    config "true";
                    type boolean {
                    }
                }
                leaf sendRouterAlert {
                    description "Configure the router to send the messages with
Router-Alert options in the IP headers globally. By default, the IP headers cont
ain Router-Alert options. ";
                    config "true";
                    type boolean {
                    }
                }
                leaf othQuerierPstTime {
                    description "Set the global Keepalive period for other queri
ers. ";
                    config "true";
                    type uint32 {
                        range "60..300";
                    }
                }
                leaf gmpEntryLimit {
```

```
                    description "Set the maximum number of entries that can be c
reated for the current instance. The value is an integer ranging from 1 to 49152
. The default value is 49152.";
                    config "false";
                    type uint32 {
                        range "1..49152";
                    }
                }
                leaf ipsecName {
                    description "SA name. The value is a string of 1 to 15 chara
cters.";
                    config "true";
                    type string {
                        length "1..15";
                        pattern "^[^ ]+$";
                    }
                }
                leaf ipsecType {
                    description "IPsec type used as a query option.";
                    config "true";
                    type enumeration {
                        enum forAll {
                            value "0";
                            description "IPsec type used as a query option.";
                        }
                        enum forQuery {
                            value "1";
                            description "IPsec type used as a query option.";
                        }
                    }
                }

                container gmp-SSM-Mappings {

                    container gmp-SSM-Mapping {

                        leaf IPV4-ssmmapingGrp {
                            description "Specify the address of a multicast grou
p.";
                            config "true";
                            mandatory "true";
                            type inet:ipv4-address;
                        }
                        leaf IPV6-ssmmapingGrp {
                            description "Specify the address of a multicast grou
p.";
                            config "true";
                            mandatory "true";
                            type inet:ipv6-address;
                        }
                        leaf isSSMapMask {
                            description "Whether a mask is configured for a mult
icast group address.";
                            config "true";
```

```
                            mandatory "true";
                            type boolean {
                            }
                        }
                        leaf IPV4-ssmmapingMask {
                            description "Specify the mask of a multicast group a
ddress.";
                            config "true";
                            type inet:ipv4-address;
                        }
                        leaf IPV6-ssmmapingMask {
                            description "Specify the mask of a multicast group a
ddress.";
                            config "true";
                            type uint32 {
                                range "0..128";
                            }
                        }

                        leaf isMaskLen {
                            description "Whether the length is set for the mask
of a multicast group address.";
                            config "true";
                            mandatory "true";
                            type boolean {
                            }
                        }
                        leaf maskLen {
                            description "Specify the mask length of a multicast
group address. In the case of an IPv4 address, the mask length ranges from 4 to
32. In the case of an IPv6 address, the mask length is 16/32/64/128.";
                            config "true";
                            type uint32 {
                                range "0..128";
                            }
                        }
                        leaf IPV4-srcAddr {
                            description "Specify the address of a multicast sour
ce.";
                            config "true";
                            mandatory "true";
                            type inet:ipv4-address;
                        }
                        leaf IPV6-srcAddr {
                            description "Specify the address of a multicast sour
ce.";
                            config "true";
                            mandatory "true";
                            type inet:ipv6-address;
                        }
                    }
                }

                container gmp-Interfaces {
```

```
                    list gmp-Interface {

                        key "ifName";
                        max-elements "unbounded";
                        min-elements "0";
                        description "Specifies an gmp interface.";

                        leaf vrfName {
                            description "Name of an gmp instance. If the name st
ring is empty the instance means a public instance whose name is _public_.";
                            config "true";
                            mandatory "true";
                            //default "_public_";
                            type string {
                                length "0..32";
                            }
                        }
                        leaf ifName {
                            description "Interface name.";
                            config "true";
                            type ifName;
                        }
                        leaf addrFamily {
                            description "Specify an address family, which determ
ines whether an address is an IPv4 or IPv6 address.";
                            config "true";
                            mandatory "true";
                            type enumeration {
                                enum ipv4unicast {
                                    value "0";
                                    description "Specify an address family, whic
h determines whether an address is an IPv4 or IPv6 address.";
                                }
                                enum ipv6unicast {
                                    value "1";
                                    description "Specify an address family, whic
h determines whether an address is an IPv4 or IPv6 address.";
                                }
                            }
                        }
                        leaf gmpEnable {
                            description "Enable protocols on an interface.";
                            config "true";
                            mandatory "true";
                            type boolean {
                            }
                        }
                        leaf ipSourcePly {
                            description "Configure a policy for filtering IGMP R
eport messages based on host addresses. By default, no policy is configured for
filtering IGMP Report messages based on host addresses.";
                            config "true";
                            type boolean {
                            }
```

```
                        }
                        leaf ipSrcAclName {
                            description "Configure an ACL that defines a host ad
dresses range. The value is an integer ranging from 2000 to 2999, or a case-sens
itive string with a maximum of 32 characters. By default, the ACL is not configu
red.";
                            config "true";
                            type string {
                                length "1..32";
                                pattern "^[^ ]+$";
                            }
                        }
                        leaf ipSrcAclIpv6 {
                            description "Configure an ACL that defines a host ad
dresses range. The value is an integer ranging from 2000 to 2999, or a case-sens
itive string with a maximum of 32 characters. By default, the ACL is not configu
red.";
                            config "true";
                            type string {
                                length "1..32";
                                pattern "^[^ ]+$";
                            }
                        }
                        leaf queryInterval {
                            description "Specify the interval at which the route
r sends general query messages. The value is an integer ranging from 1 to 18000,
 in seconds. The default value for IPv4 is 60, and 125 for IPv6.";
                            config "true";
                            default "60";
                            type uint32 {
                                range "1..18000";
                            }
                        }
                        leaf queryRspInterval {
                            description "Specify the maximum response time for a
 query message. The value is an integer ranging from 1 to 25, in seconds. The de
fault value is 10. This time is used to control the deadline of mainframe feed b
ack the relation-ship of group members.";
                            config "true";
                            default "10";
                            type uint32 {
                                range "1..25";
                            }
                        }
                        leaf robustness {
                            description "Specify the number of times for retrans
mitting messages to avoid packet loss. The value is an integer ranging from 2 to
 5. The default value is 2.";
                            config "true";
                            default "2";
                            type uint32 {
                                range "2..5";
                            }
                        }
                        leaf version {
                            description "Specify the version of IGMP or MLD runn
ing on an interface. By default, IGMPv2 or MLDv2 is used.";
                            config "true";
                            default "2";
                            type uint32 {
                                range "1..3";
                            }
```

```
                        }
                        leaf lastMemberQueryInterval {
                            description "Specify the interval at which the queri
er sends last-member query messages. The value is an integer ranging from 1 to 5
, in seconds. The default value is 1. This parameter makes sense only when the c
urrent querier runs IGMPv2, IGMPv3 or MLD.";
                            config "true";
                            default "1";
                            type uint32 {
                                range "1..5";
                            }
                        }
                        leaf requireRouterAlert {
                            description "Configure an interface to process only
messages whose IP headers contain Router-Alert options. By default, the interfac
e does not check whether the received messages contain Router-Alert options.";
                            config "true";
                            type boolean {
                            }
                        }
                        leaf sendRouterAlert {
                            description "Configure an interface to send the mess
ages with Router-Alert options in the IP headers. By default, the IP header cont
ain Router-Alert options.";
                            config "true";
                            default "true";
                            type boolean {
                            }
                        }
                        leaf othQuerierPresentTime {
                            description "Set the Keepalive period for other quer
iers on an interface. The value ranges from 60 to 300, in second. By default, no
 Keepalive period is set for other queriers.";
                            config "true";
                            type uint32 {
                                range "60..300";
                            }
                        }
                        leaf immediatelyLeave {
                            description "Configure an interface that receives a
Leave message of a certain group to immediately delete the corresponding group r
ecords, without sending a last-member query message.";
                            config "true";
                            type boolean {
                            }
                        }
                        leaf immLeaveAclName {
                            description "Configure an ACL that defines a multica
st group range. The basic ACL number ranges from 2000 to 2999, and the advanced
ACL number ranges from 3000 to 3999. The name is a string with a maximum of 32 c
ase-sensitive characters. By default, the ACL is not con";
                            config "true";
                            type string {
                                length "1..32";
                                pattern "^[^ ]+$";
                            }
                        }
                        leaf immLeaveAclIpv6 {
                            description "Configure an ACL that defines a multica
st group range. The basic ACL number ranges from 2000 to 2999, and the advanced
ACL number ranges from 3000 to 3999. The name is a string with a maximum of 32 c
ase-sensitive characters. By default, the ACL is not con";
                            config "true";
```

```
                type string {
                    length "1..32";
```

```
                            pattern "^[^ ]+$";
                        }
                    }
                    leaf gmpEntryLimit {
                        description "Specify the maximum number of entries t
hat the current interface can create. It is an integer ranging from 1 to 16384.
The default value is 16384.";
                        config "true";
                        type uint32 {
                            range "1..16384";
                        }
                    }
                    leaf exceptAclName {
                        description "Specify the range of multicast groups,
the number of IGMP entries corresponding to which needs not be limited. The basi
c ACL number ranges from 2000 to 2999. The basic ACL filters group addresses onl
y, without distinguishing (*, G) entries and (S, G) entr";
                        config "true";
                        type string {
                            length "1..32";
                            pattern "^[^ ]+$";
                        }
                    }
                    leaf exceptAclIpv6 {
                        description "Specify the range of multicast groups,
the number of IGMP entries corresponding to which needs not be limited. The basi
c ACL number ranges from 2000 to 2999. The basic ACL filters group addresses onl
y, without distinguishing (*, G) entries and (S, G) entr";
                        config "true";
                        type string {
                            length "1..32";
                            pattern "^[^ ]+$";
                        }
                    }
                    leaf ssmapEnable {
                        description "Enable SSM mapping on an interface.";
                        config "true";
                        type boolean {
                        }
                    }
                    leaf groupAclName {
                        description "Set a filter for multicast groups on an
 interface to control the range of multicast groups that hosts can join. The bas
ic ACL number ranges from 2000 to 2999, and the advanced ACL number ranges from
3000 to 3999. The name is a string with a maximum of 32 ";
                        config "true";
                        type string {
                            length "1..32";
                            pattern "^[^ ]+$";
                        }
                    }
                    leaf groupAclIpv6 {
                        description "Set a filter for multicast groups on an
 interface to control the range of multicast groups that hosts can join. The bas
ic ACL number ranges from 2000 to 2999, and the advanced ACL number ranges from
3000 to 3999. The name is a string with a maximum of 32 ";
                        config "true";
                        type string {
                            length "1..32";
                            pattern "^[^ ]+$";
                        }
                    }
```

```
                        leaf groupAclGMPVer {
                            description "Forbids hosts that use a specified vers
ion to join the multicast group. The version can be specified in this command on
ly after a multicast filtering policy is configured.";
                            config "true";
                            type uint32 {
                                range "1..3";
                            }
                        }
                        leaf queryAclName {
                            description "Configure an ACL that defines a host ad
dresses range. The value is an integer ranging from 2000 to 2999, or a case-sens
itive string with a maximum of 32 characters. By default, the ACL is not configu
red.";
                            config "true";
                            type string {
                                length "1..32";
                                pattern "^[^ ]+$";
                            }
                        }
                        leaf queryAclIpv6 {
                            description "Configure an ACL that defines a host ad
dresses range. The value is an integer ranging from 2000 to 2999, or a case-sens
itive string with a maximum of 32 characters. By default, the ACL is not configu
red.";
                            config "true";
                            type string {
                                length "1..32";
                                pattern "^[^ ]+$";
                            }
                        }
                        leaf ipsecName {
                            description "SA name. The value is a string of 1 to
15 characters.";
                            config "true";
                            type string {
                                length "1..15";
                                pattern "^[^ ]+$";
                            }
                        }
                        leaf ipsecType {
                            description "IPsec type used as a query option.";
                            config "true";
                            type enumeration {
                                enum forAll {
                                    value "0";
                                    description "IPsec type used as a query opti
on.";
                                }
                                enum forQuery {
                                    value "1";
                                    description "IPsec type used as a query opti
on.";
                                }
                            }
                        }

                        container gmp-Static-Groups {
```

```
                            container gmp-Static-Group {

                                leaf vrfName {
                                    description "Name of an GMP instance. If the
 name string is empty the instance means a public instance whose name is _public
_.";
                                    config "true";
                                    mandatory "true";
                                    //default "_public_";
                                    type string {
                                        length "0..32";
                                    }
                                }
                                leaf addrFamily {
                                    description "Specify an address family, whic
h determines whether an IPv4 or IPv6 address is used.";
                                    config "true";
                                    mandatory "true";
                                    type enumeration {
                                        enum ipv4unicast {
                                            value "0";
                                            description "Specify an address fami
ly, which determines whether an IPv4 or IPv6 address is used.";
                                        }
                                        enum ipv6unicast {
                                            value "1";
                                            description "Specify an address fami
ly, which determines whether an IPv4 or IPv6 address is used.";
                                        }
                                    }
                                }
                                leaf ifName {
                                    description "Interface name.";
                                    config "true";
                                    mandatory "true";
                                    type ifName;
                                }
                                leaf IPV4-staticGrp {
                                    description "Specifies the address of a stat
ic group.";
                                    config "true";
                                    mandatory "true";
                                    type inet:ipv4-address;
                                }
                                leaf IPV6-staticGrp {
                                    description "Specifies the address of a stat
ic group.";
                                    config "true";
                                    mandatory "true";
                                    type inet:ipv6-address;
                                }
                                leaf isSourceAddr {
                                    description "Whether an address is configure
d for the multicast source.";
                                    config "true";
                                    mandatory "true";
```

```
                                        type boolean {
                                        }
                                }
                                leaf IPV4-sourceAddr {
                                    description "Specifies the IPv4 address of a
 remote neighbor.";
                                    config "true";
                                    type inet:ipv4-address;
                                }
                                leaf IPV6-sourceAddr {
                                    description "Specifies the IPv6 address of a
 remote neighbor.";
                                    config "true";
                                    type inet:ipv6-address;
                                }
                                leaf isStepGrpMask {
                                    description "Determine whether to specify th
e step mask in batch configuration mode.";
                                    config "true";
                                    mandatory "true";
                                    type boolean {
                                    }
                                }
                                leaf IPV4-incStepGrpMask {
                                    description "Specify the step mask of a grou
p address in batch configuration mode.";
                                    config "true";
                                    type inet:ipv4-address;
                                }
                                leaf IPV6-incStepGrpMask {
                                    description "Specify the step mask of a grou
p address in batch configuration mode.";
                                    config "true";
                                    type inet:ipv6-address;
                                }
                                leaf isMaskLen {
                                    description "Determine whether to set the le
ngth for the step mask of a multicast group address in batch configuration mode.
 ";
                                    config "true";
                                    mandatory "true";
                                    type boolean {
                                    }
                                }
                                leaf maskLen {
                                    description "Specify the mask length of a mu
lticast group address. In the case of an IPv4 address, the mask length ranges fr
om 4 to 32. In the case of an IPv6 address, the mask length is 16/32/64/128.";
                                    config "true";
                                    type uint32 {
                                        range "0..128";
                                    }
                                }
                                leaf totalNum {
                                    description "Specify the number of multicast
 group addresses in batch configuration mode. It is an integer ranging from 2 to
 512.";
                                    config "true";
                                    default "2";
```

```
                                    type uint32 {
                                        range "2..512";
                                    }
                                }
                            }
                        }
                    }

                }

            }

        }
    }
}
```

5. IANA Considerations

This draft includes no request to IANA.

6. Security Considerations

The data model defined does not create any security implications. This draft does not change any underlying security issues inherent in [I-D.ietf-netmod-routing-cfg].

7. Acknowledgements

TBD

8. References

8.1. Normative References

[I-D.ietf-netmod-routing-cfg]
Lhotka, L., "A YANG Data Model for Routing Management", draft-ietf-netmod-routing-cfg-15 (work in progress), May 2014.

[RFC1112] Deering, S., "Host extensions for IP multicasting", STD 5, RFC 1112, August 1989.

[RFC2236] Fenner, W., "Internet Group Management Protocol, Version 2", RFC 2236, November 1997.

[RFC2710] Deering, S., Fenner, W., and B. Haberman, "Multicast Listener Discovery (MLD) for IPv6", RFC 2710, October 1999.

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

[RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

[RFC4604] Holbrook, H., Cain, B., and B. Haberman, "Using Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Protocol Version 2 (MLDv2) for Source-Specific Multicast", RFC 4604, August 2006.

[RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, August 2006.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

8.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

Authors' Addresses

Yisong Liu
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: liuyisong@huawei.com

Feng Guo
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: guofeng@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2015

Y. Liu
F. Guo
Huawei

October 22, 2014

YANG Data Model for PIM
draft-liu-pim-yang-00

Abstract

This document defines a YANG data model that can be used to configure and manage PIM.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 25, 2015.

Copyright Notice

Table of Contents

1. Introduction

YANG[RFC6020] is a data definition language that was introduced to define the contents of a conceptual data store that allows networked devices to be managed using NETCONF[RFC6241]. YANG is proving relevant beyond its initial confines, as bindings to other interfaces(e.g. REST) and encoding other than XML (e.g. JSON) are being defined. Furthermore, YANG data models can be used as the basis of implementation for other interface, such as CLI and Programmatic APIs.

This document defines a YANG data model that can be used to configure and manage PIM. It includes PIM SM[RFC4601], PIM SSM[RFC4607][RFC4608],PIM BSR[RFC5059]. In addition, It can extend PIM DM[RFC3973], BIDIR PIM[RFC5015] etc. features described in PIM standards other than mentioned above RFC in future version.

2. Terminology

- PIM: Protocol Independent Multicast
- SM: Sparse Mode
- SSM: Source-specific Multicast
- DM: Dense Mode
- BSR: Bootstrap Router
- RP: Rendezvous Point
- SPT: shortest-path tree
- RPT: Rendezvous Point Tree

3. Design of Data Model

3.1. Overview

Because PIM SSM is a subset of PIM SM, it is not necessary to assign a separate container for PIM SSM. The PIM Yang module is divided into two main containers :

- PIM_BSR : that contains all pim bsr&rp writable configuration objects.
- PIM_SM : that contains all pim sm writable configuration objects.

The figure below describes the overall structure of the PIM Yang module :

```
module: pim
   +--rw PIM
      +--rw PIM_BSR
      |  +--rw PIM_BSR-Instances
      |     +--rw C-BSR-Instance* [vrfName]
      |     |  ...
      |     +--rw C-RP-Instance* [vrfName]
      |     |  ...
      +--rw PIM_SM
         +--rw PIM_SM-Instances
            +--rw PIM_SM-Instance* [vrfName]
               |  ...
               +--rw PIM-Static-RPs
               |  +--rw PIM-Static-RP
               |     |  ...
               +--rw PIM-SPT-Switchs
               |  +--rw PIM-SPT-Switch* [sptGrpPlyName]
               |     |  ...
               +--rw PIM_SM-Anycast-RPs
               |  +--rw PIM_SM-Anycast-RP
               |     |  ...
               |     +--rw PIM_SM-RP-Peers
               |        +--rw PIM_SM-RP-Peer
               |           |  ...
               +--rw PIM_SM-Interfaces
                  +--rw PIM_SM-Interface* [ifName]
                  |  ...
```

Figure 1 The overview of PIM YANG data model

3.2. PIM BSR Configuration

PIM BSR configuration container has only one main container:

- PIM_BSR-instances : that contains all pim c-bsr and c-rp writable configuration objects.

PIM BSR per-instance configuration container includes c-bsr and c-rp parameters of the public PIM instance or the PIM instance binding a specific VRF. PIM BSR per-instance configuration container is divided into:

- Per-instance C-BSR configuration of the PIM BSR instance.

- Per-instance C-RP configuration of the PIM BSR instance.

3.2.1. Per-instance C-BSR Configuration of PIM BSR Instance

The per-instance C-BSR configuration includes the name of the VRF bound by the pim bsr instance, and timer parameters , policies etc. it also includes administratively scoped BSR configuration.

```
+--rw C-BSR-Instance* [vrfName]
      |     |  +--rw vrfName                string
      |     |  +--rw addressFamily          enumeration
      |     |  +--rw cBsrIfName             ifName
      |     |  +--rw IPV4-cBsrIfAddr?       inet:ipv4-address
      |     |  +--rw IPV6-cBsrIfAddr        inet:ipv6-address
      |     |  +--rw cBsrHoldTime?          uint32
      |     |  +--rw cBsrInterval?          uint32
      |     |  +--rw cBsrHashLen?           uint32
      |     |  +--rw cBsrPriority?          uint32
      |     |  +--rw cBsrPlyName?           string
      |     |  +--rw cBsrPlyIpv6?           string
      |     |  +--rw cBsrAdminScope?        boolean
      |     |  +--rw cBsrGlobalEnable?      boolean
      |     |  +--rw cBsrGlobalHashLength?  uint32
      |     |  +--rw cBsrGlobalPriority?    uint32
      |     |  +--rw isFragable             boolean
```

Figure 2 The YANG data model of PIM C-BSR

3.2.2. Per-instance C-RP Configuration of PIM BSR Instance

The per-instance C-RP configuration includes the name of the VRF bound by the pim bsr instance, specific interface or address for c-rp and timer parameters , policies etc.

```
      +--rw C-RP-Instance* [vrfName]
      |        +--rw vrfName          string
      |        +--rw cRpIfName        ifName
      |        +--rw cRpGrpPlyName?   string
      |        +--rw cRpPriority?     uint32
      |        +--rw cRpHoldTime?     uint32
      |        +--rw cRpAdvInterval?  uint32
      |        +--rw IPV4-cRpAddr?    inet:ipv4-address
      |        +--rw IPV6-cRpAddr     inet:ipv6-address
```

Figure 3 The YANG data model of PIM C-RP

3.3. PIM SM Configuration

PIM SM configuration container has only one main container:

- PIM_SM-instances : that contains all pim sm writable configuration objects.

PIM SM per-instance configuration container includes pim sm protocol parameters of the public PIM instance or the PIM instance binding a specific VRF. PIM SM per-instance configuration container is divided into:

- Per-instance parameters
- Per-static-RP configuration of the PIM SM instance
- Per-SPT-switch configuration of the PIM SM instance
- Per-Anycast-RP configuration of the PIM SM instance
- Per-interface configuration of the PIM SM instance

3.3.1. Per-instance Parameters

The per-instance parameter includes the name of the VRF bound by the PIM SM instance, and timer parameters such as hello interval etc., and varied policies etc

```
          +--rw PIM_SM-Instances
              +--rw PIM_SM-Instance* [vrfName]
                 +--rw vrfName                string
                 +--rw addressFamily          enumeration
                 +--rw assertHoldTime?        uint32
                 +--rw jpHoldTime?            uint32
                 +--rw probeInterval?         uint32
                 +--rw jpTimerInterval?       uint32
                 +--rw drPriority?            uint32
                 +--rw helloHoldtime?         uint16
                 +--rw helloLandelay?         uint16
                 +--rw helloInterval?         uint32
                 +--rw helloOverride?         uint16
                 +--rw regCheckSum?           uint16
                 +--rw regSuppInterval?       uint32
                 +--rw embeddedRp?            boolean
                 +--rw nbrCheckRecv?          boolean
                 +--rw nbrCheckSend?          boolean
                 +--rw regPlyName?            string
                 +--rw regPlyIpv6?            string
                 +--rw ssmPlyName?            string
                 +--rw ssmPlyIpv6?            string
                 +--rw srcPlyName?            string
                 +--rw srcPlyIpv6?            string
                 +--rw bsrPlyName?            string
                 +--rw bsrPlyIpv6?            string
                 +--rw embRpPlyName?          string
                 +--rw sourceLifeTime?        uint32
                 +--rw sptDetInterval?        uint32
                 +--rw ipsecName?             string
                 +--rw ipsecType?             enumeration
                 +--rw uniIpsecName?          string
```

Figure 4 The YANG data model of PIM SM instance

3.3.2. Per-static-RP configuration of the PIM SM instance

Per-static-RP configuration of the PIM SM instance includes static RP address, preference and policy for group range. For simply deployment, sometimes it is not necessary to deploy dynamic BSR/RP mechanism, and static RP mechanism is also satisfied.

```
                +--rw PIM-Static-RPs
                 |  +--rw PIM-Static-RP
                 |     +--rw IPV4-staticRpAddr      inet:ipv4-address
                 |     +--rw IPV6-staticRpAddr      inet:ipv6-address
                 |     +--rw staticRpPlyName?       string
                 |     +--rw staticRpPlyNameIpv6?   string
                 |     +--rw preference?            enumeration
```

Figure 5 The YANG data mode of PIM static-RP

3.3.3. Per-SPT-switch configuration of the PIM SM instance

Per-SPT-switch configuration of the PIM SM instance includes spt switch threshold, group range by policy.PIM SM SPT switch process as described in [RFC4601] makes multicast flows from RPT to SPT in order to optimizing the forwarding paths.

```
                 +--rw PIM-SPT-Switchs
                 |  +--rw PIM-SPT-Switch* [sptGrpPlyName]
                 |     +--rw infinity?          boolean
                 |     +--rw sptThreshHold?     uint32
                 |     +--rw isACLEnable        boolean
                 |     +--rw sptGrpPlyName      string
                 |     +--rw sptGrpPlcyOrder?   uint32
```

Figure 6 The YANG data model of PIM SPT Switch

3.3.4. Per-Anycast-RP configuration of the PIM SM instance

Anycast-RP as described in [RFC4610] is a mechanism that pim sm register packets have been used to exchange multicast source information and get fast convergence when a PIM Rendezvous Point (RP) router fails. Per-Anycast-RP configuration container is divided into :

- Per-Anycast-RP parameters
- Per-RP-peer configuration of PIM SM Anycast RP

3.3.4.1. Per-Anycast-RP Parameters

Per-Anycast-RP parameter includes the RP for anycast address, and the local address to establish RP peer link.

```
                +--rw PIM_SM-Anycast-RPs
                |  +--rw PIM_SM-Anycast-RP
                |     +--rw IPV4-rpAddress        inet:ipv4-address
                |     +--rw IPV6-rpAddress        inet:ipv6-address
                |     +--rw local-IPV4-Address    inet:ipv4-address
                |     +--rw local-IPV6-Address    inet:ipv6-address
```

Figure 7 The YANG data model of PIM SM Anycast-RP

3.3.4.2. Per-RP-peer Configuration of PIM SM Anycast RP

Per-RP-peer configuration of PIM SM anycast RP includes the peer address, source information forwarding policy etc.

```
             +--rw PIM_SM-RP-Peers
             |        +--rw PIM_SM-RP-Peer
             |           +--rw IPV4-rpPeer-Address    inet:ipv4-address
             |           +--rw IPV6-rpPeer-Address    inet:ipv6-address
             |           +--rw fwdSaSwt?              boolean
             |           +--rw fwdPolicy?             string
             |           +--rw fwdPolicyIpv6?         string
```

Figure 8 The YANG data model of PIM SM Anycast-RP peer

3.3.5. Per-interface configuration of the PIM SM instance

Per-interface configuration of the PIM SM instance includes the interface name, and the VRF name bound by the interface, and time parameters, policies etc.

```
                +--rw PIM_SM-Interfaces
                    +--rw PIM_SM-Interface* [ifName]
                       +--rw vrfName                string
                       +--rw addressFamily          enumeration
                       +--rw ifName                 ifName
                       +--rw pimsmEnable            boolean
                       +--rw drPriority?            uint32
                       +--rw helloInterval?         uint32
                       +--rw helloHoldtime?         uint16
                       +--rw helloOverride?         uint16
                       +--rw helloLanDelay?         uint16
                       +--rw jpTimerInterval?       uint32
                       +--rw jpHoldtime?            uint32
                       +--rw jpPlyName?             string
                       +--rw jpPlyIpv6?             string
                       +--rw jpAsmPlyName?          string
                       +--rw jpAsmPlyIpv6?          string
                       +--rw jpSsmPlyName?          string
                       +--rw jpSsmPlyIpv6?          string
                       +--rw nbrPlyName?            string
                       +--rw nbrPlyIpv6?            string
                       +--rw assertHoldtime?        uint32
                       +--rw requireGenId?          boolean
                       +--rw pimBsrBoundary?        enumeration
                       +--rw bfdEnable?             boolean
                       +--rw bfdMinTx?              uint32
                       +--rw bfdMinRx?              uint32
                       +--rw bfdMultiplier?         uint16
                       +--rw isSilent?              boolean
                       +--rw isDrSwtDelay?          boolean
                       +--rw drSwtDelayInterval?    uint32
                       +--rw ipsecName?             string
                       +--rw ipsecType?             enumeration
```

Figure 9 The YANG data model of PIM SM interface

4. PIM Yang Module

```
module pim {
    namespace "urn:huawei:params:xml:ns:yang:pim";
    // replace with IANA namespace when assigned - urn:ietf:params:xml:ns:yang:1
    prefix "pim";
    import ietf-inet-types {
        prefix inet;
    }
    organization
      "Huawei Technologies Co., Ltd.";
    contact
```

```
     "liuyisong@huawei.com
      guofeng@huawei.com";
    description
      "This YANG module defines the generic configuration
     data for PIM, which is common across all of the vendor
     implementations of the protocol. It is intended that the module
     will be extended by vendors to define vendor-specific
     PIM configuration parameters.";
      revision 2014-10-21 {
        description
          "Initial revision.";
      }

    typedef ifName {
            description "ifName is like ethernet1/1/1/1";
            type string {
                length "1..63";
            }
    }

    container PIM {
        container PIM_BSR {

            container PIM_BSR-Instances {

                list C-BSR-Instance {

                    key "vrfName";
                    max-elements "unbounded";
                    min-elements "0";
                    description "Specifies an PIM BSR instance.";

                    leaf vrfName {
                        description "Name of a VPN instance.";
                        config "true";
                        //default "_public_";
                        type string {
                            length "0..32";
                        }
                    }
                    leaf addressFamily {
                        config "true";
                        mandatory "true";
                        type enumeration {
                            enum ipv4unicast {
                                value "0";
                                description "Address family, which determines wh
ether an address belongs to IPv4 or IPv6.";
                            }
```

```
                            enum ipv6unicast {
                                value "1";
                                description "Address family, which determines wh
ether an address belongs to IPv4 or IPv6.";
                            }
                        }
                    }
                    leaf cBsrIfName {
                        description "Interface name.";
                        config "true";
                        mandatory "true";
                        type ifName;
                    }

                    leaf IPV4-cBsrIfAddr {
                        description "Global IPv4 unicast address of the C-BSR.";
                        config "true";
                        type inet:ipv4-address;
                    }
                    leaf IPV6-cBsrIfAddr {
                        description "Global IPv6 unicast address of the C-BSR.";
                        config "true";
                        mandatory "true";
                        type inet:ipv6-address;
                    }
                    leaf cBsrHoldTime {
                        description "Timeout period (called holdtime) during whi
ch C-BSRs wait to receive bootstrap messages from the BSR. The value is an integ
er ranging from 1 to 214748364, in seconds. The default value is 130. To prevent
 frequent BSR elections, set the same holdtime for all C-BSRs in the same domain
. If an interval (called BS_interval) at which bootstrap messages are sent has b
een set for C-BSRs, ensure that the specified holdtime is larger than the BS_int
erval.";
                        config "true";
                        default "130";
                        type uint32 {
                            range "1..214748364";
                        }
                    }
                    leaf cBsrInterval {
                        description "Interval (called BS_interval) at which a BS
R continuously sends bootstrap messages. The value is an integer ranging from 1
to 107374177, in seconds. The default value is 60. To prevent frequent BSR elect
ions, set the same BS_interval for all C-BSRs in the same domain. If a timeout p
eriod (called holdtime) during which C-BSRs wait to receive bootstrap messages f
rom the BSR has been set for C-BSRs, ensure that the specified BS_interval is sm
aller than the holdtime.";
                        config "true";
                        default "60";
                        type uint32 {
                            range "1..107374177";
                        }
                    }
                    leaf cBsrHashLen {
                        description "Global hash mask length for a C-BSR. The va
lue is an integer ranging from 0 to 32. The default value is 30. In IPv4, the va
lue is an integer ranging from 0 to 32, and the default value is 30. In IPv6, th
e value is an integer ranging from 0 to 128, and the default value is 126.";
                        config "true";
                        default "126";
                        type uint32 {
                            range "0..128";
                        }
```

```
                    }
```

```
                    leaf cBsrPriority {
                        description "Globally specify a priority for all C-BSRs
on the router. The greater the value, the higher the priority. The value is an i
nteger ranging from 0 to 255. The default value is 0. ";
                        config "true";
                        default "0";
                        type uint32 {
                            range "0..255";
                        }
                    }
                    leaf cBsrPlyName {
                        description "Policy for limiting the range of valid BSR
addresses so that a router discards the messages received from the BSRs not in t
he specified address range. The value is an integer ranging from 2000 to 2999, o
r a string of 1 to 32 case-sensitive characters. By default, the range of valid
BSR addresses is not limited.";
                        config "true";
                        type string {
                            length "1..255";
                            pattern "^[^ ]+$";
                            }
                    }
                    leaf cBsrPlyIpv6 {
                        description "Policy for limiting the range of valid BSR
addresses so that a router discards the messages received from the BSRs not in t
he specified address range. The value is an integer ranging from 2000 to 2999, o
r a string of 1 to 32 case-sensitive characters. By default, the range of valid
BSR addresses is not limited.";
                        config "true";
                        type string {
                            length "1..255";
                            pattern "^[^ ]+$";
                        }
                    }
                    leaf cBsrAdminScope {
                        description "One PIM-SM domain is divided into multiple
BSR administrative domains to implement RP-Set advertisement. By default, there
is only one BSR in the entire PIM-SM domain.";
                        config "true";
                        type boolean
                        {
                        }
                    }
                    leaf cBsrGlobalEnable {
                        description "The router is a C-BSR in the global domain.
 By default, no C-BSR is configured in the global domain.";
                        config "true";
                        type boolean
                        {
                        }
                    }
                    leaf cBsrGlobalHashLength {
                        description "Hash mask length for the C-BSR in the globa
l domain. The value is an integer ranging from 0 to 32. The default value is 30.
";
                        config "true";
                        default "30";
                        type uint32 {
                            range "0..32";
                        }
                    }
                    leaf cBsrGlobalPriority {
                        description "Priority for the C-BSR in the global domain
```

```
. The greater the value, the higher the priority. The value is an integer rangin
g from 0 to 255. The default value is 0.";
```

```
                        config "true";
                        default "0";
                        type uint32 {
                            range "0..255";
                        }
                    }
                    leaf isFragable {
                        description "Enable the C-BSR message fragmentation func
tion. By default, this function is disabled.";
                        config "true";
                        mandatory "true";
                        type boolean
                        {

                        }
                    }
                }

                list C-RP-Instance {

                    key "vrfName";
                    max-elements "unbounded";
                    min-elements "0";
                    description "Specifies a C-RP instance.";

                    leaf vrfName {
                        description "Name of a VPN name.";
                        config "true";
                        //default "_public_";
                        type string {
                            length "0..32";
                        }
                    }
                    leaf cRpIfName {
                        description "Interface name.";
                        config "true";
                        mandatory "true";
                        type ifName;
                    }
                    leaf cRpGrpPlyName {
                        description "Policy for limiting the range of valid grou
p addresses. With this policy, a router discards messages received from the addr
esses not in the specified range. The value is an integer ranging from 2000 to 2
999, or a string of 1 to 32 case-sensitive characters.";
                        config "true";
                        type string {
                            length "1..255";
                            pattern "^[^ ]+$";
                        }
                    }
                    leaf cRpPriority {
                        description "Priority of a C-RP. The greater the value,
the lower the priority. The value is an integer ranging from 0 to 255. The defau
lt value is 0.";
```

```
                        config "true";
                        default "0";
                        type uint32 {
                            range "0..255";
                        }
                    }
                    leaf cRpHoldTime {
                        description "Timeout period during which a BSR waits to
receive Advertisement messages from a C-RP. The value is an integer ranging from
 1 to 65535, in seconds. The default value is 150.";
                        config "true";
                        default "150";
                        type uint32 {
                            range "1..65535";
                        }
                    }
                    leaf cRpAdvInterval {
                        description "Interval at which a C-RP sends Advertisemen
t messages. The value is an integer ranging from 1 to 65535, in seconds. The def
ault value is 60.";
                        config "true";
                        default "60";
                        type uint32 {
                            range "1..65535";
                        }
                    }
                    leaf IPV4-cRpAddr {
                        description "Specifies a C-RP address.";
                        config "true";
                        type inet:ipv4-address;
                    }
                    leaf IPV6-cRpAddr {
                        description "Specifies a C-RP address.";
                        config "true";
                        mandatory "true";
                        type inet:ipv6-address;
                    }
                }
            }

        }

        container PIM_SM {
            container PIM_SM-Instances {
                list PIM_SM-Instance {

                    key "vrfName";
                    max-elements "unbounded";
                    min-elements "0";
                    description "Specifies an PIM-SM instance.";

                    leaf vrfName {
```

```
                        description "Name of a VPN instance.";
                        config "true";
                        //default "_public_";
                        type string {
                            length "0..32";
                        }
                    }

                    leaf addressFamily {
                        config "true";
                        mandatory "true";
                        type enumeration {
                            enum ipv4unicast {
                                value "0";
                                description "Address family, which determines wh
ether an address belongs to IPv4 or IPv6.";
                            }
                            enum ipv6unicast {
                                value "1";
                                description "Address family, which determines wh
ether an address belongs to IPv4 or IPv6.";
                            }
                        }
                    }
                    leaf assertHoldTime {
                        description "Timeout period during which PIM interfaces
wait to receive Assert messages from the forwarder. The value is an integer rang
ing from 7 to 65535, in seconds. The default value is 180. ";
                        config "true";
                        default "180";
                        type uint32 {
                            range "7..65535";
                        }
                    }
                    leaf jpHoldTime {
                        description "Holdtime for a Join/Prune message sent by a
 PIM interface. The value is an integer ranging from 1 to 65535, in seconds. The
 default value is 210. Commonly, the holdtime is 3.5 times longer than the inter
val for all the interfaces to send Join/Prune messages.";
                        config "true";
                        default "210";
                        type uint32 {
                            range "1..65535";
                        }
                    }
                    leaf probeInterval {
                        description "Interval at which Probe messages are sent t
o an RP. The value is an integer ranging from 1 to 1799, in seconds. The default
 value is 5.";
                        config "true";
                        default "5";
                        type uint32 {
                            range "1..1799";
                        }
                    }
                    leaf jpTimerInterval {
                        description "Interval at which Join/Prune messages are s
ent. The value is an integer ranging from 1 to 2147483647, in seconds. It must b
e shorter than the holdtime of Join/Prune messages. The default value is 60. ";
```

```
                        config "true";
                        default "60";
                        type uint32 {
                            range "1..18000";
                        }
                    }
                    leaf drPriority {
                        description "DR election priority for a router. The grea
ter the value, the higher the priority. The value is an integer ranging from 0 t
o 4294967295. The default value is 1.";
                        config "true";
                        default "1";
                        type uint32 {
                            range "0..4294967295";
                        }
                    }
                    leaf helloHoldtime {
                        description "Timeout period during which a PIM interface
 waits to receive Hello messages from its neighbors. The value is an integer tha
t ranging from 1 to 65535, in seconds. It must be longer than the interval for P
IM neighbors to send Hello messages. The default value is 105. ";
                        config "true";
                        default "105";
                        type uint16 {
                            range "1..65535";
                        }
                    }
                    leaf helloLandelay {
                        description "Delay for transmitting Prune messages on a
shared network segment. The value is an integer ranging from 1 to 32767, in mill
iseconds. The default value is 500.";
                        config "true";
                        default "500";
                        type uint16 {
                            range "1..32767";
                        }
                    }
                    leaf helloInterval {
                        description "Specifies the interval at which Hello messa
ges are sent. The value is an integer ranging from 1 to 2147483647, in seconds.
It must be shorter than the timeout period of PIM neighbors. The default value i
s 30.";
                        config "true";
                        default "30";
                        type uint32 {
                            range "1..18000";
                        }
                    }
                    leaf helloOverride {
                        description "Interval at which the prune action in a Hel
lo message is overridden. The value is an integer ranging from 1 to 65535, in mi
lliseconds. The default value is 2500.";
                        config "true";
                        default "2500";
                        type uint16 {
                            range "1..65535";
                        }
                    }
                    leaf regCheckSum {
                        description "Configure a router to calculate the checksu
m based on all contents of a Register message. By default, the checksum is calcu
lated based on the header of a Register message only.";
```

```
                        config "true";
                        type uint16 {
                            range "1..65535";
                        }
                    }
                    leaf regSuppInterval {
                        description "Timeout period during which a router remain
s in the registration suppression state. The value is an integer ranging from 11
 to 3600, in seconds. The default value is 60.";
                        config "true";
                        default "60";
                        type uint32 {
                            range "11..3600";
                        }
                    }
                    leaf embeddedRp {
                        description "Enable the embedded RP function. By default
, the embedded RP function is enabled.";
                        config "true";
                        default "true";
                        type boolean {
                        }
                    }
                    leaf nbrCheckRecv {
                        description "Enable the PIM neighbor check function to c
heck whether received Join/Prune and Assert messages are sent from a PIM neighbo
r. If not, these messages are discarded. By default, the PIM neighbor check func
tion is disabled.";
                        config "true";
                        type boolean {
                        }
                    }
                    leaf nbrCheckSend {
                        description "Enable the PIM neighbor check function to c
heck whether Join/Prune and Assert messages are to be sent to an IPv4/v6 PIM nei
ghbor. If not, these messages are not sent. By default, the PIM neighbor check f
unction is disabled for Join/Prune and Assert messages to be sent.";
                        config "true";
                        type boolean {
                        }
                    }
                    leaf regPlyName {
                        description "Policy for filtering Register messages. The
 value is an integer ranging from 3000 to 3999, or a string of 1 to 32 case-sens
itive characters.";
                        config "true";
                        type string {
                            length "1..255";
                            pattern "^[^ ]+$";
                        }
                    }
                    leaf regPlyIpv6 {
                        description "Policy for filtering Register messages. The
 value is an integer ranging from 3000 to 3999, or a string of 1 to 32 case-sens
itive characters.";
                        config "true";
                        type string {
                            length "1..255";
                            pattern "^[^ ]+$";
                        }
                    }
```

```
                    leaf ssmPlyName {
                        description "Policy for limiting the range of valid SSM
group addresses. The value is an integer ranging from 2000 to 2999, or a string
of 1 to 32 case-sensitive characters.";
                        config "true";
                            type string {
                                length "1..255";
                                pattern "^[^ ]+$";
                            }
                    }
                    leaf ssmPlyIpv6 {
                        description "Range of SSM group addresses. The value is
an integer ranging from 2000 to 2999, or a string of 1 to 32 case-sensitive char
acters.";
                        config "true";
                        type string {
                            length "1..255";
                            pattern "^[^ ]+$";
                        }
                    }
                    leaf srcPlyName {
                        description "Policy for filtering multicast entries base
d on source addresses or based on both source and group addresses. The value is
an integer ranging from 2000 to 3999, or a string of 1 to 32 case-sensitive char
acters.";
                        config "true";
                            type string {
                                length "1..255";
                                pattern "^[^ ]+$";
                            }
                    }
                    leaf srcPlyIpv6 {
                        description "Policy for filtering multicast entries base
d on source addresses or based on both source and group addresses. The value is
an integer ranging from 2000 to 3999, or a string of 1 to 32 case-sensitive char
acters.";
                        config "true";
                            type string {
                                length "1..255";
                                pattern "^[^ ]+$";
                            }
                    }
                    leaf bsrPlyName {
                        description "Policy for limiting the range of valid BSR
addresses so that a router discards the messages received from the BSRs not in t
he specified address range. The value is an integer ranging from 2000 to 2999, o
r a string of 1 to 32 case-sensitive characters. By default, the range of valid
BSR addresses is not limited.";
                        config "true";
                            type string {
                                length "1..255";
                                pattern "^[^ ]+$";
                            }
                    }
                    leaf bsrPlyIpv6 {
                        description "Policy for limiting the range of valid BSR
addresses so that a router discards the messages received from the BSRs not in t
he specified address range. The value is an integer ranging from 2000 to 2999, o
r a string of 1 to 32 case-sensitive characters. By default, the range of valid
BSR addresses is not limited.";
                        config "true";
                            type string {
                                length "1..255";
```

```
                    pattern "^[^ ]+$";
                }
            }
```

```
                    leaf embRpPlyName {
                        description "Policy for limiting the range of multicast
groups to which an embedded-RP applies. The value is an integer ranging from 200
0 to 2999, or a string of 1 to 32 case-sensitive characters.";
                        config "true";
                            type string {
                                length "1..255";
                                pattern "^[^ ]+$";
                            }
                    }
                    leaf sourceLifeTime {
                        description "Timeout period for (S, G) entries on a rout
er. The value is an integer ranging from 60 to 65535, in seconds. The default va
lue is 210.";
                        config "true";
                        default "210";
                        type uint32 {
                            range "60..65535";
                        }
                    }
                    leaf sptDetInterval {
                        description "Interval for checking the multicast message
 rate is configured. When the multicast message rate reaches the specified thres
hold, a switchover from the RPT to the SPT is performed. The value is an integer
 ranging from 15 to 65535, in seconds.";
                        config "true";
                        default "15";
                        type uint32 {
                            range "15..65535";
                        }
                    }
                    leaf ipsecName {
                        description "Description of an SA. The value is a string
 of 1 to 15 characters.";
                        config "true";
                            type string {
                                length "1..15";
                                pattern "^[^ ]+$";
                            }
                    }
                    leaf ipsecType {
                        description "Hello options based on the IPsec type.";
                        config "true";
                        type enumeration {
                            enum forAll {
                                value "0";
                                description "Hello options based on the IPsec ty
pe.";
                            }
                            enum forHello {
                                value "1";
                                description "Hello options based on the IPsec ty
pe.";
                            }
                        }
                    }
                    leaf uniIpsecName {
                        description "Description of an SA. The value is a string
 of 1 to 15 characters.";
```

```
                        config "true";
                            type string {
                                length "1..15";
                                pattern "^[^ ]+$";
                            }
                    }
                    container PIM-Static-RPs {

                        container PIM-Static-RP {

                            leaf IPV4-staticRpAddr {
                                description "Specifies a static RP address.";
                                config "true";
                                mandatory "true";
                                type inet:ipv4-address;
                            }
                            leaf IPV6-staticRpAddr {
                                description "Specifies a static RP address.";
                                config "true";
                                mandatory "true";
                                type inet:ipv6-address;
                            }
                            leaf staticRpPlyName {
                                description "Static RP policy. The value is an i
nteger ranging from 2000 to 2999, or a string of 1 to 32 case-sensitive characte
rs.";
                                config "true";
                                type string {
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf staticRpPlyNameIpv6 {
                                description "Static RP policy. The value is an i
nteger ranging from 2000 to 2999, or a string of 1 to 32 case-sensitive characte
rs.";
                                config "true";
                                type string {
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf preference {
                                description "Whether the static RP is preferred.
 The value can be: 0: the static RP is not preferred. 1: the static RP is prefer
red. The default value is 0.";
                                config "true";
                                default "NotPrefer";
                                type enumeration {
                                    enum NotPrefer {
                                        value "0";
                                        description "Whether the static RP is pr
eferred. The value can be: 0: the static RP is not preferred. 1: the static RP i
s preferred. The default value is 0.";
                                    }
                                    enum Prefer {
```

```
                                        value "1";
                                        description "Whether the static RP is pr
eferred. The value can be: 0: the static RP is not preferred. 1: the static RP i
s preferred. The default value is 0.";
                                    }
                                }
                            }

                        }

                    }

                    container PIM-SPT-Switchs {

                        list PIM-SPT-Switch {

                            key "sptGrpPlyName";
                            max-elements "unbounded";
                            min-elements "0";
                            description "Rate threshold of multicast packets tha
t can trigger a receiver’s DR to send a Join message to a multicast source to se
t up an SPT. By default, the receiver’s DR performs the switchover from the RPT
to the SPT after receiving the first multicast data packet.";

                            leaf infinity {
                                description "The receiver’s DR never triggers th
e switchover from the RPT to the SPT.";
                                config "true";
                                type boolean {
                                }
                            }
                            leaf sptThreshHold {
                                description "Multicast packet rate threshold tha
t triggers the switchover from the RPT to the SPT. The value is an integer rangi
ng from 1 to 4194304, in kbit/s. ";
                                config "true";
                                default "1";
                                type uint32 {
                                    range "1..4194304";
                                }
                            }
                            leaf isACLEnable {
                                description "Whether to enable a group policy. "
;
                                type boolean {
                                }
                                config "true";
                                mandatory "true";
                            }
                            leaf sptGrpPlyName {
                                description "Name of a named ACL. The value is a
n integer ranging from 2000 to 2999, or a string of 1 to 32 case-sensitive chara
cters. If the parameter is not set, the threshold is applied to all multicast gr
oups. \"0\" means that no ACL is specified.";
                                config "true";
                                type string {
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
```

```
                            leaf sptGrpPlcyOrder {
                                description "Adjust the order of the ACLs in the
 group-policy list. If a group matches multiple ACLs, the threshold is selected
in the order specified by the order-value parameter. order-value specifies the u
pdated number. It is an integer. The value is any value other than original one
in the current group-policy list. If the parameter is not set, the order of the
ACLs in the group-policy list does not change.";
                                config "true";
                                default "4294967295";
                                type uint32 {
                                    range "1..4294967295";
                                }
                            }
                        }

                    }

                    container PIM_SM-Anycast-RPs {

                        container PIM_SM-Anycast-RP {

                            leaf IPV4-rpAddress {
                                description "Address of an IPV4 Anycast-RP peer"
;
                                config "true";
                                mandatory "true";
                                type inet:ipv4-address;
                            }

                            leaf IPV6-rpAddress {
                                description "Address of an IPV6 Anycast-RP peer"
;
                                config "true";
                                mandatory "true";
                                type inet:ipv6-address;
                            }

                            leaf local-IPV4-Address {
                                description "Address of an IPV4 Anycast-RP peer"
;
                                config "true";
                                mandatory "true";
                                type inet:ipv4-address;
                            }

                            leaf local-IPV6-Address {
                                description "Address of an IPV4 Anycast-RP peer"
;
                                config "true";
                                mandatory "true";
                                type inet:ipv6-address;
                            }

                            container PIM_SM-RP-Peers {

                                container PIM_SM-RP-Peer {
```

```
                                    leaf IPV4-rpPeer-Address {
                                        description "Address of an IPV4 Anycast-
RP peer";
                                        config "true";
                                        mandatory "true";
                                        type inet:ipv4-address;
                                    }
                                    leaf IPV6-rpPeer-Address {
                                        description "Address of an IPV6 Anycast-
RP peer";
                                        config "true";
                                        mandatory "true";
                                        type inet:ipv6-address;
                                    }
                                    leaf fwdSaSwt {
                                        description "Whether to forward to Anyca
st-RP peers source information learned through SA messages.";
                                        config "true";
                                        type boolean {
                                        }
                                    }
                                    leaf fwdPolicy {
                                        description "Policy for filtering SA mes
sages to be forwarded to Anycast-RP peers. The value is an integer ranging from
2000 to 2999, or a string of 1 to 32 case-sensitive characters.";
                                        config "true";
                                        type string {
                                            length "1..32";
                                            pattern "^[^ ]+$";
                                        }
                                    }
                                    leaf fwdPolicyIpv6 {
                                        description "Policy for filtering SA mes
sages to be forwarded to Anycast-RP peers. The value is an integer ranging from
2000 to 2999, or a string of 1 to 32 case-sensitive characters.";
                                        config "true";
                                        type string {
                                            length "1..32";
                                            pattern "^[^ ]+$";
                                        }
                                    }
                                }

                            }

                        }

                    }

                    container PIM_SM-Interfaces {
                        list PIM_SM-Interface {
                            key "ifName";
                            max-elements "unbounded";
                            min-elements "0";
                            description "Specifies an PIM_SM interface.";
```

```
                            leaf vrfName {
                                description "Name of an PIM SM instance. If the
name string is empty the instance means a public instance whose name is _public_
.";
                                config "true";
                                mandatory "true";
                                //default "_public_";
                                type string {
                                    length "0..32";
                                }
                            }
                            leaf addressFamily {
                                config "true";
                                mandatory "true";
                                type enumeration {
                                    enum ipv4unicast {
                                        value "0";
                                        description "Address family, which deter
mines whether an address belongs to IPv4 or IPv6.";
                                    }
                                    enum ipv6unicast {
                                        value "1";
                                        description "Address family, which deter
mines whether an address belongs to IPv4 or IPv6.";
                                    }
                                }
                            }
                            leaf ifName {
                                description "Interface name.";
                                config "true";
                                mandatory "true";
                                type ifName;
                            }
                            leaf pimsmEnable {
                                description "Enable PIM-SM on an interface.";
                                config "true";
                                mandatory "true";
                                type boolean {
                                }
                            }
                            leaf drPriority {
                                description "DR priority. The greater the value,
 the higher the priority. The value is an integer ranging from 0 to 4294967295.
The default value is 1.";
                                config "true";
                                default "1";
                                type uint32 {
                                    range "0..4294967295";
                                }
                            }
                            leaf helloInterval {
                                description "Interval at which Hello messages ar
e sent. The value is an integer ranging from 1 to 2147483647, in seconds. It mus
t be shorter than the timeout period of PIM neighbors. The default value is 30."
;
                                config "true";
                                default "30";
```

```
                                type uint32 {
                                    range "1..18000";
                                }
                            }
                            leaf helloHoldtime {
                                description "Timeout period during which a route
r waits for Hello messages sent from its PIM neighbors. The value is an integer
ranging from 1 to 65535, in seconds. It must be longer than the interval for PIM
 neighbors to send Hello messages. The default value is 105.";
                                config "true";
                                default "105";
                                type uint16 {
                                    range "1..65535";
                                }
                            }
                            leaf helloOverride {
                                description "Interval at which the prune action
in a Hello message is overridden. The value is an integer ranging from 1 to 6553
5, in milliseconds. The default value is 2500.";
                                config "true";
                                default "2500";
                                type uint16 {
                                    range "1..65535";
                                }
                            }
                            leaf helloLanDelay {
                                description "Period from the time when a router
receives a Prune message from a downstream device to the time when the router pe
rforms the prune action. The value is an integer ranging from 1 to 32767, in mil
liseconds. The default value is 500.";
                                config "true";
                                default "500";
                                type uint16 {
                                    range "1..32767";
                                }
                            }
                            leaf jpTimerInterval {
                                description "Interval at which Join/Prune messag
es are sent. The value is an integer ranging from 1 to 2147483647, in seconds. T
he default value is 60. The interval must be shorter than the holdtime of Join/P
rune messages.";
                                config "true";
                                default "60";
                                type uint32 {
                                    range "1..18000";
                                }
                            }
                            leaf jpHoldtime {
                                description "Holdtime for Join/Prune messages se
nt by a router. The value is an integer ranging from 1 to 65535, in seconds. The
 default value is 210. The holdtime must be longer than the interval at which Jo
in/Prune messages are sent.";
                                config "true";
                                default "210";
                                type uint32 {
                                    range "1..65535";
                                }
                            }
                            leaf jpPlyName {
                                description "Policy for filtering Join/Prune mes
sages. The value is an integer ranging from 2000 to 2999, or a string of 1 to 32
 case-sensitive characters. By default, no policy is configured to filter Join/P
rune messages.";
```

```
            config "true";
                type string {
```

```
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf jpPlyIpv6 {
                                description "Policy for filtering Join/Prune mes
sages. The value is an integer ranging from 2000 to 2999, or a string of 1 to 32
 case-sensitive characters. By default, no policy is configured to filter Join/P
rune messages.";
                                config "true";
                                type string {
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf jpAsmPlyName {
                                description "Policy for filtering ASM Join/Prune
 messages. The value is an integer ranging from 2000 to 2999, or a string of 1 t
o 32 case-sensitive characters. By default, no policy is configured to filter AS
M Join/Prune messages.";
                                config "true";
                                type string {
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf jpAsmPlyIpv6 {
                                description "Policy for filtering ASM Join/Prune
 messages. The value is an integer ranging from 2000 to 2999, or a string of 1 t
o 32 case-sensitive characters. By default, no policy is configured to filter AS
M Join/Prune messages.";
                                config "true";
                                type string {
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf jpSsmPlyName {
                                description "Policy for filtering SSM Join/Prune
 messages. The value is an integer ranging from 3000 to 3999, or a string of 1 t
o 32 case-sensitive characters. By default, no policy is configured to filter SS
M Join/Prune messages.";
                                config "true";
                                type string {
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf jpSsmPlyIpv6 {
                                description "Policy for filtering SSM Join/Prune
 messages. The value is an integer ranging from 3000 to 3999, or a string of 1 t
o 32 case-sensitive characters. By default, no policy is configured to filter SS
M Join/Prune messages.";
                                config "true";
                                type string {
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf nbrPlyName {
                                description "Policy for filtering PIM neighbors.
 The value is an integer ranging from 2000 to 2999, or a string of 1 to 32 case-
sensitive characters. By default, no policy is configured to filter PIM neighbor
```

```
s.";
                        config "true";
                            type string {
```

```
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf nbrPlyIpv6 {
                                description "Policy for filtering PIM neighbors.
 The value is an integer ranging from 2000 to 2999, or a string of 1 to 32 case-
sensitive characters. By default, no policy is configured to filter PIM neighbor
s.";
                                config "true";
                                type string {
                                    length "1..32";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf assertHoldtime {
                                description "Timeout period during which PIM int
erfaces wait to receive Assert messages from the forwarder. The value is an inte
ger ranging from 7 to 65535, in seconds. The default value is 180.";
                                config "true";
                                default "180";
                                type uint32 {
                                    range "7..65535";
                                }
                            }
                            leaf requireGenId {
                                description "Configure a PIM interface to deny H
ello messages that do not carry generation IDs. By default, a PIM interface rece
ives Hello messages that do not carry generation IDs.";
                                config "true";
                                type boolean {
                                }
                            }
                            leaf pimBsrBoundary {
                                description "Boundary for a PIM domain. The valu
e can be: 0: no boundary; 1: bidirectional domain boundary; 2: inbound unidirect
ional domain boundary.";
                                config "true";
                                default "None";
                                type enumeration {
                                    enum None {
                                        value "0";
                                        description "Boundary for a PIM domain.
The value can be: 0: no boundary; 1: bidirectional domain boundary; 2: inbound u
nidirectional domain boundary.";
                                    }
                                    enum Both {
                                        value "1";
                                        description "Boundary for a PIM domain.
The value can be: 0: no boundary; 1: bidirectional domain boundary; 2: inbound u
nidirectional domain boundary.";
                                    }
                                    enum Incoming {
                                        value "2";
                                        description "Boundary for a PIM domain.
The value can be: 0: no boundary; 1: bidirectional domain boundary; 2: inbound u
nidirectional domain boundary.";
                                    }
                                }
                            }
                            leaf bfdEnable {
                                description "Enable PIM BFD on an interface. By
default, PIM BFD is not enabled on an interface.";
```

```
                config "true";
```

```
                                type boolean {
                                }
                            }
                            leaf bfdMinTx {
                                description "Minimum interval at which BFD messa
ges are sent. The value is an integer ranging from 3 to 1000, in milliseconds. "
;
                                config "true";
                                default "0";
                                type uint32 {
                                    range "0..1000";
                                }
                            }
                            leaf bfdMinRx {
                                description "Minimum interval at which BFD messa
ges are received. The value is an integer ranging from 3 to 1000, in millisecond
s. ";
                                config "true";
                                default "0";
                                type uint32 {
                                    range "0..1000";
                                }
                            }
                            leaf bfdMultiplier {
                                description "Local detect multiplier for BFD mes
sages. The value is an integer ranging from 3 to 50. The default value is 3.";
                                config "true";
                                default "3";
                                type uint16 {
                                    range "3..50";
                                }
                            }
                            leaf isSilent {
                                description "Enable the PIM silent function on a
n interface. By default, PIM silent is disabled on an interface. ";
                                config "true";
                                default "false";
                                type boolean {
                                }
                            }
                            leaf isDrSwtDelay {
                                description "Enable DR switchover delay on an in
terface. By default, this function is disabled on an interface.";
                                config "true";
                                default "false";
                                type boolean {
                                }
                            }
                            leaf drSwtDelayInterval {
                                description "Delay for a DR switchover. The valu
e is an integer ranging from 10 to 3600. The default value is 10. The setting is
 ineffective if the DR switchover delay function is not enabled.";
                                config "true";
                                default "10";
                                type uint32 {
                                    range "10..3600";
                                }
```

```
                            }
                            leaf ipsecName {
                                description "Description of an SA. The value is
a string of 1 to 15 characters.";
                                config "true";
                                type string {
                                    length "1..15";
                                    pattern "^[^ ]+$";
                                }
                            }
                            leaf ipsecType {
                                description "Hello options based on the IPsec ty
pe.";
                                config "true";
                                type enumeration {
                                    enum forAll {
                                        value "0";
                                        description "Hello options based on the
IPsec type.";
                                    }
                                    enum forHello {
                                        value "1";
                                        description "Hello options based on the
IPsec type.";
                                    }
                                }
                            }
                        }
                    }

                }

            }
        }
    }
}
```

5. IANA Considerations

This draft includes no request to IANA.

6. Security Considerations

The data model defined does not create any security implications. This draft does not change any underlying security issues inherent in [I-D.ietf-netmod-routing-cfg].

7. Acknowledgements

TBD

8. References

8.1. Normative References

[I-D.ietf-netmod-routing-cfg]
Lhotka, L., "A YANG Data Model for Routing Management", draft-ietf-netmod-routing-cfg-15 (work in progress), May 2014.

[RFC3973] Adams, A., Nicholas, J., and W. Siadak, "Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)", RFC 3973, January 2005.

[RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 4601, August 2006.

[RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, August 2006.

[RFC4608] Meyer, D., Rockell, R., and G. Shepherd, "Source-Specific Protocol Independent Multicast in 232/8", BCP 120, RFC 4608, August 2006.

[RFC4610] Farinacci, D. and Y. Cai, "Anycast-RP Using Protocol Independent Multicast (PIM)", RFC 4610, August 2006.

[RFC5015] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, October 2007.

[RFC5059] Bhaskar, N., Gall, A., Lingard, J., and S. Venaas, "Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM)", RFC 5059, January 2008.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

8.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

Authors' Addresses

Yisong Liu
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: liuyisong@huawei.com

Feng Guo
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: guofeng@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 30, 2015

P. Pfister

October 27, 2014

Multicast enabled Home Network using PIM-SSBIDIR and HNCP
draft-pfister-homenet-multicast-00

Abstract

This document specifies a possible solution enabling multicast routing in a home network. It relies on the Source-Specific Bidirectional variant of the Protocol Independent Multicast routing protocol (PIM-SSBIDIR). HNCP is used to elect the Rendezvous Point address and a Proxy Controller connected to the Rendezvous Point Link. Additionally, PIM-SSBIDIR routers behavior is slightly modified on the Rendezvous Point Link so that the Proxy Controller may know the home-wide subscription state. Note that this document defines one single working solution to the stated problem: Inputs regarding other possibilities are welcome.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 30, 2015.

Copyright Notice

Table of Contents

1. Introduction

IP multicast is used not only for link-local communications but also for site-local exchanges (UPnP [UPnP] or TV over IP). Additionally, we can expect new connected objects will make use of this technique for diverse purposes. Most link types like Ethernet or 802.11 support link-local multicast natively, but a multicast routing protocol is required when multiple links are present. The Protocol Independent Multicast [RFC4601] is one of the most widely used multicast routing protocol. Unfortunately, home networks have some peculiarities that makes it unsuitable without changes.

This document lists the specificities of home networks regarding multicast, the problems resulting from these peculiarities and specifies how homenet routers must behave in order to enable multicast routing for both in-home and ISP originated traffic in multi-homed environments.

The solution makes use of the Source-Specific Bidirectional variant of the Protocol Independent Multicast routing protocol (PIM-SSBIDIR - [pim-ssbidir]) for routing multicast traffic inside the home, and PIM Border Proxies ([pim-border-proxy]) for subscribing on all uplink interfaces. Two new HNCP TLVs are defined. One is used in the Rendezvous Point Address (RPA) and Proxy Controller election process, the other is used for advertising PIM Border Proxies. In addition, PIM-SSBIDIR behavior is slightly modified on the RP Link allowing the Proxy Controller, connected on the RP Link, to acquire the home-wide subscription state.

This document specifies a functional solution enabling multicast routing in multi-homed home networks. Inputs regarding other possibilities are very welcome and expected, so the best design may be adopted.

2. Problem Analysis

Current home networks usually consist of a single link and therefore support link-local multicast using MLDv2 [RFC3810] or IGMPv3 [RFC3376] for both all-source (ASM) and source-specific (SSM) multicast. Future home networks ([I-D.ietf-homenet-arch]) will consist of multiple links, which means multicast routing will be required.

This section discusses home network requirements and problems related to multicast routing.

2.1. Requirements

Future home networks should at least provide the same multicast features as the existing home networks.

In-home traffic: Devices inside the home should be able to send and receive multicast traffic originated inside the home.

ISP to Home traffic: Devices inside the home should be able to receive multicast traffic coming from an ISP.

Home to ISP traffic: Although traffic originated inside the home MUST NOT be forwarded on external interfaces by default, it should not be precluded.

On top of that, home network environments add the following constraints, defined in the Homenet architecture document.

Autoconfiguration: It must function without human interactions.

Multi-Homing: It must support multiple uplinks and therefore multiple default routes.

This document makes no assumptions on the technique used by ISPs to provide multicast traffic. It allows border routers to act as PIM Border Proxies, translating the home-wide subscription state toward every multicast enabled home uplink. Border router default behavior SHOULD consist in using MLDv2 and IGMPv3 on all uplink interfaces. Similarly, multicast enabled ISPs SHOULD listen to MLDv2 and IGMPv3 subscriptions coming from CPEs, and provide multicast traffic accordingly.

Note that this document doesn't preclude the use of different techniques. For example, an ISP-provided CPE may be specifically configured to translate in-home multicast subscriptions into PIM requests on the ISP link. But this is outside the scope of this document.

2.2. Specific Problems

Both PIM Bootstrap Mechanism (PIM BSR - [RFC5059]) and the Homenet Configuration Protocol (HNCP - [I-D.ietf-homenet-hncp]) could be used for autoconfiguration purposes. As HNCP support is already required in all homenet routers, this document proposes to use it instead of its PIM equivalent.

PIM-SM [RFC4601], PIM-BIDIR [RFC5015] and PIM-SSM were designed to function in single routed domains. Extensions allow multiple domains to be connected one with each other, but they all require specific PIM interactions between the domains, and a non-ambiguous knowledge of the next hop router for any multicast source. Given homenet constraints, we encounter the two following problems.

2.2.1. Uplink subscription problem

Initially, PIM reacts to two types of events. MLDv2/IGMPv3 subscriptions and multicast traffic origination. As receiving traffic from the ISP requires a subscription to happen first, border routers need some knowledge of the home-wide subscription state. In a single-homed network, the border router could be the RP, but in a multi-homed network, this subscription information must be shared between all border routers.

2.2.2. Uplink source localization problem

In multi-homed networks, routers have multiple default routes (one for each uplink). Unicast routing is achieved by looking at both

source and destination addresses, but this technique can’t be used when forwarding Join/Prune messages.

When multiple default routes point to different next-hop routers, Source-Specific Join/Prune messages’ next-hop cannot be reliably determined. A possible but not very scalable solution would consist in letting all the routers dynamically know where are every sources located. This document proposes to makes use of PIM-SSBIDIR instead.

3. Homenet Multicast Support Specifications

3.1. General Requirements

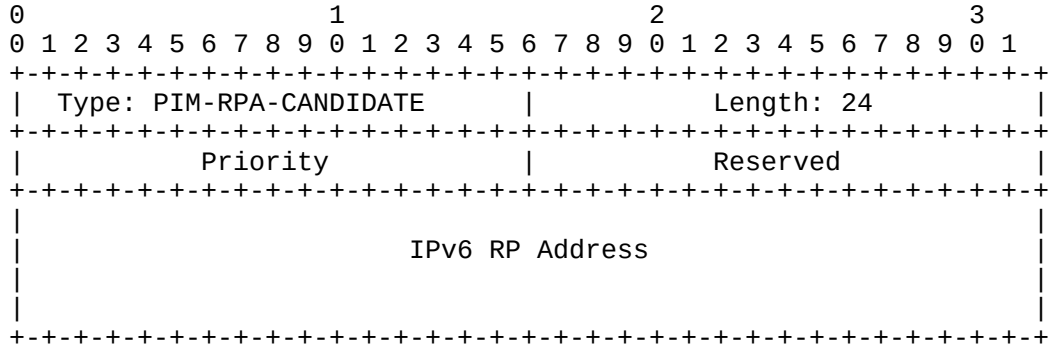
In order to deliver multicast traffic to subscribed devices, all homenet routers MUST implement PIM-SSBIDIR as well as the specifications presented in the present document.

Whenever the present document doesn’t conform to PIM specifications, behavior and configuration values described in this document take precedence.

3.2. Rendezvous Point Address Election Process

PIM-SSBIDIR and PIM-BIDIR both rely on the mapping between group ranges and Rendezvous Point Addresses. In PIM-SSBIDIR a Rendezvous point address doesn’t need to belong to an actual router but rather identify the Rendezvous Point Link. This is still true in the present document, but in addition to the RP Address, HNCP is used to elect a single Proxy Controller, directly connected to the RP Link.

In order to elect the RPA and Proxy Controller, the following HNCP TLV is defined.



PIM RPA Candidate TLV

The Rendezvous Point Address is chosen among all the advertised PIM RPA Candidate TLVs. The TLV with the highest priority is chosen first. In case of tie, the highest RPA address is preferred. The elected Proxy Controller is the router with the highest router ID advertising the elected PIM RPA Candidate TLV.

A router MUST start advertising a PIM RPA Candidate TLV (and thus candidate as Proxy Controller) whenever one of the two following condition is met.

- There is no currently advertised PIM RPA Candidate TLV network-wide.
- All the advertised PIM RPA Candidate TLVs have priority values lower than the one specified in the router's configuration and it is specifically stated by configuration that the router should try overcoming the currently elected RP.

A router MUST stop advertising a PIM RPA Candidate TLV whenever another advertised PIM RPA Candidate TLV takes precedence over its own one.

A router MUST NOT advertise more than one PIM RPA Candidate TLV. An advertised PIM RPA Candidate TLV MUST contain an IPv6 address known by all home routers and associated with a directly connected link. A Priority value of 0 SHOULD be used, unless stated otherwise by dynamic (DHCP, netconf, ...) or static (file) configuration.

When the RP Address is not valid anymore, the elected Proxy Controller MUST replace the advertised RP Address with a new, valid, RP Address. Such an event SHOULD be avoided. Therefore, an address with a long valid lifetime SHOULD be preferred.

3.3. PIM Border Proxy behavior

All routers with at least one uplink interface SHOULD behave as PIM Border Proxies, as specified in [pim-border-proxy], unless specified otherwise by static or dynamic configuration. They SHOULD proxy the received subscription state onto uplink interfaces for all groups of global scope.

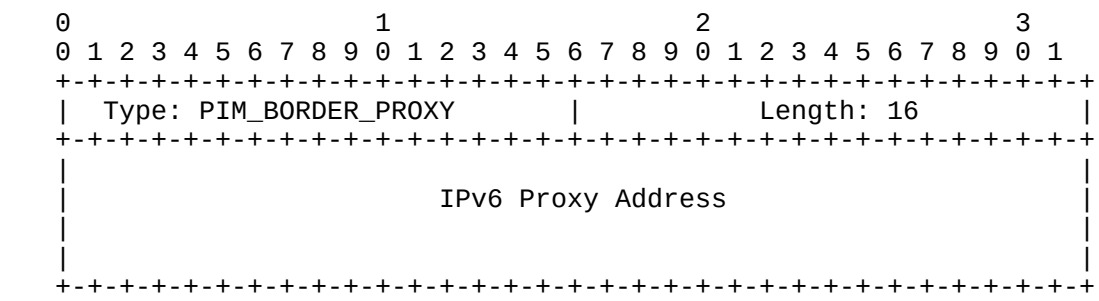
Multicast proxying is a local operation subject to numerous optimizations and configuration, particularly on ISP-provided CPEs. The following list specifies the default behavior.

- All groups with non-global scope SHOULD be ignored.

- The home-wide subscription state SHOULD be proxied on all uplink interfaces.

- The uplink default protocols are MLDv2 for IPv6 groups and IGMPv3 for IPv4 groups.

In addition, PIM Border Proxy routers MUST advertise the following TLV in their HNCP Node State.



PIM Border Proxy TLV

The elected Proxy Controller must behave as specified in [pim-border-proxy]. It MUST establish one peering for each address specified in PIM Border Proxy TLVs. It MUST reflect the home-wide subscription state toward all border proxies, computed based on all per-interface PIM downstream state machines and on-link local subscriptions, as if the RP was reachable on a virtual uplink interface.

3.4. PIM-SSBIDIR changes

This section specifies the changes made to PIM-SSBIDIR, required in the homenet context.

3.4.1. Router’s behavior on the RP Link

PIM-SSBIDIR always forwards the multicast traffic toward the RP Link and therefore never sends Join/Prune packets on the RP Link nor requires routers to listen to local subscriptions on the RP Link. But the elected Proxy Controller needs to know the home-wide subscription state. Which is why router’s behavior is modified on the RP Link.

All routers MUST operate the (*,G), (S,G) and (S,G,rpt) upstream state machines on all their interfaces, including the RP Link. On the RP Link, no DF Election process takes place. When sending Join/

Prune messages on the RP Link, the DF address is replaced with the RP Address.

The elected Proxy Controller MUST as well operate the downstream per-interface (*,G), (S,G) and (S,G,rpt) state machines on the RP Link, as well as enable multicast querying. Other routers connected to the RP Link SHOULD enable both downstream state machines and multicast querying as well in order to improve transition whenever the Proxy Controller would change.

3.4.2. Timing Considerations

PIM is an unreliable protocol. When a Join message is lost, the protocol waits for the next one, which by default comes after 60 seconds. A very typical use case for IP multicast is TV over IP, but we can't expect a user to wait 60 seconds when it changes the TV channel. Therefore, the default period between Join/Prune messages is reduced.

t_periodic: Default = 5 secs.

Similarly, PIM sends Hello messages every 30 seconds, which means dead neighbor detection occurs after 90 seconds. Therefore, the Hello period is reduced.

Hello_Period: Default = 10 secs.

4. Security Considerations

This document mostly relies on HNCP and PIM-SSBIDIR and therefore doesn't add much new threats.

The RP election process could be attacked whenever HNCP is not protected. Similarly, an attacker could advertise numerous PIM Border Proxy TLVs as a Deny of Service attack vector.

In order to operate securely, both HNCP and PIM-SSBIDIR should be secured.

5. IANA Considerations

IANA is kindly requested to reserve two new HNCP TLV identifiers:

- PIM Border Proxy TLV: PIM_BORDER_PROXY
- PIM RPA Candidate TLV: PIM-RPA-CANDIDATE

6. References

6.1. Normative References

[RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

[I-D.ietf-homenet-hncp] Stenberg, M. and S. Barth, "Home Networking Control Protocol", draft-ietf-homenet-hncp-00 (work in progress), April 2014.

[pim-ssbidir] Pierre Pfister, "Source Specific support for Bidirectional Protocol Independent Multicast", October 2014, <http://tools.ietf.org/html/draft-pfister-pim-ssbidir-00>.

[pim-border-proxy] Pierre Pfister, "Protocol Independent Multicast Border Proxying", October 2014, <http://tools.ietf.org/html/draft-pfister-pim-border-proxy-00>.

6.2. Informative References

[RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 4601, August 2006.

[RFC5015] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, October 2007.

[RFC5059] Bhaskar, N., Gall, A., Lingard, J., and S. Venaas, "Bootstrap Router (BSR) Mechanism for Protocol Independent Multicast (PIM)", RFC 5059, January 2008.

[UPnP] UPnP Forum, "Internet Gateway Device (IGD) Standardized Device Control Protocol V 1.0", November 2001.

[I-D.ietf-homenet-arch] Chown, T., Arkko, J., Brandt, A., Troan, O., and J. Weil, "IPv6 Home Networking Architecture Principles", draft-ietf-homenet-arch-11 (work in progress), October 2013.

Appendix A. Acknowledgments

The author would like to thank Steven Barth and Mohammed Hawari for their help in the specification and implementation process, as well as Mark Townsley, Stig Venaas, IJsbrand Wijnands and Markus Stenberg for their useful inputs.

Author's Address

Pierre Pfister
Paris
France

Email: pierre@darou.fr

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 30, 2015

P. Pfister

October 27, 2014

Protocol Independent Multicast Border Proxying
draft-pfister-pim-border-proxy-00

Abstract

This document describes an extension to the Protocol Independent Multicast (PIM) multicast routing protocol that enables PIM proxying from PIM domains toward other multicast domains. It relies on PIM Proxies located on domain borders and Proxy Controllers which can be located anywhere. This document describes how subscriptions can be proxied toward domain's border interfaces using MLDv2 and IGMPv3, but other protocols could be used as well. Once multicast traffic is received on a proxied interface, it can be forwarded as if originated by a directly connected source.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 30, 2015.

Table of Contents

1. Introduction

The Protocol Independent Multicast (PIM - [RFC4601]) routing protocol initially reacts to two different event types: multicast traffic reception from a connected source, and local multicast subscription (MLD or IGMP) state change on a connected link. This approach works when the network consists of a single PIM multicast domain, but does not when border routers are connected to different multicast domains. In such situations, the border routers need to be told to which group they should subscribe to on their egress interfaces before multicast traffic can be received.

This document defines PIM Proxy's and PIM Proxy Controller's behavior. There may be one or multiple instances of each in the same PIM domain. Each controller opens a TCP connection toward every proxy it wants to interact with and sends updates every time the Proxy's state should be updated.

In PIM-SM domains, one possible application of this extension consists in using the Rendezvous as a PIM Proxy Controller for all border routers, which in turn reflects network-wide subscription

state on domain's external interfaces using MLDv2 [RFC3810] or IGMPv3 [RFC3376]. In PIM-BIDIR domains, the same approach could be used, but would not support Source-Specific multicast. Instead, every router can reflect the local subscription state of links on which it is the Designated Forwarder.

This extension was designed in order to allow ISP originated traffic to reach subscribed hosts located inside a home network [I-D.ietf-homenet-arch]. Input from PIM and Homenet working group regarding other possible solutions enabling multicast routing in home networks are very welcome.

2. Protocol Specifications

This protocol allows controllers to push PIM subscription state toward proxies. The state a given controller pushes toward a given proxy may differ depending on the proxy, the local configuration, and may not reflect the local MLD or IGMP querier state. It is an arbitrary choice and depends on the purpose of the proxy.

The following state is maintained for every established TCP connection and every PIM Group/Source state ((*,G), (S,G,rpt), (S,G)). Although this is implementation specific, the generic behavior would consist in keeping the state for every Group/Source pair in their respective Encoded-Group and Encoded-Source formats (e.g. different states whether [B]idir bit is set or not).

Mode: One of { "NoInfo", "Join", "Prune" }.

If no state is stored for a given Group/Source pair, it is equivalent to the "NoInfo" state. Similarly, after being switched to "NoInfo" state, a stored state may be removed.

2.1. Proxy Controller Behavior

A proxy controller opens a TCP connection with every proxy it wants a peering with. When a new connection is opened, the complete state is first transmitted in order to synchronize the proxy with the controller's state. Then, each time some state is changed, an update is sent through the TCP connection.

2.2. Proxy Behavior

A proxy MUST listen on port PIM_PROXY_TCP_PORT for incoming TCP connections. When a new connection is established, a new subscription state is created.

2.3. PIM Proxy Message format

2.3.1. Message Header

PIM Proxy messages are transported over TCP and make use of the following TLV format.

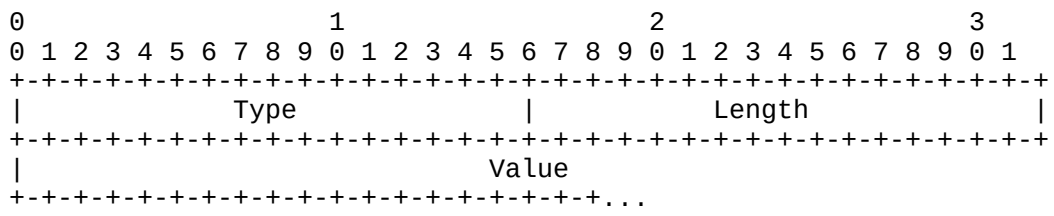


Figure 1

Type:
One of the defined message types.

Length:
The byte length of the value.

Value:
The value.

2.3.2. State Update Message

PIM Proxy State Update messages use the following format.

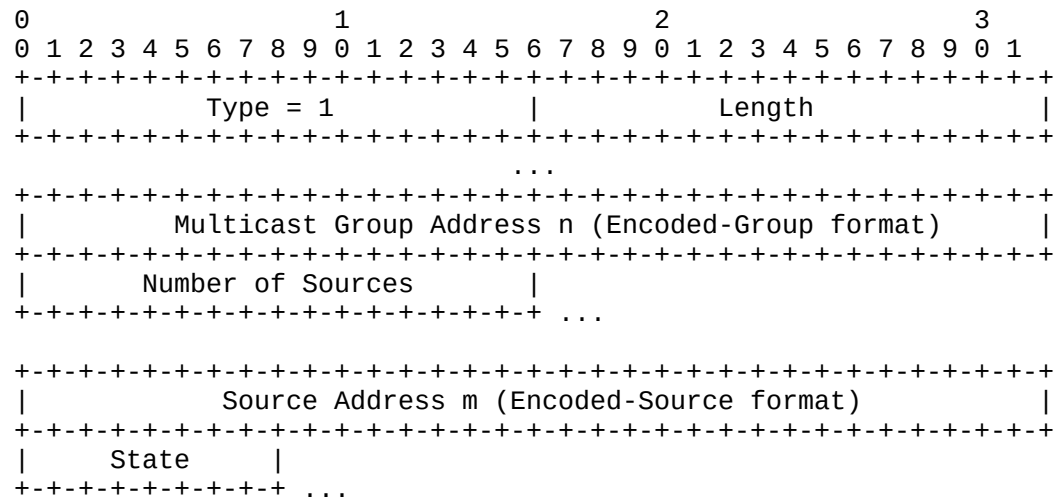


Figure 2

Type:
State Update message type (1)

Number of Sources:
The number of sources associated with the group.

State:
The state the controller wants the proxy to save for the given group/source pair.

* NoInfo (0)

* Join (1)

* Prune (2)

Upon reception of a State Update message, a proxy will switch the state associated with every Group/Source pairs included in the message to the state specified in the message.

2.3.3. Keep Alive Message

Keep Alive messages are used to keep the TCP connection open and have the following format (Most current TCP implementations support TCP keep-alives, but not all. The Keep-Alive TLV is specified because keeping the connection open is a requirement for not losing state).

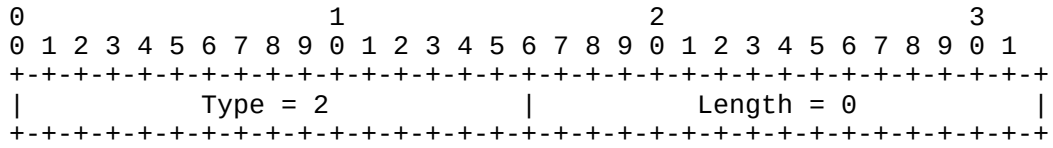


Figure 3

3. Different proxy types

This document does not intend to specify all the possible proxy behavior. Requests could be translated into MLD and IGMP, relayed toward another separated PIM domain, translated into another multicast delivery protocol, or even be used for monitoring purposes.

Any proxy behavior CAN be overridden by local policies. For instance, proxying behavior may depend on the group's scope or firewalling rules.

Once multicast traffic is requested on an egress interface, PIM should be used as usual in order to decide whether incoming traffic should be forwarded on an ingress interface.

3.1. IGMP/MLD proxy

This section describes how to translate a PIM Proxy group's state toward an MLDv2/IGMPv3 listener state. It can be used for both PIM-SM and PIM-BIDIR (states are considered regardless of the BIDIR bit).

According to PIM-SM and PIM-BIDIR specifications, (*,G) or (S,G) can only be in state "NoInfo" or "Join", and (S,G,rpt) can only be in state "NoInfo" or "Prune".

If (*,G) is set to "Join", the MLDv2/IGMPv3 group state should be set to "Exclude" and the Exclude Sources List should contain all sources S for which (S, G, rpt) is set to "Prune" and (S,G) is set to "NoInfo".

If (*,G) is set to "NoInfo", the MLDv2/IGMPv3 group state should be set to "Include" and the Include Sources List should contain all sources S for which (S, G) is set to "Join".

When multiple controllers are pushing state to the same proxy, the algorithm detailed in MLDv2 and IGMPv3 specifications should be used in order to merge the different requests.

3.2. IGMP/MLD controller

This section describes how to translate an MLDv2/IGMPv3 querier state into a PIM subscription state.

If the group's MLDv2/IGMPv3 state is "Include", the PIM state consists in (S,G) states set to "Join" for all S in the MLDv2/IGMPv3 source include list.

If the group's MLDv2/IGMPv3 state is "Exclude", the (*,G) state is set to "Join" and all (S,G,rpt) for S in the MLDv2/IGMPv3 source exclude list are set to "Prune".

4. Security Considerations

PIM Proxy messages are sent multiple hops away and are used in order to control other router's behavior. Attackers could open a connection from outside or inside the network and trigger multicast requests and forwarding. TLS or IPSec could be used in order to secure the connection. If not, connections should at least be filtered based on the controller's IP source address.

5. IANA Considerations

IANA is kindly requested to:

- Reserve a TCP port number for PIM Proxies.
- Create a new registry for PIM Proxy TLVs.
- Create a new registry for Group/Source states.

6. References

6.1. Normative References

[RFC5015] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, October 2007.

[RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 4601, August 2006.

[RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

6.2. Informative References

[I-D.ietf-homenet-arch]
Chown, T., Arkko, J., Brandt, A., Troan, O., and J. Weil, "IPv6 Home Networking Architecture Principles", draft-ietf-homenet-arch-11 (work in progress), October 2013.

Appendix A. Acknowledgments

The author would like to thank Steven Barth and Mohammed Hawari for their help in the protocol design and implementation process, as well as Mark Townsley, Stig Venaas, Markus Stenberg and IJsbrand Wijnands for their useful input.

Appendix B. Discussion

Another message type we would probably need is a "Traffic Present" message. Sent from the Proxy toward the Controller, it would let the controller take actions when the same traffic is provided by two different border routers. But that gets inefficient when there are more than one controller.

Author’s Address

Pierre Pfister
Paris
France

Email: pierre@darou.fr

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 30, 2015

P. Pfister

October 27, 2014

Source Specific support for Bidirectional Protocol Independent Multicast
draft-pfister-pim-ssbidir-00

Abstract

This document adds Source-Specific capabilities to the Bidirectional Protocol Independent Multicast (PIM-BIDIR) routing protocol. Similarly to PIM-BIDIR, multicast traffic is always forwarded to the RP Link, but packets are also filtered based on their source address and source-specific subscription state. This operation mode is backwards compatible with PIM-BIDIR, provides a simpler alternative to PIM-SM and does not suffer when the multicast source location cannot be determined.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 30, 2015.

Copyright Notice

Table of Contents

1. Introduction

The Protocol Independent Multicast (PIM - [RFC4601]) Sparse-Mode provides All-Source (ASM) and Source-Specific multicast (SSM) routing using an optimal source-specific routing path, but at the cost of two major drawbacks:

1. It relies on packet tunneling for All-Source Multicast (Pretty complex in terms of implementation).

2. It does not provide native multicast when some multicast source's location cannot be determined (e.g. when multiple default routes are in the RIB).

PIM BIDIR was specified as an alternative to PIM-SM: Simpler in terms of implementation and execution, at the cost of dropping source-specific multicast support. It is particularly suited in situations where multiple multicast sources also subscribe to other sources' sent packets. This document specifies the Source-Specific Bidirectional Mode (PIM-SSBIDIR), which adds source-specificity support to PIM-BIDIR. Unlike PIM-SM, all the traffic, including the source specific traffic, is forwarded on the RP-tree. Therefore, PIM-SSBIDIR does not provide source-specific path optimization, but still filters traffic according to multicast packet's source address, and thus reduces undesired traffic forwarding.

PIM-SSBIDIR, specified in this document, has two major advantages compared to PIM-SM.

1. Implementation and execution is simpler.

   * A single RP-rooted forwarding tree is used.

   * Assert state machine and messages are not used.

   * Register and Register-Stop messages are not used (No tunneling).

2. Packets are forwarded natively even when the source's location can't be determined.

This extension was designed for multi-homed home networks ([I-D.ietf-homenet-arch] - a typical case in which source-location cannot be determined for ISP-origininated traffic because of the multiple default routes). PIM-SM using PIM RPF Vectors [RFC5496] could be another alternative, but it is not clear how routers should behave when they receive conflicting RPF vectors. The protocol simplicity was also an important consideration as home routers usually have little resources and software reliability is important. Inputs from PIM and Homenet working group regarding other possible solutions are very welcome.

2. Protocol Overview

PIM-BIDIR specifies how a single RP-rooted tree can be maintained and used for forwarding multicast traffic to group subscribers. In the original document, only the (*,G) state machine was defined. This

document specifies (S,G) and (S,G,rpt) state machines when operating in BIDIR mode.

Similarly to PIM-SM, (S,G) Join/Prune messages allow subscribing to multicast traffic originated by a given source for a given group while (S,G,rpt) Join/Prune messages provide support for source-specific pruning. But unlike PIM-SM, SSBIDIR Join/Prune messages are always sent to the Designated Forwarder, and downstream multicast packets are always forwarded from the RP link to the subscribed hosts. Consequently, one major difference with PIM-SM is that (*,G) and (S,G,rpt) upstream states do not operate concurrently. A router's either operates in "include" mode using (S,G) states, or in "exclude" mode using (*,G) and (S,G,rpt) states.

Depending on the RP and sources locations, PIM-SSBIDIR does not always provide optimal routing of source-specific traffic. PIM-SM or PIM-SSM should be used in situations where source-specific path optimization is required.

PIM-SSBIDIR can operate with PIM-BIDIR neighbors. Neighbor's capabilities are determined using the new Source-Specific Bidirectional Capable PIM Hello option which is included in all Hello messages sent by SSBIDIR routers. When at least one neighbor, on a given link, does not support PIM-SSBIDIR, (S,G,rpt) states are not used, but (S,G) state can be. Additionaly, when the Designated Forwarder does not support PIM-SSBIDIR, (S,G) downstream states are converted into (*,G) subscriptions. In any case, requested multicast traffic is always forwarded to the subscribed hosts. Although BIDIR-only router presence may increase undesired traffic overhead.

3. Protocol Specifications

The specification of PIM-SSBIDIR is broken into several parts:

- Section 3.1 details the protocol state.
- Section 3.1.5 recalls that the Designated Forwarder election must take place.
- Section 3.2 specifies the multicast data forwarding rules.
- Section 3.3 specifies the PIM-SSBIDIR Join/Prune generation and processing rules.

3.1. PIM-SSBIDIR Protocol State

This section describes the state that must be kept by routers operating in PIM-SSBIDIR mode, in addition to the state specified in PIM-BIDIR specifications.

3.1.1. Per Neighbor State

Routers MUST keep track of neighbor's SSBIDIR and BIDIR capabilities as specified in PIM Hello BIDIR and SSBIDIR capable options (Section 3.4.2).

This state is used by "ssbidir_neighbor(N)" macro defined in section Section 3.1.6.

3.1.2. (*,G) State

The (*,G) state is similar to the (*,G) state described in PIM-BIDIR's specifications.

3.1.3. (S,G) State

For every source S and group G operating in SSBIDIR mode, routers keep the following state:

(S,G) state:

For each interface:

Local Membership:

- State: One of {"NoInfo", "Include"}

PIM (S,G) Join/Prune:

- State: One of {"NoInfo", "Join", "Prune-Pending"}
- Prune-Pending Timer
- Join/Prune Expiry Timer

Not interface specific:

Upstream (S,G) Join/Prune State:

- State: One of {"NotJoined", "Joined"}
- Upstream (S,G) Join/Prune Timer

Local membership is the result of the local membership mechanism (such as MLD [RFC3810] or IGMP [RFC3376]) running on that interface. This information is used by pim_include(S,G) macro described in Section 3.1.6.

PIM (S,G) Join/Prune state is the result of receiving PIM (S,G) Join/Prune messages on this interface, and is specified in Section 3.3.2.

The upstream (S,G) Join/Prune State reflects the state of the upstream (S,G) state machine and the upstream (S,G) Join/Prune Timer is used to send periodic (S,G) Joins and override Prune(S,G) messages from peers on an upstream interface. Details are specified in Section 3.3.5.

3.1.4. (S,G,rpt) State

For every group G and source S operating in SSBIDIR mode, routers keep the following state:

(S,G,rpt) state:

For each interface:

Local Membership:

- State: One of {"NoInfo", "Exclude"}

PIM (S,G,rpt) Join/Prune:

- State: One of {"NoInfo", "Prune", "Prune-Pending", "PruneTmp", "Prune-Pending-Tmp"}
- Prune-Pending Timer
- Join/Prune Expiry Timer

Not interface specific:

Upstream (S,G,rpt) Join/Prune State:

- State: One of {"NotPruned", "Pruned"}
- Upstream (S,G) Override Timer

Local membership is the result of the local membership mechanism (such as MLD or IGMP) running on that interface. This information is used by pim_include(S,G) macro described in Section 3.1.6.

PIM (S,G,rpt) Join/Prune state is the result of receiving PIM (S,G,rpt) Join/Prune messages on this interface, and is specified in Section 3.3.3.

The upstream (S,G,rpt) Join/Prune State reflects the state of the upstream (S,G,rpt) state machine and the upstream (S,G,rpt) Join/Prune Timer is used to send periodic (S,G,rpt) Prunes and override Prune(S,G,rpt) messages from peers on the upstream interface. The upstream state machine's behavior is specified in Section 3.3.7.

3.1.5. Designated Forwarder Election

PIM-SSBIDIR routers MUST comply to the Designated Forwarder (DF) election process, as defined in PIM-BIDIR specifications.

3.1.6. State Summerization Macro

This section describes the macros used by PIM-SSBIDIR state machines.

RPA(G), RPF_interface(rpa) and RPF_neighbor(rpa) are related to the Designated Forwarder election. They are defined in PIM-BIDIR's specifications.

The following pseudo code describes the set of interfaces requesting (S,G) traffic.

```
olist(S,G) =
    RPF_interface(RPA(G))
    (+) joins(S,G)
    (+) (joins(*,G) (-) prunes(S,G,rpt))
    (+) pim_include(S,G)
    (+) (pim_include(*,G) - pim_exclude(S,G))
```

The set "joins(*,G)" is the set of interfaces on which the router has received (*,G) Joins:

```
joins(*,G) =
    { all interfaces I such that
      I_am_DF(RPA(G),I) AND
      DownstreamJPState(*,G,I) is either Join or Prune-Pending }
```

The set "joins(S,G)" is the set of interfaces on which the router has received (S,G) Joins:

```
joins(S,G) =
    { all interfaces I such that
      I_am_DF(RPA(G),I) AND
      DownstreamJPState(S,G,I) is either Join or Prune-Pending }
```

The set "prunes(S,G,rpt)" is the set of interfaces on which the router has received (S,G,rpt) prunes:

```
prunes(S,G,rpt) =
    { all interfaces I such that
      I_am_DF(RPA(G),I) AND
      DownstreamJPState(S,G,rpt,I) is either Prune or PruneTmp }
```

The sets "pim_include(S,G)" and "pim_include(*,G)" indicate interfaces to which multicast traffic might be forwarded because of hosts that are local members on that interface. The set "pim_exclude(S,G)" is the set of interfaces where local members required to not receive traffic from source S:

```
pim_include(*,G) =
    { all interfaces I such that
      I_am_DF(RPA(G),I) AND
      local_receiver_include(*,G,I) }

pim_include(S,G) =
    { all interfaces I such that
      I_am_DF(RPA(G),I) AND
      local_receiver_include(S,G,I) }

pim_exclude(S,G) =
    { all interfaces I such that
      I_am_DF(RPA(G),I) AND
      local_receiver_exclude(S,G,I) }
```

Clauses "local_receiver_include(*,G,I)", "local_receiver_include(S,G,I)" and "local_receiver_exclude(S,G,I)" reflect local membership subscriptions, as defined in PIM-SM specifications.

In order to operate with BIDIR routers, SSBIDIR routers keep track of neighbors' capabilities. The clause "ssbidir_neighbor(N)" is TRUE if the last received Hello from neighbor N contained a Source-Specific Bidirectional Capable Hello Option. It is FALSE otherwise. In addition, the "ssbidir_link(I)" is TRUE if "ssbidir_neighbor(N)" is TRUE for any neighbor reachable through the interface I. It is FALSE otherwise. These clauses are used in JoinDesired(G), JoinDesired(S,G) and PruneDesired(S,G,rpt) macro specified in Section 3.3.

3.2. Forwarding Rules

Similarly to PIM-BIDIR, forwarding takes place on the RP-tree, defined as the set of uplink interfaces RPF_interface(RPA(G)) and interfaces where the router is elected Designated Forwarder.

The process of accepting a packet for forwarding, before building the outgoing interface list, is similar to PIM-BIDIR's as well.

In PIM-SSBIDIR, both group and source addresses are used in order to build the outgoing interface list, based on router's source-specific state. This differs from PIM-BIDIR, where only the group address is used.

A packet received on an interface for which we are the Designated Forwarder is always relayed upstream.

A packet received on the upstream interface, or on an interface on which we are the DF, is forwarded on all interfaces that requested the corresponding multicast traffic (Except the input interface).

In summary, on receipt of data from S to G on interface iif:

```
if( iif == RPA_interface(RPA(G)) || I_am_DF(RPA(G), iif) ) {
    oiflist = olist(S,G) (-) iif
    forward packet on all interfaces in oiflist
}
```

3.3. PIM Join/Prune Messages

PIM SSBIDIR adds source-specific states and messaging to the existing PIM-BIDIR specifications. Therefore, this document only specifies (S,G) and (S,G,rpt) state machines and do not modify the existing BIDIR (*,G) state machine.

3.3.1. Receiving (*,G) Join/Prune Messages

The per-interface (*,G) state machine is similar to the state machine described in PIM-BIDIR.

3.3.2. Receiving (S,G) Join/Prune Messages

When a router receives a Join(S,G) or Prune(S,G) for which SSBIDIR is enabled, it MUST first check if the group's [B]idir flag is set. If not, the packet MUST be discarded.

The per-interface state machine for receiving (S,G) Join/Prune has three states.

NoInfo (NI)
The interface has no (S,G) Join state and no timers running.

Join (J)
One of the routers reachable on interface I has requested receiving source-specific traffic sent by S to G. If the router is the DF on this interface (I_am_DF(RPA(G),I) is TRUE), the Join state will cause us to forward packets sent by S to G on this interface.

PrunePending (PP)
The router has received a a Prune(S,G) message and is waiting for the PrunePending timer to timeout before moving to the NoInfo state. During that lapse of time, forwarding takes place exactly like in the Join state and on-link routers can override the Prune(S,G) by sending a Join(S,G) message.

(S,G) per-interface state machine events, transitions and timer values are similar to PIM-BIDIR's (*,G) (By replacing (*,G) by (S,G)). Details can be found in Section 3.4.1 [1] from [RFC5015].

3.3.3. Receiving (S,G,rpt) Join/Prune Messages

When a router receives a Join(S,G,rpt) or a Prune(S,G,rpt) for which SSBIDIR is enabled, it MUST first check if the group's [B]idir flag is set. If not, the packet MUST be discarded.

When a message is parsed for a given group G, (*,G) Joins must be processed before processing any (S,G,rpt) Prunes. Transient state "PruneTmp" and "Prune-Pending-Tmp" are used during the parsing process.

The per-interface state machine for receiving (S,G,rpt) Join/Prune messages has four states.

NoInfo (NI)
The interface has no (S,G,rpt) Prune state and no timers running.

Prune (P)
The per-interface (*,G) state is set to Join and we received a Prune(S,G,rpt) message. It means that all interface's neighbors that are interested in (*,G) do not want to receive packets sent by source S. If the router is the DF on this interface (I_am_DF(RPA(G),I) is TRUE), the Prune state will cause us to *not* forward packets sent by S to G on this interface.

PrunePending (PP)

The router has received a Prune(S,G,rpt) message and is waiting for the PrunePending timer to timeout before moving to the Prune state. During that lapse of time, forwarding takes place exactly like the NoInfo state and neighboring routers can override the Prune(S,G,rpt) by sending a Join(S,G,rpt) message.

PruneTmp (P') This state is a transient state that is used as we parse a Join/Prune message that contains a (*,G) Join. A (S,G,rpt) Prune present in the same message will reinstate the Prune state. However, if we reach the end of the message without encountering such a (S,G,rpt) Prune, we will revert to NoInfo state in this state machine. For forwarding purposes, PruneTmp behaves exactly like Prune.

Prune-Pending-Tmp (PP') This state is a transient state that is identical to P' except that it is associated with the PP state rather than the P state. For forwarding purposes, PP' behaves exactly like PP state.

(S,G,rpt) per-interface state machine events, transitions and timer values are similar to PIM-SM (S,G,rpt) per-interface state. Details can be found in Section 4.5.4 [2] from [RFC4601].

In addition, the state machine needs to react to the "Stop Being DF on I" event. When this event occurs, all (S,G,rpt) states associated with I are moved to "NoInfo" and PrunePending and Expiry timers are stopped (if running).

3.3.4. Sending (*,G) Join/Prune Messages

The Upstream (*,G) state machine is similar to the state machine described in PIM-BIDIR's specifications.

For BIDIR compatibility support, the JoinDesired(G) macro must be modified as follows:

```
bool JoinDesired(G) {
    out = joins(*,G) (+) pim_include(*,G)
    if(!ssbidir_neighbor(RPF_neighbor(RPA(G)))) {
        out = out (+) {olist(S,G) for any source S}
    }
    return (out (-) RPF_interface(RPA(G))) != {}
}
```

In case the upstream neighbor is not SSBIDIR capable, downstream (S,G) Joins are translated into upstream (*,G) Joins.

3.3.5. Sending (S,G) Join/Prune Messages

The upstream (S,G) state machine holds join state from downstream routers and local membership information. It controls (S,G) Join/ Prune message emission.

This state machine has two possible states.

Not Joined
: The downstream state machines and local membership information do not indicate that the router needs to send (S,G) Joins or override other routers' (S,G) prunes.

Joined
: The downstream state machines and local membership information indicate that the router needs to send (S,G) Joins and override other routers' (S,G) prunes.

In addition, one timer JT(S,G) is kept that is used to trigger the sending of a Join(S,G) to the upstream Designated Forwarder RPF_neighbor(RPA(G)).

The detailed state machine is derived from PIM-BIDIR (*,G) upstream state machine by replacing (*,G) references with (S,G).

The JoinDesired(S,G) macro is defined as follows:

```
bool JoinDesired(S,G) {
    if(!ssbidir_neighbor(RPF_neighbor(RPA(G))) OR
        JoinDesired(G))
        return FALSE
    return (olist(S,G) (-) RPF_interface(RPA(G))) != {}
}
```

In case the upstream neighbor is not SSBIDIR capable, downstream (S,G) Joins are translated into upstream (*,G) Joins.

3.3.6. Sending (S,G,rpt) Join/Prune in (*,G) messages

Whenever a Join(*,G) message is sent in response to a Prune(*,G), it must include a Prune(S,G,rpt) for every (S,G,rpt) upstream state that is set to Pruned.

When sending a periodic Join(*,G), it must include a Prune(S,G,rpt) for every (S,G,rpt) state set to Pruned which Override Timer is not running.

Note that if the message construction is made after processing the Join(*,G) event in the upstream (S,G,rpt) state machine, the first paragraph can be ignored and the second paragraph applied in all cases. That is because the Join(*,G) reception event cancels all (S,G,rpt) Override Timers.

3.3.7. Sending triggered (S,G,rpt) Join/Prune Messages

The upstream (S,G,rpt) state machine holds join state from downstream routers and local membership information. It controls (S,G,rpt) Join /Prune message emission. It is only used when upstream (*,G) state is Joined.

The upstream (S,G,rpt) state machine has two states:

NotPruned
: The upstream (*,G) state is "Joined" but downstream routers and local membership indicates that at least one downstream interface should receive traffic sent from S to G.

Pruned
: The upstream (*,G) state is "Joined" but not a single downstream interface requested traffic sent from S to G.

ASNotJoined(G)
: The All-Sources (*,G) state is NotJoined. This state is kept because the (*,G) state machine takes care of sending (S,G,rpt) Prunes when JoinDesired(G) changes to true. Similarly, when JoinDesired(G) changes to false, all (S,G,rpt) states are moved back to ASNotJoined.

In addition, there is an (S,G,rpt) Override Timer, OT(S,G,rpt). In the NotPruned state, it is used to delay triggered Join(S,G,rpt) messages to prevent explosion of triggered messages. In the Pruned state, it is used to delay Prune(S,G,rpt) message origination whenever we received a Join(S,G,rpt) on that interface. This timer can be either cancelled, decreased to t_override or increased to t_suppressed (see PIM-BIDIR specifications for t_override and t_suppressed values).

The following tables show (S,G,rpt) upstream state machine transitions and events.

Event	State	
	ASNotJoined(NJ)	NotPruned(NP) & Pruned(P)
JoinDesired(G) -> true	If PD(S,G,rpt) -> P Else -> NP	_
JoinDesired(G) -> false	_	-> NP
Other events	Do nothing	

Event	State	
	NotPruned(NP)	Pruned(P)
PruneDesired(S,G,rpt) -> true	-> P; Cancel OT; Send Prune(S,G,rpt)	_
PruneDesired(S,G,rpt) -> false	_	-> NP; Cancel OT; Send Join(S,G,rpt)
See Prune(S,G,rpt) to RPF_DF(RPA(G))	Decrease OT to t_override	Cancel OT
See Join(S,G,rpt) to RPF_DF(RPA(G))	Cancel OT	Increase OT to t_suppress
See Prune(*,G) to RPF_DF(RPA(G))	Do nothing	Cancel OT
DF Changes	Cancel OT	Send Join/Prune to old/new DF
DF GenId changes	Do nothing	Decrease OT to t_override
Override Timer timeouts	Send Join(S,G,rpt)	Send Prune(S,G,rpt)

Upstream (S,G,rpt) state machine.

The (S,G,rpt) upstream state machine uses the following macro:

```
bool PruneDesired(S,G,rpt) {
    if(!ssbidir_link(RPF_interface(RPA(g))))
        return FALSE

    return (olist(S,G) (-) RPF_interface(RPA(G))) == {}
}
```

The PruneDesired(S,G,rpt) macro can return true even if JoinDesired(G) is false (And thus does not always means that Prune(S,G,rpt) should be sent upstream). That is to differentiate PruneDesired(S,G,rpt) transitions that are due to JoinDesired(G) transition and those which aren't.

3.4. SSBIDIR-PIM Packet Formats

This section describes the details of the packet formats for PIM-SSBIDIR control messages.

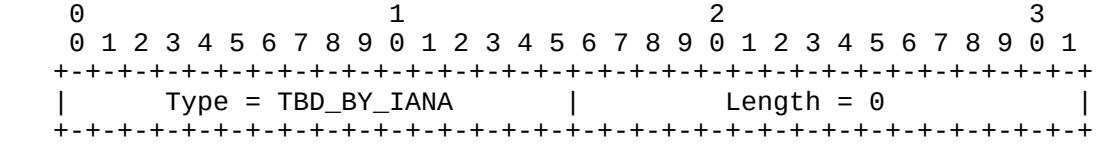
3.4.1. Encoded-Group Format

For all groups G operating in SSBIDIR mode, the [B]idirectional bit must be set in the Group-Encoded format included in Join/Prune messages.

3.4.2. SSBIDIR Capable PIM-Hello Option

SSBIDIR-PIM introduces one new PIM-Hello option.

o OptionType TBD_BY_IANA: Source-Specific Bidirectional Capable.



This option MUST be included in all PIM Hello messages sent by SSBIDIR capable routers.

SSBIDIR capable routers MUST as well include the Bidirection PIM-BIDIR Hello Option in sent Hello messages.

4. PIM-BIDIR Compatibility

Compatibility problems between SSBIDIR and BIDIR routers occur when a BIDIR router joins (*,G) while an SSBIDIR router is pruning (S,G,rpt) on the same link: The BIDIR router will not suppress (S,G,rpt) Prunes, which will prevent it from receiving traffic from source S. Similarly, if the Designated Forwarder is not SSBIDIR capable, it will reject (S,G) and (S,G,rpt) messages.

PIM-SSBIDIR interoperates with PIM-BIDIR by detecting BIDIR-only neighbors and suppressing undesired behaviors. When at least one neighbor, on a given link, is BIDIR-only, (S,G,rpt) messages are not used anymore (all traffic to G will be requested). When the upstream router is BIDIR-only, both (S,G,rpt) and (S,G) messages are not used anymore and only All-Source Multicast is supported.

PIM-BIDIR compatibility SHOULD be supported. If an implementer chooses not to implement the compatibility support, its implementation MUST behave as if all neighbors were SSBIDIR capable. Additionaly, an error MUST be logged in a rate-limited manner if a Hello message that does not include the SSBIDIR Capable Option is received.

PIM-BIDIR capable routers send (*,G) Join/Prune messages with the [B]idir bit set, but do not deal with (S,G) or (S,G,rpt) Join/Prune messages. The expected behavior from BIDIR capable routers in such situations is not specified. In order to support (S,G) subscriptions while operating in compatibility mode, we expect BIDIR implementations to silently ignore source-specific Join/Prunes.

5. Security Considerations

Similarly to PIM-SM, introducing Source-Specific support to PIM BIDIR makes it vulnerable to easy deny of service attacks by generating lots of (S,G) Join or (S,G,rpt) Prune states for different sources.

In order to operate securely, PIM messages SHOULD be authenticated and local subscriptions should be limited in rate and number (On a per-host basis if the link-layer provides authentication, on a per-link basis otherwise).

6. IANA Considerations

IANA is kindly asked to assign a new PIM-Hello Option Type to be used for Source-Specific Bidirectional BIDIR-PIM Hello options.

7. References

7.1. Normative References

[RFC5015] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, October 2007.

[RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 4601, August 2006.

[RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

7.2. Informative References

[I-D.ietf-homenet-arch] Chown, T., Arkko, J., Brandt, A., Troan, O., and J. Weil, "IPv6 Home Networking Architecture Principles", draft-ietf-homenet-arch-11 (work in progress), October 2013.

[RFC5496] Wijnands, IJ., Boers, A., and E. Rosen, "The Reverse Path Forwarding (RPF) Vector TLV", RFC 5496, March 2009.

7.3. URIs

[1] http://tools.ietf.org/html/rfc5015#section-3.4.1

[2] http://tools.ietf.org/html/rfc4601#section-4.5.4

Appendix A. Acknowledgments

The author would like to thank Steven Barth, Mohammed Hawari, Mark Townsley, Stig Venaas and IJsbrand Wijnands for their useful comments.

Author’s Address

Pierre Pfister
Paris
France

Email: pierre@darou.fr

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 27, 2015

S. Venaas
Cisco Systems

October 24, 2014

PIM Join Attribute Assignment Policy Update
draft-venaas-pim-join-attr-assignment-policy-00.txt

Abstract

This document updates the assignment policy of the PIM Join Attribute registry, changing the assignment policy from IETF Review to Specification Required.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 27, 2015.

Copyright Notice

Table of Contents

1. Introduction

This document changes the assignment policy of the PIM Join Attribute registry from IETF Review to Specification Required. The assignment polices are defined in [RFC5226]. With this change there is no longer a need for an RFC to be published to assign new join attributes, but a specification must be publicly available, and it will be reviewed by a Designated Expert as defined in [RFC5226].

2. Motivation

The assignment policy for the PIM Join Attribute registry was initially IETF Review as specified in [RFC5384]. However, this requires an RFC to be published prior to assignment. There are cases where there is a strong desire to deploy a new protocol or product relying on new Join Attributes without having to wait for the IETF standardisation process. By changing the policy to Specification Required, there will still be a public specification and a review process to ensure it is technically sound, but without waiting for an RFC to be published.

3. Review Criteria

The expert is expected to ensure that the specification is of sufficient quality to ensure interoperability between implementations, that it does not conflict with how PIM operates, that it will not cause potential deployment issues, and that it does not conflict with other multicast protocols or work in the IETF. Also, potential security implications must be considered.

In line with [RFC5384], join attribute specifications are required to specify the procedure to apply if there are multiple instances of the same attribute type. Also it should be considered whether it is appropriate for the attribute to be transitive or not. The conflict resolution procedure must also be considered. If a procedure is

specified, does it work as desired, or if not specified, is the default procedure specified in [RFC5384] appropriate for the attribute.

4. Security Considerations

This document by itself only changes a registry assignment policy which does not have any security issues in itself. When a Designated Expert reviews a new attribute specification, it is expected that the reviewer also considers the security aspects.

5. IANA Considerations

The assignment policy for the PIM Join Attribute registry is changed to Specification Required. IANA will need to update the registry description and accept and process assignment requests accordingly.

6. Acknowledgments

There have been discussions about assignment policies for the Join Attribute registry in the PIM WG, with several participants, William Atwood in particular. Based on this the author believes that this document is needed to change the assignment policy.

7. References

7.1. Informative References

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

7.2. Normative References

[RFC5384] Boers, A., Wijnands, I., and E. Rosen, "The Protocol Independent Multicast (PIM) Join Attribute Format", RFC 5384, November 2008.

Author’s Address

Stig Venaas
Cisco Systems
Tasman Drive
San Jose, CA 95134
USA

Email: stig@cisco.com

Network Working Group
Internet Draft

Category: Standard Track

L. Yong
W. Hao
D. Eastlake
Huawei
A. Qu
MetiaTek
J. Hudson
Brocade
U. Chunduri
Ericsson

Expires: May 2015 November 9, 2014

IGP Multicast Architecture

draft-yong-rtgwg-igp-multicast-arch-01

Abstract

This document specifies Interior Gateway Protocol (IGP) network architecture to support multicast transport. It describes the architecture components and the algorithms to automatically build a distribution tree for transporting multicast traffic and provides a method of pruning that tree for improved efficiency.

Status of this document

Table of Contents

1. Introduction

This document specifies Interior Gateway Protocol (IGP) network architecture to support multicast transport. It describes the architecture components and the algorithms to automatically build a distribution tree for transporting multicast traffic and provides a method of pruning that tree for improved efficiency.

An IGP network is built to transport unicast traffic. Traditionally, transporting multicast traffic relies on a protocol independent mechanism and a different protocol, i.e. PIM [RFC4601] [RFC5015]. The PIM protocol builds on top of IGP network and maintains its own states, which results longer convergence time for multicast traffic

Data Center infrastructure and advanced systems for cloud applications are looking for an IGP network to transport both unicast and multicast packets in a simpler and more efficient way than use of a separate protocol beyond IGP protocol. (see Section 1.1 for motivation)

This draft proposes the architecture and algorithms for an IGP based multicast transport. The architecture and algorithms automatically build a bi-directional distribution tree and pruned bi-directional tree for a multicast group without use of PIM. IGP protocol extension for this architecture is addressed in the [ISEXT].

1.1. Motivation

Network-as-a-service technically can be achieved by decoupling network IP space from service IP space as with a VxLAN [RFC7348] based network overlay. Decoupling network IP space from service IP address space also provides network agility and programmability to applications in a Data Center environment. To support all service applications, such IP network fabric must support both unicast and multicast. If network IP space is decoupled from service IP space, the network itself no longer needs manual configuration; automatically forming an IP network fabric can be done. The resulting "plug and play" can greatly simplify network operation.

With the goal of automation in forming a network fabric and support of any type of forwarding behavior the service applications require, IGP protocol should be extended to support:

1. Network formation

2. Multi destination distribution tree computation

Using external PIM prohibits the "automatic" nature requirement and results a longer convergence time of multicast transport than unicast transport because the convergence time for PIM is added to the basic IGP unicast route convergence time.

IGP based multicast reduces the number of protocols, states, and convergence time for multicast, which means a simpler underlay IP network that supports both unicast and multicast transport.

1.2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. IGP Architecture for Multicast Transport

An IGP multicast domain defined in this document contains edge routers and transit routers. Multicast source(s) and receiver(s) in a service space locally attach to edge routers or connect to edge routers through a layer 2 or layer 3 network that belongs to the same service space. When an ingress edge router receives a multicast packet from a multicast source in the service space, it replicates it along a pruned tree in the IGP domain. When an egress edge router receives a multicast packet from the IGP domain, it forwards the packet to the L2 or L3 service network that the receivers on and replicates the packet along the pruned tree in the domain. When a transit router receives a multicast packet from another router in the domain, it replicates the packet to its neighbor router(s) in the domain along a pruned tree.

An IGP multicast domain is used to carry L2 or L3 multicast traffic in a service (tenant) space in multi-tenant environment. Upon receiving a multicast packet from a source, the edge router first encapsulates the packet, adds its IP address as the source address and the corresponding underlay multicast IP address as the destination address on the encapsulated packet, then replicates it along a pruned tree. Egress edge router(s) decapsulate the packet before sending toward the receiver(s).

In an IGP multicast domain, each router has a unique IP address and the router IP address is advertised as a host address by IGP protocol. An IGP domain can be an IGP multicast domain if all

routers support the multicast capability described in this document; a subset of an IGP domain can be an IGP multicast domain where only some edge routers and transit routers have IGP multicast capability described in this draft and the draft [ISEXT]. In the case where the IGP multicast domain is subset of an IGP domain, a router in an IGP multicast domain must have at least one adjacency (next hop) to another router that is in the IGP multicast domain, that is, the IGP multicast domain must be connected. Configuring an IP tunnel between two routers in an IGP multicast domain can achieve this. How to configure such tunnel is outside the scope of this document.

In an IGP multicast domain, a default distribution tree is established automatically (see Section of 3.1). Operators may configure other distribution trees with different priorities in the domain as well and specify the associated multicast groups carried by these configured trees. By default, all the multicast groups use the default distribution tree.

The distribution tree computation algorithm is described in Section 3.2. The tree pruning for a particular multicast group is described in Section 3.3. Section 3.4 describes multiple trees to support one multicast group. Section 4 describes router forwarding procedures.

3. Computation Algorithms in IGP Multicast Domain

3.1. Automatic Tree Root Node Selection

By default the tree root is the router with the largest magnitude Router ID, considering the Router ID, i.e. router IPv4 address, to be an unsigned integer. Note that the algorithms in following sections use Router ID for router identifier, i.e. unique IP address assigned to a router in a IGP multicast domain.

Operators may configure a default tree root node (based on the topology) that takes precedence over the default tree root auto-calculated. This configured tree root node would advertise its IP address as the default tree root for all multicast groups that are not assigned to a distribution tree in a IGP multicast domain.

3.2. Distribution Tree Computation

The Distribution Tree Computation Algorithm uses the existing IGP Link State Database (LSDB). Based on the LSDB and shortest path algorithm, all routers in an IGP multicast domain calculate the distribution tree that has the default tree root node and reaches all the edge routers.

If an operator configures other distribution tree roots on other routers, the operator specifies what multicast groups use those trees and the tree root routers will advertise themselves as the tree root for those multicast groups by use of the new RTADDR TLV [ISEXT]. All routers in the domain will track the tree root nodes and calculate the path toward to each configured tree root node by using the shortest path algorithm, which form multiple distribute trees.

It is important that all the routers calculate the identical branches in a distribution tree in an IGP multicast domain. Section 3.2.1 and 3.3.2 specifies the tiebreaking rules for parent router selection in case of equal-cost path and for the link selection in case of multiple local links. Because link costs can be asymmetric, it is important for all tree construction calculations to use the cost towards the root.

3.2.1. Parent Selection

When there are equal costs from a potential child router to more than one possible parent router, all routers need to use the same tiebreakers. It is desirable to allow splitting traffic on as many links as possible in such situations when multiple distribution trees presents. This document uses the following tiebreaker rules:

If there are k distribution trees in the domain, when each router computes these trees, the k trees calculated are ordered and numbered from 0 to k-1 in ascending order according by root IP addresses.

The tiebreaker rule is: when building the tree number j, remember all possible equal cost parents for router N. After calculating the entire "tree" (actually, directed graph), for each router N, if N has "p" parents, then order the parents in ascending order according to the 7-octet IS-IS System ID considered as an unsigned integer, and number them starting at zero. For tree j, choose N’s parent as choice (j-1) mod p.

3.2.2. Parallel Local Link Selection

If there are parallel point-to-point links between two routers, say R1 and R2, these parallel links would be visible to R1 and R2, but not to other routers. If this bundle of parallel links is included in a tree, it is important for R1 and R2 to decide which link to use; if the R1-R2 link is the branch for multiple trees, it is desirable to split traffic over as many links as possible. However the local link selection for a tree is irrelevant to other Routers.

Therefore, the tiebreaking algorithm need not be visible to any Routers other than R1 and R2.

When there are L parallel links between R1 and R2 and they both are on K trees. L links are ordered from 0 to L-1 in ascending order of Circuit ID as associated with the adjacency by the router with the highest System ID, and K trees are ordered from 0 to K-1 in ascending order of root IP addresses. The tiebreaker rule is: for tree k, select the link as choice k mod L.

Note that if multiple distribution trees are configured in a domain or on a router, better load balance among parallel links through the tie-breaking algorithm can be achieved. Otherwise, if there is only one tree is configured, then only one link in parallel links can be used for the corresponding distribution tree. However, calculating and maintaining many trees is resource consuming. Operators need to balance between two.

Another alternative is to use a lower level link aggregation protocol, such as [802.1AX-2011] on the parallel point-to-point links between R1 and R2. They will then appear to be a single link to the IGP and it will be the link aggregation protocol that spreads traffic across the actual lower level parallel links.

3.3. Multiple Distribute Trees for a Multicast Group

It is possible that a multicast group is associated with multiple trees that may have the same or different priority. When a multicast group associates with more than one tree, all routers have to select the same tree for the group. The tiebreaker rules specified in PIM [RFC4601] are used here. They are:

- Perform longest match on group-range to get a list of trees.
- Select the tree with highest priority.
- If only one tree with the highest priority, select the tree for the group-range.
- If multiple trees are with the highest priority, use the PIM hash function to choose one. PIM hash function is described in section 4.1.1 in RFC 4601 [RFC4601].

3.4. Pruning a Distribution Tree for a Group

Routers prune the distribution tree for each associated multicast group, i.e. eliminating branches that have no potential downstream

receivers. Multi-destination packets SHOULD only be forwarded on branches that are not pruned. The assumption here is that a multicast source is also a multicast receiver but a multicast receiver may not be a multicast source.

All routers in the domain receive LSP messages with GRADD-TLV [RFC7176] from the edge routers, which indicate which multicast group that an edge router is the receiver. According that, the routers prune the corresponding distribution tree for each multicast group and maintain a list of adjacency interfaces that are on the pruned tree for a multicast group. Among these interfaces, one interface will be toward the tree-root router (unless the router is the root) and zero or more interfaces will be toward some edge routers.

4. Router Forwarding Procedures

4.1. Packet Forwarding Along a Pruned Distribution Tree

Forwarding a multi-destination packet follows the pruned tree for the group that the packet belongs to. It is done as follows.

- If the router receives a multi-destination packet with group IP address that does not associated with any configured tree, the packet MUST be considered associated with the default tree.

- Else check if the link that the packet arrives on is one of the ports in the pruned distribution tree. If not, the packet MUST be dropped.

- Else optionally perform RPF checking (section 4.4). If the check is performed and it fails, the packet SHOULD be dropped.

- Else the packet is forwarded onto all the adjacency interfaces in the pruned tree for the group except the interface where the packet receive.

4.2. Local Forwarding at Edge Router

Upon receiving a multicast packet, besides forwarding it along the pruned tree, an edge router may also need to forward the packet to the local hosts attached to it. This is referred to as local forwarding in this document. Local forwarding table and multicast forwarding table in IGP domain should be stitched at each edge router. Local forwarding table can be generated using IGMP/PIM protocol running in the network between host and the edge router.

The local group database is needed to keep track of the group membership of attached hosts. Each entry in the local group database is a [group, host] pair, which indicates that the attached hosts belonging to the multicast group. When receiving a multicast packet, the edge router forwards the packet to the host that match the [group, host] pair in the local group database.

The local group database is built through the operation of the IGMPv3 [RFC3376]. An edge router sends periodic IGMPv3 Host Membership Queries to attached hosts. Hosts then respond with IGMPv3 Host Membership Reports, one for each multicast group to which they belong. Upon receiving a Host Membership Report for a multicast group A, the router updates its local group database by adding/refreshing the entry [group A, host] pair. If at a later time Reports for Group A cease to be heard from the host, the entry is then deleted from the local group database. The edge router further sends the LSP message with GRADDR TLV to inform other routers about the group memberships in the local group database.

4.2.1. Overlay Multicast Transport

An IGP multicast domain may be used to carry overlay multicast traffic. [RFC7365] There are two architecture scenarios:

1) IGP multicast domain edge router separates with overlay network edge device [RFC7365]. Before multicast traffic is forwarded, Overlay network should trigger underlay multicast domain to construct multicast tree using IGMP protocol in beforehand. Group address in the protocol is underlay multicast group address. Outer layer traffic encapsulation is performed on the overlay network edge device, IGP multicast domain acts as pure underlay network.

2) IGP multicast domain edge router collapses with overlay network edge device. Before multicast traffic is forwarded, local connecting host should trigger underlay multicast domain to construct multicast tree using IGMP like protocol beforehand. Group address in the protocol is overlay multicast group address, edge router should map the group address into underlay multicast group address.

The IGP multicast domain can support both scenarios. To carry overlay multicast traffic, a (designated) edge router (see Section below on Multi-Homing Access) further necessarily maintains the mapping between an overlay multicast group and a underlying multicast group, and performs packet encapsulation/descapsulation upon receiving a packet from a host or the underlay IGP network. Mapping between an overlay multicast group and a underlay multicast group can be manually configured, automatically generated by an

algorithm at a (designated) edge router. The same edge router MUST be selected as the Designated Forwarder for the overlay multicast group and underlying multicast group that are associated. If multiple overlay multicast groups attach to same edge router sets, these overlay multicast groups can be mapped to the same underlying multicast group to reduce underlay network multicast forwarding table size on each router. The mapping method is beyond the scope of this document.

4.3. Multi-homing Access Through Active-active MC-LAG

A multicast group receiver may attach to multiple edge routers through an active-active MC-LAG [802.1AX-2011] to enhance reliability.

When a remote edge router ingresses a multicast packet w/ multicast group address from local multicast source, if all egress routers in an MC-LAG forward the packet to the local host (receiver), the host will receive multiple copies of the multicast frame from the remote multicast source. To avoid duplicated packets received from the IGP domain to a local network, a Designated Forwarder (DF) mechanism is required. All the edge routers associated to a same MC-LAG use the same algorithm to select one DF edge router for a multicast group. Each MC-LAG should be assigned with a unique MC-LAG identifier in an IGP multicast domain, which may be manually configured or automatically provisioned. When an edge router in a MC-LAG receives a multicast group receiver joining message using IGMP/PIM like protocols, it announces its self MC-LAG ID and the multicast group correspondence to other routers in its IGP LSP. After network state reaches steady state, all edge routers in a MC-LAG elect the same router as DF for each multicast group. Upon receiving a multicast packet from the domain, only the DF edge router will forward the packet towards the receiver. All non-DF edge routers do not forward the packet towards the receiver.

All edge routers, including DF and non-DF, can ingress the traffic to IGP domain as usual. DF and non-DF state has influence only on the egress multicast traffic forwarding process.

If a multicast group source host attaches to multiple edge routers through an active-active MC-LAG, loop prevention, i.e. the packet sent by source host loops back to the source host via the edge routers in a MC-LAG, is necessary. The solutions for two scenarios are described below.

- When the multicast IGP domain edge routers separate with overlay network edge devices that carry overlay network traffic, these routers don't replace traffic source IP address when they inject the traffic into IGP domain. In this case, edge routers should acquire multicast source IP address in beforehand using a mechanism like IGMPv3 explicit tracking, and then the source IP addresses are synchronized among each edge routers in same MC-LAG. Then same split-horizon mechanism described in the above section can be used.

- When the multicast IGP domain edge routers collapse with overlay network edge devices, the edge router connecting to multicast source performs multicast encapsulation when it injects local multicast traffic into the IGP domain, source IP is the edge router's IP. Each edge router tracks the IP address(es) associated with the other edge router(s) with which it has shared MC-LAG. When the edge router receives a packet from an IGP domain, it examines the source IP address and filters out the packet on all local interfaces in the same MC-LAG. With this approach, local bias forwarding is required on the ingress edge router. It performs replication locally to all directly attached receivers no matter DF or non-DF state of the out interface connecting to each receiver.

4.4. Reverse Path Forwarding Check (RPFC)

The routing transients resulting from topology changes can cause temporary transient loops in distribution trees. If no precautions are taken, and there are fork points in such loops, it is possible for multiple copies of a packet to be forwarded. If this is a problem for a particular use, a Reverse Path Forwarding Check (RPFC) may be implemented.

In this case, the RPFC works by a router determining for each port, based on the source and destination IP address of a multicast packet, whether the port is a port that router expects to receive such a packet. In other words, is there an edge router with reachability to the source IP address such that, starting at that router and using the tree indicated by the destination IP address, the packet would have arrived at the port in question. If so, it is further distributed. If not, it is discarded. An RPFC can be implemented at some routers and not at others.

5. Security Considerations

To come in future version

6. IANA Considerations

This document does not request any IANA action.

7. Acknowledgements

Authors like to thank Mike McBride and Linda Dunbar for their valuable inputs.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC2119, March 1997.

[RFC3376] Cain B., etc, "Internet Group Management Protocol, Version 3", rfc4604, October 2002

[RFC4601] Fenner, B., et al, "Protocol Independent multicast - Sparse Mode (PIM-SM): Protocol Specification", rfc4601, August 2006

[RFC5015] Handley, M., et al, "Bidirectional Protocol Independent Multicast (BIDIR-PIM", rfc5015, October 2007

[ISEXT] Yong, L., el al, "IS-IS Extension For Building Distribution Tree", draft-yong-isis-ext-4-distribution-tree, work in progress.

[802.1AX-2011] IEEE, "IEEE Standard for Local and metropolitan area networks - Link Aggregation", IEEE802.1AX, 2011

8.2. Informative References

[RFC7348] Mahalingam, M., Dutt, D., etc, "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC7348, 2014

[RFC7365] Lasserre, M., "Framework for DC Network Virtualization", RFC7364, 2014.

Authors' Addresses

Lucy Yong
Huawei USA

Phone: 918-808-1918
Email: lucy.yong@huawei.com

Weiguo Hao
Huawei Technologies
101 Software Avenue,
Nanjing 210012
China

Phone: +86-25-56623144
Email: haoweiguo@huawei.com

Donald Eastlake
Huawei
155 Beaver Street
Milford, MA 01757 USA

Phone: +1-508-333-2270
EMail: d3e3e3@gmail.com

Andrew Qu
MediaTek
San Jose, CA 95134 USA

Email: laodulaodu@gmail.com

Jon Hudson
Brocade
130 Holger Way
San Jose, CA 95134 USA

Phone: +1-408-333-4062
Email: jon.hudson@gmail.com

Uma Chunduri

Ericsson Inc.
300 Holger Way,
San Jose, California 95134
USA

Phone: 408-750-5678
Email: uma.chunduri@ericsson.com