Abstract

This document specifies multicast procedures with RIFT. Multicast in RIFT is similar to Bidirectional Protocol Independent Multicast (PIM-Bidir), with the Rendezvous Point Link (RP-Link) simulated by a spanning tree of some Top of Fabric (ToF) nodes and sub-ToF nodes.

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 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 https://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 January 9, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of
1. Introduction

Because of the simple north-south regular topology in Fat Tree networks, the PIM-Bidir [RFC5015] solution is extended for multicast in RIFT (referred to as MRIFT in this document). The following is a summary of the changes and adaptations compared to PIM-Bidir.

With PIM-Bidir, PIM joins are sent towards a Rendezvous Point Address, which could be an address not belonging to any router. The RPA does belong to a RP Link (RPL), which could be attached to a single router or multiple routers (e.g., RPL is a LAN). With MRIFT, there is no concept of RPA any more (joins are simply sent northbound). The joins are terminated on some sub-ToF nodes and the RPL is simulated by a spanning tree among some ToF and sub-ToF nodes.

Instead of (*,G) trees in PIM-Bidir, MRIFT uses (*,G-Prefix) trees, where the G-Prefix could be *, G, or anything in between (e.g., 225.1.1.0/24). For light flows, they could just follow the (*,*) tree. For heavy flows, individual (*,G) trees could be built. For medium flows, some (*,G-prefix) trees could be shared. All the First Hop Routers (FHRs, connecting to sources) and the Last Hop Routers (LHRs, connecting to receivers) of a particular (*,G) flow must agree on whether a (*,*) or (*,G) or (*,G-prefix) tree is used for the flow so that they all join the same tree. This is done via out of band control outside the scope of this document.

Because of the rich connections in Fat Trees, a router has to choose one of its many north neighbors to send join to. This is done through hashing. The hashing algorithm should lead to several but not too many routers choosing the same north neighbor, so that fewer
routers are involved in multicast traffic forwarding, yet none of those routers are overburdened by replicating to too many downstream neighbors.

Instead of PIM messages, RIFT’s own TIEs are used. This is similar to the concept in [draft-zzhang-pim-pds]. Specifically, RIFT Policy Guided Prefixes (PGP) [draft-atlas-rift-pgp] are used. The TIEs are consumed, processed at each hop and then regenerated for the next hop.

When a join reaches a sub-ToF node, the normal join process stops. This forms a sub-tree rooted at this sub-ToF node. Multiple sub-trees of the same tree may be joined by a single ToF node, or they may have to be connected by a spanning tree serving as the RPL. For example, in the following topology, in normal situations the two sub-tree roots for the two pods, say Spine111 and Spine121, may be joined by ToF21, but if the ToF21-Spine121 link is down, then ToF22 may be used, and if the ToF22-Spine111 link is also down, then Spine111 and Spine121 will have to be joined via Spine111-ToF21-Spine112-ToF22-Spine121.
The following algorithm is used to form the spanning tree.

1. Each sub-tree root (a sub-ToF node) hashes to a ToF neighbor as its parent and advertises the parent’s SystemID in a N-TIE for the tree. This allows different trees to have different RPLs for load-balancing. In the above example, Suppose Spine111 advertises its choice of ToF21, and Spine121 advertises its choice of ToF22.

2. Each ToF node advertises the highest SystemID in its S-TIE for a tree, of all the ToF nodes chosen and advertised by sub-ToF nodes for the same tree. The S-TIE also includes the SystemID of the sub-ToFs who made the choice. A ToF node knows the choices
either because it is the neighbor of a sub-ToF who made a choice (e.g. ToF21 knows Spine121’s choice is ToF22 because of Spine121’s N-TIE), or because it received another ToF’s S-TIE reflected by a common south neighbor (e.g. if the ToF21-Spine121 link is down, ToF21 still knows ToF22 was chosen by Spine121 because of ToF22’s S-TIE for the tree reflected by Spine122).

3. If a sub-ToF node sees ToF nodes with higher SystemIDs (than that of its own chosen parent) advertised for the tree, it reparents to the one that is its neighbor and has the highest SystemID, and re-advertises the new parent. In the above example, Spine111 will reparent to ToF22, assuming ToF22 has higher SystemID than ToF21.

4. A ToF parent (with remaining sub-ToF children who could not reparent) joins towards the ToF parent with the highest SystemID (as determined in step #2) via a south neighbor by including in its S-TIE for the tree the identity of the south neighbor, who either advertised its choice of the highest SystemID ToF parent, or reflected a ToF node’s S-TIE about sub-ToF node’s choice of the highest SystemID ToF parent. In the above example, if the ToF22-Spine111 link is down, ToF21 will join ToF22 either via Spine112 or Spine122.

The above procedures may repeat multiple times before the spanning tree is settled; unless the connections among ToF and sub-ToF nodes are badly broken, the process should be fairly simple.

2. Specifications

More details will be specified in future revisions.

3. Security Considerations

To be provided.

4. Acknowledgements

The authors thank Bruno Rijsman and Antoni Przygenda for their review and suggestions.

5. References

5.1. Normative References

[I-D.ietf-rift-rift]
5.2. Informative References


Authors’ Addresses

Zhaohui Zhang
Juniper Networks
EMail: zzhang@juniper.net

Pascal Thubert
Cisco Systems, Inc
EMail: pthubert@cisco.com