Protocols for IP Multicast
Internet-Draft
Intended status: Informational
Expires: 7 November 2025

B. Fenner
S. Kumar
Arista Networks, Inc.
S. Venaas
Cisco Systems, Inc.

6 May 2025

Deterministic Upstream Neighbor Selection for PIM Joins
draft-ietf-pim-deterministic-ecmp-00

Abstract

In densely interconnected networks, a PIM node may have many choices as to what upstream neighbor to send a JOIN message to, for a given source and group. This document describes a mechanism for multiple nodes (e.g., leaf nodes in a data center) to pick the same upstream node (e.g., spine node) to avoid redundant traffic flows.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://fenner.github.io/pim-deterministic-ecmp/draft-ietf-pim-deterministic-ecmp.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-pim-deterministic-ecmp/.

Discussion of this document takes place on the Protocols for IP Multicast Working Group mailing list (mailto:pim@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/pim/. Subscribe at https://www.ietf.org/mailman/listinfo/pim/.

Source for this draft and an issue tracker can be found at https://github.com/fenner/pim-deterministic-ecmp.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

In a densely interconnected network, there may be many equal-cost paths to a given source or RP. RFC7761 is silent on the issue of how to choose among these, just indicating that RPF_interface(S) and RPF_interface(RP) have a single answer. If different leaf routers make different choices, then traffic can flow over extra paths.

This document introduces two mechanisms: one for two-tier networks and one for arbitrary multi-tier networks, to allow routers to make the same decision of which neighbor to use in an ECMP scenario. This eliminates undesired redundant traffic flow.

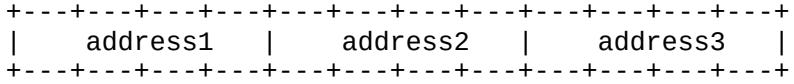
2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

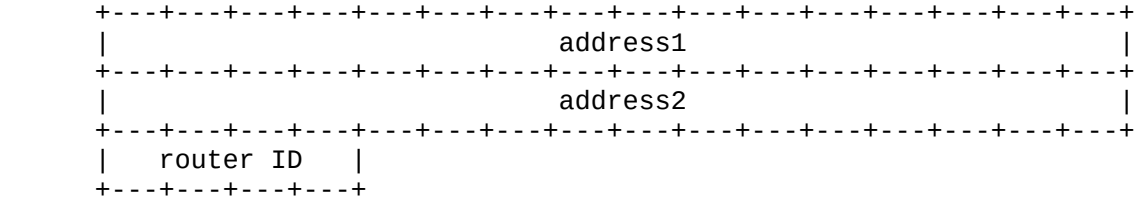
3. Hash Algorithm

In this document, the hash algorithm used is Bob Jenkins' one-at-a-time hash. This is a very high quality, but fast hash function. Wikipedia (https://en.wikipedia.org/wiki/Jenkins_hash_function#one_at_a_time) has one description of the algorithm. This hash function is defined on sequences of octets; it is performed across all of the addresses given in network byte order.

Pseudocode like hash(address1, address2, address3) conceptually lays out these addresses adjacent to each other in memory in network byte order, and performs a single hash operation across all 12 octets.



Similarly, when there are two IPv6 addresses and a router ID, we conceptually lay these out identically, simply with larger addresses, and perform the hash operation across all 36 octets.



When the hash involves a color, that 32-bit value in network byte order takes the place of the 32-bit router ID.

4. Deterministic Selection by Router-ID

We use the [RFC6395] Hello Option to allow multiple routers to hash a given (S,G) join to the same RPF neighbor. The procedure consists of two phases: first, we compute hash(S, G, routerId) for each eligible RPF neighbor, and select the highest hash value among this list. If there are multiple entries with the highest hash value, we re-hash among this sub-list using hash(S, G, local-information), and use the highest single resulting hash value. If multiple hops still hash to the same value, we simply take the first one in the list. This results in no coordination between nodes, since each node may have a different order for the list.

The local-information is a value local to the router that can be influenced by the deployment to have the same result between multiple peers - e.g., it could be an interface name, and the deployment on multiple routers uses the same interface to connect to the same peer. It could also be a locally-configured value on each interface, which results in higher configuration overhead but more deployment flexibility.

```
viaMultipathRouterId( source, group, vias )
    bestHash = 0
    bestVias = []
    for via in vias:
        routerId = getRouterId( via )
        curHash = hash( source, group, routerId )
        if curHash > bestHash:
            bestVia = [ via ]
            bestHash = curHash
        else if curHash == bestHash:
            bestVia.append( via )
    bestHash = 0
    if len( bestVia ) == 1:
        return bestVia[0]
    for via in bestVia:
        curHash = hash( source, group, local-information )
        if curHash > bestHash:
            bestVia = via

    return bestVia
```

Figure 1: Pseudocode for Deterministic Hashing based on Router ID

```
// pseudocode format TBD
```

5. Hello Option to Exchange Color

We describe a Hello Option to exchange "Color", an abstract notion of grouping of nodes. For example, in a 3-tier network, the routers in the middle tier could be colored by the spine to which they connect in the top tier. In this way, the color value presented to the leaf routers by the middle tier is a proxy for the routers in the top tier.

This Hello option should only be advertised "downwards" towards the lower levels of the tree.

5.1. Color Hello Option Message Format

The PIM Hello Option used to exchange Color values is shown below.

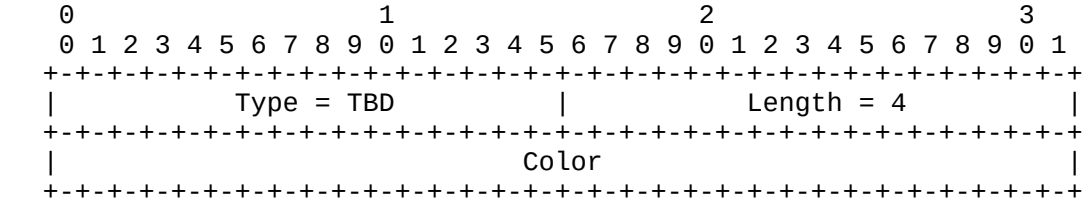


Figure 2: Color Hello Option

Type: TBD (see Section 8 and Appendix A)

Length: In bytes for the value part of the Type/Length/Value encoding. The Color will be 4 bytes long.

Color: The color value being advertised by this router.

6. Deterministic Selection by Color

TBD: If not all neighbors advertise color, do we pick from the subset that do, or we fall back to Section 4?

We use the above Hello Option to add an initial round of hashing, falling back to the algorithm in Figure 1 to break ties. With this mechanism, the first round is to calculate hash(S, G, color) for each eligible RPF neighbor, and select the highest hash value among this list. If there are multiple entries with the highest hash value, we re-hash among this sub-list with viaMultipathRouterId defined above.

```
viaMultipathColor( source, group, vias )
    bestHash = 0
    bestVias = []
    for via in vias:
        color = getNeighborColor( via )
        curHash = hash( source, group, color )
        if curHash > bestHash:
            bestVia = [ via ]
            bestHash = curHash
        else if curHash == bestHash:
            bestVia.append( via )
    bestHash = 0
    if len( bestVia ) == 1:
        return bestVia[0]
    return viaMultipathRouterId( source, group, bestVia )
```

Figure 3: Pseudocode for Deterministic Hashing based on Color

```
// pseudocode format TBD
```

7. Security Considerations

TODO Security

8. IANA Considerations

IANA is requested to allocate a value from the PIM-Hello Options registry as shown:

Value	Length	Name	Reference
TBD	4	Color	This Document

Table 1

9. Normative References

[RFC6395] Gulrajani, S. and S. Venaas, "An Interface Identifier (ID) Hello Option for PIM", RFC 6395, DOI 10.17487/RFC6395, October 2011, <https://www.rfc-editor.org/rfc/rfc6395>.

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <https://www.rfc-editor.org/rfc/rfc7761>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

Appendix A. Arista Networks Compatibility

This non-normative appendix describes the Arista Proprietary Color option, for the benefit of compatibility with the deployed base. A standards-compliant implementation SHOULD NOT emit or parse these options by default, but MAY have a configuration option to emit and parse these options on a given interface for interoperability.

A pair of PIM Hello options is required for compatibility with the deployed base of Arista EOS. Both types are allocated from the "Private Use" reserved range. The first option, with type 65001, only serves to indicate with a "magic number" that the type 65002 option is indeed the Arista Proprietary Color option (as opposed to some other Private Use).

These option formats are shown below.

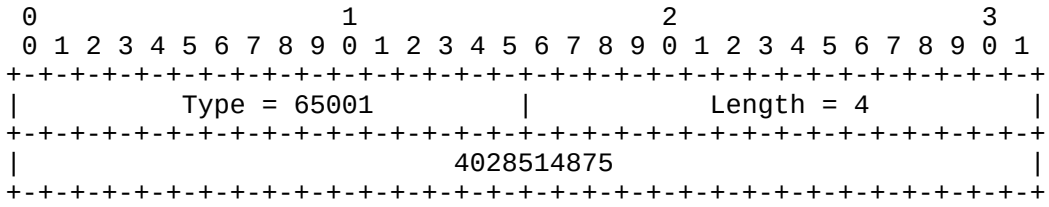


Figure 4: Enable Arista Proprietary Hello options

By including this PIM Hello option, with type 65001 and 32-bit value 4028514875, you indicate that the rest of the PIM Hello options that you are including are Arista-proprietary.

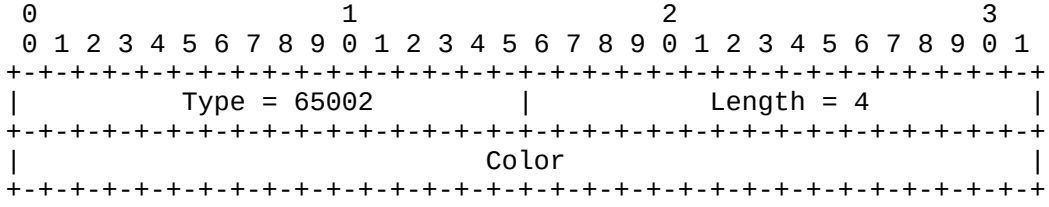


Figure 5: Arista Proprietary Color option

The Arista Proprietary Color option is identical to the option described in figure Figure 2, except for the Type field. It is only recognized if the Arista Proprietary Hello options are enabled by the option above.

A.1. Arista Color Hash Algorithm

The Arista-compatible hash algorithm stores the color in little-endian byte order when hashing.

address1	address2	color(little-endian)

When all routers in a set of vias are exchanging color information via the RFC-specified COLOR option, they MUST use the standard hash with the color in network byte order. When at least one router in a set of vias is exchanging color information via the Arista Proprietary Color option, they MUST use the Arista-compatible hash algorithm to compare colors.

Appendix B. Cisco Systems Compatibility

Cisco has, independently of this draft, implemented hashing based on Router-ID as discussed in Section 4, except for a few differences listed here.

The hash algorithm used is the RP hash defined in [RFC7761] section 4.7.2. Using this hash, the hash value is calculated by doing hash(router-id, 32, hash(group, 32, source)). In case multiple entries hash to the same value, the re-hash is using the interface ID announced in the [RFC6395] Hello Option rather than local information.

Hashing based on color is not implemented.

Appendix C. Sample Hash Values

Hashing with Router-ID:

```
hash( 192.0.0.2, 224.1.1.1, 10.0.0.1 ) = 361722995
hash( 192.0.0.2, 224.1.1.1, 10.0.0.2 ) = 4027394415
hash( 192.0.0.2, 224.1.1.1, 10.0.0.3 ) = 670832976
```

In this case, the neighbor with Router-ID 10.0.0.2 would be chosen.

Hashing with Color, Arista-compatible:

```
hash( 192.0.0.2, 224.1.1.1, 10 ) = 1271947512
hash( 192.0.0.2, 224.1.1.1, 20 ) = 3140394629
hash( 192.0.0.2, 224.1.1.1, 30 ) = 3675908571
```

In this case, the neighbor with color 30 would be chosen.

Hashing with Color, RFC-compatible:

```
hash( 192.0.0.2, 224.1.1.1, 10 ) = 3358313248
hash( 192.0.0.2, 224.1.1.1, 20 ) = 2756903791
hash( 192.0.0.2, 224.1.1.1, 30 ) = 2580115048
```

In this case, the neighbor with color 10 would be chosen.

Appendix D. Change history

This section is to be removed before publishing as an RFC.

D.1. Changes since draft-fenner-pim-deterministic-ecmp-00

* Remove the note about coming up with a different term than Color, it feels like Color will suffice
* Added memory layout for IPv6 address hashing
* Added Appendix B contributed by Stig about Cisco's deterministic hashing
* Corrected sample hash values in Appendix C

D.2. Changes since draft-fenner-pim-deterministic-ecmp-01

* Accepted as PIM WG work item

Acknowledgments

Thanks to Kalpesh Suthar of Juniper Networks for pointing out that the sample hash values in draft-fenner-pim-deterministic-ecmp-00 were calculated incorrectly.

Authors' Addresses

Bill Fenner
Arista Networks, Inc.
Email: fenner@fenron.com

Santosh Kumar
Arista Networks, Inc.
Email: skumar@arista.com

Stig Venaas
Cisco Systems, Inc.
Email: stig@cisco.com