Workgroup: RIFT WG
Published: 16 December 2021
Intended Status: Informational
Expires: 19 June 2022
Authors: Yuehua. Wei, Ed.   Zheng. Zhang
ZTE Corporation   ZTE Corporation
Dmitry. Afanasiev   P. Thubert   T. Przygienda
Yandex   Cisco Systems   Juniper Networks

RIFT Applicability

Abstract

This document discusses the properties, applicability and operational considerations of RIFT in different network scenarios. It intends to provide a rough guide how RIFT can be deployed to simplify routing operations in Clos topologies and their variations.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction
2. Terminology
3. Problem Statement of Routing in Modern IP Fabric Fat Tree Networks
4. Applicability of RIFT to Clos IP Fabrics
   4.1. Overview of RIFT
   4.2. Applicable Topologies
      4.2.1. Horizontal Links
      4.2.2. Vertical Shortcuts
      4.2.3. Generalizing to any Directed Acyclic Graph
      4.2.4. Reachability of Internal Nodes in the Fabric
   4.3. Use Cases
      4.3.1. Data Center Topologies
      4.3.2. Metro Fabrics
      4.3.3. Building Cabling
      4.3.4. Internal Router Switching Fabrics
      4.3.5. CloudCO
5. Operational Considerations
   5.1. South Reflection
   5.2. Suboptimal Routing on Link Failures
   5.3. Black-Holing on Link Failures
   5.4. Zero Touch Provisioning (ZTP)
   5.5. Mis-cabling Examples
   5.6. Positive vs. Negative Disaggregation
   5.7. Mobile Edge and Anycast
   5.8. IPv4 over IPv6
   5.9. In-Band Reachability of Nodes
   5.10. Dual Homing Servers
   5.11. Fabric With A Controller
      5.11.1. Controller Attached to ToFs
      5.11.2. Controller Attached to Leaf
   5.12. Internet Connectivity Within Underlay
      5.12.1. Internet Default on the Leaf
      5.12.2. Internet Default on the ToFs
   5.13. Subnet Mismatch and Address Families
   5.14. Anycast Considerations
   5.15. IoT Applicability
   5.16. Key Management
6. Security Considerations
7. IANA Considerations
8. Acknowledgments
9. Contributors
10. Normative References
11. Informative References
Authors' Addresses

1. Introduction

This document discusses the properties and applicability of "Routing in Fat Trees" [RIFT] in different deployment scenarios and highlights the operational simplicity of the technology compared to traditional routing solutions. It also documents special considerations when RIFT is used with or without overlays and/or controllers, and how RIFT identifies topology mis-cablings and reroutes around node and link failures.

2. Terminology

Clos/Fat Tree:

This document uses the terms Clos and Fat Tree interchangeably whereas it always refers to a folded spine-and-leaf topology with possibly multiple Points of Delivery (PoDs) and one or multiple Top of Fabric (ToF) planes.

Directed Acyclic Graph (DAG):

A finite directed graph with no directed cycles (loops). If links in a Clos are considered as either being all directed towards the top or vice versa, each of such two graphs is a DAG.

Disaggregation:

Process in which a node decides to advertise more specific prefixes Southwards, either positively to attract the corresponding traffic, or negatively to repel it. Disaggregation is performed to prevent black-holing and suboptimal routing to the more specific prefixes.

TIE:

This is an acronym for a "Topology Information Element". TIEs are exchanged between RIFT nodes to describe parts of a network such as links and address prefixes. A TIE has always a direction and a type. North TIEs (sometimes abbreviated as N-TIEs) are used when dealing with TIEs in the northbound representation and South-TIEs (sometimes abbreviated as S-TIEs) for the southbound equivalent. TIEs have different types such as node and prefix TIEs.

Node TIE:

This stands as acronym for a "Node Topology Information Element", which contains all adjacencies the node discovered and information about the node itself. Node TIE should NOT be confused with a North TIE since "node" defines the type of TIE rather than its direction. Consequently North Node TIEs and South Node TIEs exist.

Prefix TIE:

This is an acronym for a "Prefix Topology Information Element" and it contains all prefixes directly attached to this node in case of a North TIE and in case of South TIE the necessary default routes and disaggregated routes the node advertises southbound.

South Reflection:

Often abbreviated just as "reflection", it defines a mechanism where South Node TIEs are "reflected" from the level south back up north to allow nodes in the same level without East- West links to "see" each other's node Topology Information Elements (TIEs).

LIE:

This is an acronym for a "Link Information Element" exchanged on all the system's links running RIFT to form ThreeWay adjacencies and carry information used to perform Zero Touch Provisioning (ZTP) of levels.

Shortest-Path First (SPF):

A well-known graph algorithm attributed to Dijkstra that establishes a tree of shortest paths from a source to destinations on the graph. SPF acronym is used due to its familiarity as general term for the node reachability calculations that RIFT can employ to ultimately calculate routes of which Dijkstra algorithm is a possible one.

North SPF (N-SPF):

A reachability calculation that is progressing northbound, as example SPF that is using South Node TIEs only. Normally it progresses a single hop only and installs default routes.

South SPF (S-SPF):

A reachability calculation that is progressing southbound, as example SPF that is using North Node TIEs only.

3. Problem Statement of Routing in Modern IP Fabric Fat Tree Networks

Clos [CLOS] topologies (called commonly a fat tree/network in modern IP fabric considerations as homonym to the original definition of the term Fat Tree [FATTREE]) have gained prominence in today's networking, primarily as a result of the paradigm shift towards a centralized data-center based architecture that deliver a majority of computation and storage services.

Current routing protocols were geared towards a network with an irregular topology with isotropic properties, and low degree of connectivity. When applied to Fat Tree topologies:

- They tend to need extensive configuration or provisioning during bring up and adding or removing Rift nodes from the fabric.

- All nodes including spine and leaf nodes learn the entire network topology and routing information, which is in fact, not needed on the leaf nodes during normal operation.

- They flood significant amounts of duplicate link state information between spine and leaf nodes during topology updates and convergence events, requiring that additional CPU and link bandwidth be consumed. This may impact the stability and scalability of the fabric, make the fabric less reactive to failures, and prevent the use of cheaper hardware at the lower levels (i.e. spine and leaf nodes).

4. Applicability of RIFT to Clos IP Fabrics

Further content of this document assumes that the reader is familiar with the terms and concepts used in OSPF [RFC2328] and IS-IS [ISO10589-Second-Edition] link-state protocols. The sections of RIFT [RIFT] outline the requirements of routing in IP fabrics and RIFT protocol concepts.

4.1. Overview of RIFT

RIFT is a dynamic routing protocol that is tailored for use in Clos, Fat-Tree, and other anisotropic topologies. A core property of RIFT is that its operation is sensitive to the structure of the fabric - it is anisotropic. RIFT acts as a link-state protocol when "pointing north" - advertising southwards routes to northwards peer routers (parents) through flooding and database synchronization- but operates hop-by-hop like a distance-vector protocol when "pointing south" - typically advertising a fabric default route directed towards the Top of Fabric (ToF, aka superspine) to southwards peer routers (children).

The fabric default is typically the default route, as described in Section 4.2.3.8 "Southbound Default Route Origination" of RIFT [RIFT]. The ToF nodes may alternatively originate more specific prefixes (P') southbound instead of the default route. In such a scenario, all addresses carried within the RIFT domain must be contained within P', and it is possible for a leaf that acts as gateway to the internet to advertise the default route instead.

RIFT floods flat link-state information northbound only so that each level obtains the full topology of levels south of it. That information is never flooded east-west or back south again. So a top tier node has full set of prefixes from the Shortest Path First (SPF) calculation.

In the southbound direction, the protocol operates like a "fully summarizing, unidirectional" path-vector protocol or rather a distance-vector with implicit split horizon. Routing information, normally just the default route, propagates one hop south and is "re-advertised" by nodes at next lower level.

```
          +---------------+       +----------------+
          |      ToF      |       |       ToF      |     LEVEL 2
  +       ++------+--+--+-+       ++-+--+----+-----+
  |        |      |  |  |          | |  |    |        ^
  +        |      |  |  +-------------------------+   |
  Distance |   +-------------------+ |  |    |    |   |
  Vector   |   |  |  |               |  |    |    |   +
  South    |   |  |  |      +--------+  |    |    |   Link-State
  +        |   |  |  |      |           |    |    |   Flooding
  |        |   |  +----------------+    |    |    |   North
  v        |   |     |      |      |    |    |    |   +
          ++---+-+   +------+    +-+----+   ++----++  |
          |SPINE |   |SPINE |    | SPINE|   | SPINE|  |  LEVEL 1
  +       ++----++   ++---+-+    +-+--+-+   ++----++  |
  +        |    |     |   |        |  |       |    |   |     ^ N
  Distance |    +-------+ |        |  +--------+  |   |     |   E
  Vector   |          | | |        |         | |  |   |  +------>
  South    |  +-------+ | |        |  +------+ |  |   |     |
  +        |  |         | |        |  |        |  |   |     +
  v       ++--++      +-+-++      ++--++      ++--++  +
          |LEAF|      |LEAF|      |LEAF|      |LEAF|     LEVEL 0
          +----+      +----+      +----+      +----+
```

Figure 1: RIFT overview

A spine node has only information necessary for its level, which is all destinations south of the node based on SPF calculation, default route, and potential disaggregated routes.

RIFT combines the advantage of both link-state and distance-vector:

*Fastest possible convergence

*Automatic detection of topology

*Minimal routes/info on Top-of-Rack (ToR) switches, aka leaf nodes

*High degree of ECMP

*Fast de-commissioning of nodes

*Maximum propagation speed with flexible prefixes in an update

So there are two types of link-state database which are "north representation" North Topology Information Elements (N-TIEs) and "south representation" South Topology Information Elements (S-TIEs). The N-TIEs contain a link-state topology description of lower levels and S-TIEs carry simply default and disaggregated routes for the lower levels.

RIFT also eliminates major disadvantages of link-state and distance-vector with:

*Reduced and balanced flooding

*Level constrained automatic neighbor detection

To achieve this, RIFT builds on the art of IGPs, not only OSPF and IS-IS but also MANET and IoT, to provide unique features:

*Automatic (positive or negative) route disaggregation of northwards routes upon fallen leaves

*Recursive operation in the case of negative route disaggregation

*Anisotropic routing that extends a principle seen in RPL [RFC6550] to wide superspines

*Optimal flooding reduction that derives from the concept of a "multipoint relay" (MPR) found in OLSR [RFC3626] and balances the flooding load over northbound links and nodes.

Additional advantages that are unique to RIFT are listed below, the details of which can be found in RIFT [RIFT].

*True ZTP(Zero Touch Provisioning)

*Minimal blast radius on failures

*Can utilize all paths through fabric without looping

*Simple leaf implementation that can scale down to servers

*Key-Value store

*Horizontal links used for protection only

4.2. Applicable Topologies

Albeit RIFT is specified primarily for "proper" Clos or Fat Tree topologies, the protocol natively supports Points of Delivery (PoD) concepts, which, strictly speaking, are not found in the original Clos concept.

Further, the specification explains and supports operations of multi-plane Clos variants where the protocol recommends the use of inter-plane rings at the Top-of-Fabric level to allow the reconciliation of topology view of different planes to make the negative disaggregation viable in case of failures within a plane. These observations hold not only in case of RIFT but also in the generic case of dynamic routing on Clos variants with multiple planes and failures in bi-sectional bandwidth, especially on the leafs.

4.2.1. Horizontal Links

RIFT is not limited to pure Clos divided into PoD and multi-planes but supports horizontal (East-West) links below the top of fabric level. Those links are used only for last resort northbound forwarding when a spine loses all its northbound links or cannot compute a default route through them.

A full-mesh connectivity between nodes on the same level can be employed and that allows N-SPF to provide for any node losing all its northbound adjacencies (as long as any of the other nodes in the level are northbound connected) to still participate in northbound forwarding.

Note that a "ring" of horizontal links at any level below ToF does not provide a "ring-based protection" scheme since the SPF computation would have to deal necessarily with breaking of "loops" in Dijkstra sense--an application for which RIFT is not intended.

4.2.2. Vertical Shortcuts

Through relaxations of the specified adjacency forming rules, RIFT implementations can be extended to support vertical "shortcuts". The RIFT specification itself does not provide the exact details since the resulting solution suffers from either much larger blast radius with increased flooding volumes or in case of maximum aggregation routing, bow-tie problems.

4.2.3. Generalizing to any Directed Acyclic Graph

RIFT is an anisotropic routing protocol, meaning that it has a sense of direction (northbound, southbound, east-west) and that it operates differently depending on the direction.

- Northbound, RIFT operates as a link-state protocol, whereby the control packets are reflooded first all the way north and only interpreted later. All the individual fine grained routes are advertised.

- Southbound, RIFT operates as a distance-vector protocol, whereby the control packets are flooded only one-hop, interpreted, and the consequence of that computation is what gets flooded one more hop south. In the most common use-cases, a ToF node can reach most of the prefixes in the fabric. If that is the case, the ToF node advertises the fabric default and negatively disaggregates the prefixes that it cannot reach. On the other hand, a ToF node that can reach only a small subset of the prefixes in the fabric will preferably advertise those prefixes and refrain from aggregating.

  In the general case, what gets advertised south are:

  1. A fabric default that aggregates all the prefixes that are reachable within the fabric, and that could be a default route or a prefix that is dedicated to this particular fabric.

  2. The loopback addresses of the northbound nodes, e.g., for inband management.

  3. The disaggregated prefixes for the dynamic exceptions to the fabric default, advertised to route around the black hole that may form.

- East-West routing can optionally be used, with specific restrictions. It is used when a sibling has access to the fabric default but this node does not.

Since a Directed Acyclic Graph (DAG) provides a sense of north (the direction of the DAG) and of south (the reverse), it can be used to apply RIFT--an edge in the DAG that has only incoming vertices is a ToF node.

There are a number of caveats though:

- The DAG structure must exist before RIFT starts, so there is a need for a companion protocol to establish the logical DAG structure.

- A generic DAG does not have a sense of east and west. The operation specified for east-west links and the southbound reflection between nodes are not applicable. Also ZTP(Zero Touch Provisioning) will derive a sense of depth that will eliminate some links. Variations of ZTP(Zero Touch Provisioning) could be derived to meet specific objectives, e.g., make it so that most routers have at least 2 parents to reach the ToF.

- RIFT applies to any Destination-Oriented DAG (DODAG) where there's only one ToF node and the problem of disaggregation does not exist. In that case, RIFT operates very much like RPL [RFC6550], but using Link State for southbound routes (downwards in RPL's terms). For an arbitrary DAG with multiple destinations (ToFs) the way disaggregation happens has to be considered.

- Positive disaggregation expects that most of the ToF nodes reach most of the leaves, so disaggregation is the exception as opposed to the rule. When this is no more true, it makes sense to turn off disaggregation and route between the ToF nodes over a ring, a full mesh, transit network, or a form of area zero. There again, this operation is similar to RPL operating as a single DODAG with a virtual root.

- In order to aggregate and disaggregate routes, RIFT requires that all the ToF nodes share the full knowledge of the prefixes in the fabric.

- This can be achieved with a ring as suggested by "RIFT" [RIFT], by some preconfiguration, or using a synchronization with a common repository where all the active prefixes are registered.

4.2.4. Reachability of Internal Nodes in the Fabric

RIFT does not require that nodes have reachable addresses in the fabric, though it is clearly desirable for operational purposes. Under normal operating conditions this can be easily achieved by injecting the node's loopback address into North and South Prefix TIEs or other implementation specific mechanisms.

Special considerations arise when a node loses all northbound adjacencies, but is not at the top of the fabric. If a spine node loses all northbound links, the spine node doesn't advertise default route. But if the level of the spine node is auto-determined by ZTP, it will "fall down" as despicted in Figure 8.

4.3. Use Cases

4.3.1. Data Center Topologies

4.3.1.1. Data Center Fabrics

RIFT is suited for applying in data center (DC) IP fabrics underlay routing, vast majority of which seem to be currently (and for the foreseeable future) Clos architectures. It significantly simplifies operation and deployment of such fabrics as described in Section 5 for environments compared to extensive proprietary provisioning and operational solutions.

4.3.1.2. Adaptations to Other Proposed Data Center Topologies

```
.  +-----+        +-----+
.  |     |        |     |
.+-+ S0  |        | S1  |
.| ++---++        ++---++
.|  |   |          |   |
.|  | +------------+   |
.|  | | +------------+ |
.|  | |            | |
.| ++-+--+        +--+-++
.| |     |        |     |
.| | A0  |        | A1  |
.| +-+--++        ++---++
.|   |  |          |   |
.|   |  +------------+ |
.|   | +-----------+ | |
.|   | |           | |
.| +-+-+-+        +--+-++
.+-+     |        |     |
.  | L0  |        | L1  |
.  +-----+        +-----+
```

Figure 2: Level Shortcut

RIFT is not strictly limited to Clos topologies. The protocol only requires a sense of "compass rose directionality" either achieved through configuration or derivation of levels. So, conceptually, shortcuts between levels could be included. Figure 2 depicts an example of a shortcut between levels. In this example, sub-optimal routing will occur when traffic is sent from L0 to L1 via S0's default route and back down through A0 or A1. In order to avoid that, only default routes from A0 or A1 are used, all leaves would be required to install each others routes.

While various technical and operational challenges may require the use of such modifications, discussion of those topics are outside the scope of this document.

4.3.2. Metro Fabrics

The demand for bandwidth is increasing steadily, driven primarily by environments close to content producers (server farms connection via DC fabrics) but in proximity to content consumers as well. Consumers are often clustered in metro areas with their own network architectures that can benefit from simplified, regular Clos structures and hence from RIFT.

4.3.3. Building Cabling

Commercial edifices are often cabled in topologies that are either Clos or its isomorphic equivalents. The Clos can grow rather high with many levels. That presents a challenge for traditional routing protocols (except BGP and by now largely phased-out PNNI) which do not support an arbitrary number of levels which RIFT does naturally. Moreover, due to the limited sizes of forwarding tables in network elements of building cabling, the minimum FIB size RIFT maintains under normal conditions is cost-effective in terms of hardware and operational costs.

4.3.4. Internal Router Switching Fabrics

It is common in high-speed communications switching and routing devices to use fabrics when a crossbar is not feasible due to cost, head-of-line blocking or size trade-offs. Normally such fabrics are not self-healing or rely on 1:/+1 protection schemes but it is conceivable to use RIFT to operate Clos fabrics that can deal effectively with interconnections or subsystem failures in such module. RIFT is not IP specific and hence any link addressing connecting internal device subnets is conceivable.

4.3.5. CloudCO

The Cloud Central Office (CloudCO) is a new stage of telecom Central Office. It takes the advantage of Software Defined Networking (SDN) and Network Function Virtualization (NFV) in conjunction with general purpose hardware to optimize current networks. The following figure illustrates this architecture at a high level. It describes a single instance or macro-node of cloud CO that provides a number of Value Added Services (VAS), a Broadband Access Abstraction (BAA), and virtualized nerwork services. An Access I/O module faces a Cloud CO access node, and the Customer Premises Equipments (CPEs) behind it. A Network I/O module is facing the core network. The two I/O modules are interconnected by a leaf and spine fabric [TR-384].

```
    +---------------------+           +----------------------+
    |         Spine       |           |     Spine            |
    |         Switch      |           |     Switch           |
    +------+---+------+-+-+           +--+-+-+-+-----+-------+
    |      |   |      | | |              | | | |     |       |
    |      |   |      | | +-------------------------------+  |
    |      |   |      | |                | | | |     |    |  |
    |      |   |      | +-------------------------+  |    |  |
    |      |   |      |                  | | | |  |  |    |  |
    |      |   +----------------------+  | | | |  |  |    |  |
    |      |          |               |  | | | |  |  |    |  |
    |  +---------------------------------+ | | |  |  |    |  |
    |  |   |          |               |    | | |  |  |    |  |
    |  |   |   +-----------------------------+ |  |  |    |  |
    |  |   |   |      |               |    |   |  |  |    |  |
    |  |   |   |      |   +--------------------+  |  |    |  |
    |  |   |   |      |   |           |    |      |  |    |  |
    +--+ +-+---+--+ +-+---+--+     +--+----+--+ +-+--+--+ +--+
    |L | | Leaf   | | Leaf   |     |  Leaf    | | Leaf  | |L |
    |S | | Switch | | Switch |     |  Switch  | | Switch| |S |
    ++-+ +-+-+-+--+ +-+-+-+--+     +--+-+--+--+ ++-+--+-+ +-++
     |     | | |      | | |           | | |    | |  |     |
     |   +-+-+-+--+ +-+-+-+--+     +--+-+--+--+ ++-+--+-+   |
     |   |Compute | |Compute |     | Compute  | |Compute|   |
     |   |Node    | |Node    |     | Node     | |Node   |   |
     |   +--------+ +--------+     +----------+ +-------+   |
     |   || VAS5 || || vDHCP||     || vRouter|| ||VAS1 ||   |
     |   |--------| |--------|     |----------| |-------|   |
     |   |--------| |--------|     |----------| |-------|   |
     |   || VAS6 || || VAS3 ||     || v802.1x|| ||VAS2 ||   |
     |   |--------| |--------|     |----------| |-------|   |
     |   |--------| |--------|     |----------| |-------|   |
     |   || VAS7 || || VAS4 ||     ||  vIGMP || ||BAA  ||   |
     |   |--------| |--------|     |----------| |-------|   |
     |   +--------+ +--------+     +----------+ +-------+   |
     |                                                      |
    ++-----------+                                +---------++
    |Network I/O |                                |Access I/O|
    +------------+                                +----------+
```

Figure 3: An example of CloudCO architecture

The Spine-Leaf architecture deployed inside CloudCO meets the network requirements of adaptable, agile, scalable and dynamic.

5. Operational Considerations

RIFT presents the opportunity for organizations building and operating IP fabrics to simplify their operation and deployments while achieving many desirable properties of a dynamic routing on such a substrate:

- RIFT only floods routing information to the devices that absolutely need it. RIFT design follows minimum blast radius and minimum necessary epistemological scope philosophy which leads to good scaling properties while delivering maximum reactiveness.

- RIFT allows for extensive Zero Touch Provisioning within the protocol. In its most extreme version RIFT does not rely on any specific addressing and for IP fabric can operate using IPv6 ND [RFC4861] only.

- RIFT has provisions to detect common IP fabric mis-cabling scenarios.

- RIFT negotiates automatically BFD per link. This allows for IP and micro-BFD [RFC7130] to replace Link Aggregation Groups (LAGs) which do hide bandwidth imbalances in case of constituent failures. Further automatic link validation techniques similar to [RFC5357] could be supported as well.

- RIFT inherently solves many difficult problems associated with the use of traditional routing topologies with dense meshes and high degrees of ECMP by including automatic bandwidth balancing, flood reduction and automatic disaggregation on failures while providing maximum aggregation of prefixes in default scenarios.

- RIFT reduces FIB size towards the bottom of the IP fabric where most nodes reside and allows with that for cheaper hardware on the edges and introduction of modern IP fabric architectures that encompass e.g. server multi-homing.

- RIFT provides valley-free routing and with that is loop free. This allows the use of any such valley-free path in bi-sectional fabric bandwidth between two destination irrespective of their metrics which can be used to balance load on the fabric in different ways.

- RIFT includes a key-value distribution mechanism which allows for many future applications such as automatic provisioning of basic overlay services or automatic key roll-overs over whole fabrics.

- RIFT is designed for minimum delay in case of prefix mobility on the fabric. In conjunction with [RFC8505], RIFT can differentiate anycast advertisements from mobility events and retain only the most recent advertisement in the latter case.

- Many further operational and design points collected over many years of routing protocol deployments have been incorporated in RIFT such as fast flooding rates, protection of information lifetimes and operationally easily recognizable remote ends of links and node names.

5.1. South Reflection

South reflection is a mechanism that South Node TIEs are "reflected" back up north to allow nodes in same level without east-west links to "see" each other.

For example, Spine111\Spine112\Spine121\Spine122 reflects Node S-TIEs from ToF21 to ToF22 separately. Respectively, Spine111\Spine112\Spine121\Spine122 reflects Node S-TIEs from ToF22 to ToF21 separately. So ToF22 and ToF21 see each other's node information as level 2 nodes.

In an equivalent fashion, as the result of the south reflection between Spine121-Leaf121-Spine122 and Spine121-Leaf122-Spine122, Spine121 and Spine 122 knows each other at level 1.

5.2. Suboptimal Routing on Link Failures

```
              +--------+          +--------+
              | ToF21  |          |  ToF22 |                LEVEL 2
              ++--+-+-++          ++-+--+-++
               |  | | |            | |  | +
               |  | | |            | |  | linkTS8
  +------------+  | +-+linkTS3+-+  | |  | +-------------+
  |               |   |         |  | |  +               |
  |    +---------------------------+ |  linkTS7         |
  |    |          |   |         +    +  +               |
  |    |          |   +-------+linkTS4+------------+    |
  |    |          |             +    +  |          |    |
  |    |          |    +-------------+--+          |    |
  |    |          |    |        |  linkTS6         |    |
+-+----+-+      +-+----+-+     ++--------+       +-+----+-+
|Spine111|      |Spine112|     |Spine121 |       |Spine122| LEVEL 1
+-+---+--+      +-+----+-+     +-+---+---+       +-+----+-+
  |   |           |    |         |   |             |    |
  |   +-------------+  |         +   ++XX+linkSL6+---+  +
  |               | |  |      linkSL5              | |  linkSL8
  |   +-----------+ |  |         +   +---+linkSL7+-+ |  +
  |   |             |  |         |   |               |  |
+-+---+-+        +--+--+-+     +-+---+-+          +--+--+-+
|Leaf111|        |Leaf112|     |Leaf121|          |Leaf122| LEVEL 0
+-+-----+        +-+-----+     +-----+-+          +-+-----+
  +                +                 +              +
Prefix111        Prefix112     Prefix121          Prefix122
```

Figure 4: Suboptimal routing upon link failure use case

As shown in Figure 4, as the result of the south reflection between Spine121-Leaf121-Spine122 and Spine121-Leaf122-Spine122, Spine121 and Spine 122 knows each other at level 1.

Without disaggregation mechanism, when linkSL6 fails, the packet from leaf121 to prefix122 will probably go up through linkSL5 to linkTS3 then go down through linkTS4 to linkSL8 to Leaf122 or go up through linkSL5 to linkTS6 then go down through linkTS8 and linkSL8 to Leaf122 based on pure default route. It's the case of suboptimal routing or bow-tieing.

With disaggregation mechanism, when linkSL6 fails, Spine122 will detect the failure according to the reflected node S-TIE from Spine121. Based on the disaggregation algorithm provided by RIFT, Spine122 will explicitly advertise prefix122 in Disaggregated Prefix S-TIE PrefixesElement(prefix122, cost 1). The packet from leaf121 to prefix122 will only be sent to linkSL7 following a longest-prefix match to prefix 122 directly then go down through linkSL8 to Leaf122 .

5.3. Black-Holing on Link Failures

```
                +--------+          +--------+
                | ToF 21 |          | ToF 22 |                LEVEL 2
                ++-+--+-++          ++-+--+-++
                 | |  | |            | |  | +
                 | |  | |            | |  | linkTS8
  +--------------+ |  +-+linkTS3+X+  | |  | +--------------+
  linkTS1          |    |         |  | |  +                |
  +    +-----------------------------+ |  linkTS7          |
  |    |           +    |         +    +  +                |
  |    |      linkTS2   +-------+linkTS4+X+----------+     |
  |    +           +              +    +  |          |     |
  |   linkTS5      +-+    +------------+--+          |     |
  |    +             |    |       |  linkTS6         |     |
+-+----+-+         +-+----+-+    ++-------+        +-+-----++
|Spine111|         |Spine112|    |Spine121|        |Spine122| LEVEL 1
+-+---+--+         ++----+--+    +-+---+--+        +-+----+-+
  |   |             |    |         |   |             |    |
  +   +---------------+  |         +   +---+linkSL6+---+  +
  linkSL1           | |  |      linkSL5              | |  linkSL8
  +   +--+linkSL3+--+ |  |         +   +---+linkSL7+-+ |  +
  |   |               |  |         |   |               |  |
+-+---+-+          +--+--+-+     +-+---+-+          +--+--+-+
|Leaf111|          |Leaf112|     |Leaf121|          |Leaf122| LEVEL 0
+-+-----+          +-+-----+     +-----+-+          +-----+-+
  +                  +                 +                  +
Prefix111          Prefix112     Prefix121          Prefix122
```

Figure 5: Black-holing upon link failure use case

This scenario illustrates a case when double link failure occurs and with that black-holing can happen.

Without disaggregation mechanism, when linkTS3 and linkTS4 both fail, the packet from leaf111 to prefix122 would suffer 50% black-holing based on pure default route. The packet supposed to go up through linkSL1 to linkTS1 then go down through linkTS3 or linkTS4 will be dropped. The packet supposed to go up through linkSL3 to linkTS2 then go down through linkTS3 or linkTS4 will be dropped as well. It's the case of black-holing.

With disaggregation mechanism, when linkTS3 and linkTS4 both fail, ToF22 will detect the failure according to the reflected node S-TIE of ToF21 from Spine111\Spine112. Based on the disaggregation algorithm provided by RIFT, ToF22 will explicitly originate an S-TIE with prefix 121 and prefix 122, that is flooded to spines 111, 112, 121 and 122.

The packet from leaf111 to prefix122 will not be routed to linkTS1 or linkTS2. The packet from leaf111 to prefix122 will only be routed to linkTS5 or linkTS7 following a longest-prefix match to prefix122.

5.4. Zero Touch Provisioning (ZTP)

RIFT is designed to require a very minimal configuration to simplify its operation and avoid human errors; based on that minimal information, Zero Touch Provisioning (ZTP) autoconfigures the key operational parameters of all the RIFT nodes, including the SystemID of the node that must be unique in the RIFT network and the level of the node in the Fat Tree, which determines which peers are northwards "parents" and which are southwards "children".

ZTP is always on, but its decisions can be overridden when a network administrator prefers to impose its own configuration. In that case, it is the responsibility of the administrator to ensure that the configured parameters are correct, in other words that the SystemID of each node is unique, and that the administratively set levels truly reflect the relative position of the nodes in the fabric. It is recommended to let ZTP configure the network, and when not, it is recommended to configure the level of all the nodes to avoid an undesirable interaction between ZTP and the manual configuration.

ZTP requires that the administrator points out the Top-of-Fabric (ToF) nodes to set the baseline from which the fabric topology is derived. The Top-of-Fabric nodes are configured with TOP_OF_FABRIC flag which are initial 'seeds' needed for other ZTP nodes to derive their level in the topology. ZTP computes the level of each node based on the Highest Available Level (HAL) of the potential parent(s) nearest that baseline, which represents the superspine. In a fashion, RIFT can be seen as a distance-vector protocol that computes a set of feasible successors towards the superspine and auto-configures the rest of the topology.

The autoconfiguration mechanism computes a global maximum of levels by diffusion. The derivation of the level of each node happens then based on Link Information Elements (LIEs) received from its neighbors whereas each node (with possibly exceptions of configured leaves) tries to attach at the highest possible point in the fabric. This guarantees that even if the diffusion front reaches a node from "below" faster than from "above", it will greedily abandon already negotiated level derived from nodes topologically below it and properly peer with nodes above.

The achieved equilibrium can be disturbed massively by all nodes with highest level either leaving or entering the domain (with some finer distinctions not explained further). It is therefore recommended that each node is multi-homed towards nodes with respective HAL offerings. Fortunately, this is the natural state of things for the topology variants considered in RIFT.

A RIFT node may also be configured to confine it to the leaf role with the LEAF_ONLY flag. A leaf node can also be configured to support leaf-2-leaf procedures with the LEAF_2_LEAF flag. In either case the node cannot be TOP_OF_FABRIC and its level cannot be configured. RIFT will fully determine the node's level after it is attached to the topology and ensure that the node is at the "bottom of the hierarchy" (southernmost).

5.5. Mis-cabling Examples

```
      +----------------+              +-----------------+
      |     ToF21      |       +------+      ToF22      |   LEVEL 2
      +-------+----+---+       |      +----+---+--------+
      |       |    |   |       |      |    |   |        |
      |       |    |   +----------------------------+   |
      |   +---------------------------+    |   |    |   |
      |   |   |    |           |           |   |    |   |
      |   |   |    |   +-----------------------+    |   |
      |   |   +------------------------+   |        |   |
      |   |        |   |       |       |   |        |   |
    +-+---+--+   +-+---+--+    |    +--+---+-+  +--+---+-+
    |Spine111|   |Spine112|    |    |Spine121|  |Spine122| LEVEL 1
    +-+---+--+   ++----+--+    |    +--+---+-+  +-+----+-+
      |   |       |    |       |       |   |       |    |
      |   +---------+  |     link-M    |   +---------+  |
      |           | |  |       |       |           | |  |
      |   +-------+ |  |       |       |   +-------+ |  |
      |   |         |  |       |       |   |         |  |
    +-+---+-+    +--+--+-+     |     +-+---+-+    +--+--+-+
    |Leaf111|    |Leaf112+-----+     |Leaf121|    |Leaf122| LEVEL 0
    +-------+    +-------+           +-------+    +-------+
```

Figure 6: A single plane mis-cabling example

Figure 6 shows a single plane mis-cabling example. It's a perfect Fat Tree fabric except link-M connecting Leaf112 to ToF22.

The RIFT control protocol can discover the physical links automatically and be able to detect cabling that violates Fat Tree topology constraints. It reacts accordingly to such mis-cabling attempts, at a minimum preventing adjacencies between nodes from being formed and traffic from being forwarded on those mis-cabled links. Leaf112 will in such scenario use link-M to derive its level (unless it is leaf) and can report links to Spine111 and Spine112 as mis-cabled unless the implementations allows horizontal links.

Figure 7 shows a multiple plane mis-cabling example. Since Leaf112 and Spine121 belong to two different PoDs, the adjacency between Leaf112 and Spine121 can not be formed. link-W would be detected and prevented.

```
  +-------+    +-------+           +-------+    +-------+
  |ToF  A1|    |ToF  A2|           |ToF  B1|    |ToF  B2| LEVEL 2
  +-------+    +-------+           +-------+    +-------+
  |       |    |       |           |       |    |       |
  |       |    |       +-----------------+ |    |       |
  |       +--------------------------+   | |    |       |
  |     +------+                   | |   | +------+     |
  |     |        +-----------------+ |   |      | |     |
  |     |        |   +--------------------------+ |     |
  |  A  |        | B |               | A |        |  B  |
  +-----+--+   +-+---+--+         +--+---+-+   +--+-----+
  |Spine111|   |Spine112|     +---+Spine121|   |Spine122| LEVEL 1
  +-+---+--+   ++----+--+     |   +--+---+-+   +-+----+-+
    |   |       |    |        |      |   |       |    |
    |   +---------+  |        |      |   +---------+  |
    |           | |  |      link-W   |           | |  |
    |   +-------+ |  |        |      |   +-------+ |  |
    |   |         |  |        |      |   |         |  |
  +-+---+-+    +--+--+-+      |    +-+---+-+    +--+--+-+
  |Leaf111|    |Leaf112+------+    |Leaf121|    |Leaf122| LEVEL 0
  +-------+    +-------+           +-------+    +-------+
 +--------PoD#1----------+       +---------PoD#2---------+
```

Figure 7: A multiple plane mis-cabling example

RIFT provides an optional level determination procedure in its Zero Touch Provisioning mode. Nodes in the fabric without their level configured determine it automatically. This can have possibly counter-intuitive consequences however. One extreme failure scenario is depicted in Figure 8 and it shows that if all northbound links of spine11 fail at the same time, spine11 negotiates a lower level than Leaf11 and Leaf12.

To prevent such scenario where leafs are expected to act as switches, LEAF_ONLY flag can be set for Leaf111 and Leaf112. Since level -1 is invalid, Spine11 would not derive a valid level from the topology in Figure 8. It will be isolated from the whole fabric and it would be up to the leafs to declare the links towards such spine as mis-cabled.

```
+-------+    +-------+        +-------+    +-------+
|ToF  A1|    |ToF  A2|        |ToF  A1|    |ToF  A2|
+-------+    +-------+        +-------+    +-------+
|       |    |       |                |            |
|    +-------+       |                |            |
+    +  |            |  ====>         |            |
X    X  +------+     |                +------+     |
+    +         |     |                       |     |
+----+--+    +-+-----+                     +-+-----+
|Spine11|    |Spine12|                     |Spine12|
+-+---+-+    ++----+-+                     ++----+-+
  |   |       |    |                        |    |
  |   +---------+  |                        |    |
  |   +-------+ |  |                +-------+    |
  |   |         |  |                |            |
+-+---+-+    +--+--+-+        +-----+-+    +-----+-+
|Leaf111|    |Leaf112|        |Leaf111|    |Leaf112|
+-------+    +-------+        +-+-----+    +-+-----+
                                |            |
                                |   +---------+
                                |   |
                              +-+---+-+
                              |Spine11|
                              +-------+
```

Figure 8: Fallen spine

5.6. Positive vs. Negative Disaggregation

Disaggregation is the procedure whereby [RIFT] advertises a more specific route southwards as an exception to the aggregated fabric-default north. Disaggregation is useful when a prefix within the aggregation is reachable via some of the parents but not the others at the same level of the fabric. It is mandatory when the level is the ToF since a ToF node that cannot reach a prefix becomes a black hole for that prefix. The hard problem is to know which prefixes are reachable by whom.

In the general case, [RIFT] solves that problem by interconnecting the ToF nodes. So the ToF nodes can exchange the full list of prefixes that exist in the fabric and figure out when a ToF node lacks reachability to some prefixes. This requires additional ports at the ToF, typically 2 ports per ToF node to form a ToF-spanning ring. [RIFT] also defines the southbound reflection procedure that enables a parent to explore the direct connectivity of its peers, meaning their own parents and children; based on the advertisements received from the shared parents and children, it may enable the parent to infer the prefixes its peers can reach.

When a parent lacks reachability to a prefix, it may disaggregate the prefix negatively, i.e., advertise that this parent can be used to reach any prefix in the aggregation except that one. The Negative Disaggregation signaling is simple and functions transitively from ToF to top-of-pod (ToP) and then from ToP to Leaf. But it is hard for a parent to figure which prefix it needs to disaggregate, because it does not know what it does not know; it results that the use of a spanning ring at the ToF is required to operate the Negative Disaggregation. Also, though it is only an implementation problem, the programmation of the FIB is complex compared to normal routes, and may incur recursions.

The more classical alternative is, for the parents that can reach a prefix that peers at the same level cannot, to advertise a more specific route to that prefix. This leverages the normal longest prefix match in the FIB, and does not require a special implementation. But as opposed to the Negative Disaggregation, the Positive Disaggregation is difficult and inefficient to operate transitively.

Transitivity is not needed to a grandchild if all its parents received the Positive Disaggregation, meaning that they shall all avoid the black hole; when that is the case, they collectively build a ceiling that protects the grandchild. But until then, a parent that received a Positive Disaggregation may believe that some peers are lacking the reachability and readvertise too early, or defer and maintain a black hole situation longer than necessary.

In a non-partitioned fabric, all the ToF nodes see one another through the reflection and can figure if one is missing a child. In that case it is possible to compute the prefixes that the peer cannot reach and disaggregate positively without a ToF-spanning ring. The ToF nodes can also ascertain that the ToP nodes are connected each to at least a ToF node that can still reach the prefix, meaning that the transitive operation is not required.

The bottom line is that in a fabric that is partitioned (e.g., using multiple planes) and/or where the ToP nodes are not guaranteed to always form a ceiling for their children, it is mandatory to use the Negative Disaggregation. On the other hand, in a highly symmetrical and fully connected fabric, (e.g., a canonical Clos Network), the Positive Disaggregation methods allows to save the complexity and cost associated to the ToF-spanning ring.

Note that in the case of Positive Disaggregation, the first ToF node(s) that announces a more-specific route attracts all the traffic for that route and may suffer from a transient incast. A ToP node that defers injecting the longer prefix in the FIB, in order to receive more advertisements and spread the packets better, also keeps on sending a portion of the traffic to the black hole in the meantime. In the case of Negative Disaggregation, the last ToF node(s) that injects the route may also incur an incast issue; this problem would occur if a prefix that becomes totally unreachable is disaggregated.

5.7. Mobile Edge and Anycast

When a physical or a virtual node changes its point of attachement in the fabric from a previous-leaf to a next-leaf, new routes must be installed that supersede the old ones. Since the flooding flows northwards, the nodes (if any) between the previous-leaf and the common parent are not immediately aware that the path via previous-leaf is obsolete, and a stale route may exist for a while. The common parent needs to select the freshest route advertisement in order to install the correct route via the next-leaf. This requires that the fabric determines the sequence of the movements of the mobile node.

On the one hand, a classical sequence counter provides a total order for a while but it will eventually wrap. On the other hand, a timestamp provides a permanent order but it may miss a movement that happens too quickly vs. the granularity of the timing information. It is not envisioned that an average fabric supports Precision Time Protocol [IEEEstd1588] in the short term, nor that the precision available with the Network Time Protocol [RFC5905] (in the order of 100 to 200ms) may not be necessarily enough to cover, e.g., the fast mobility of a Virtual Machine.

Section 4.3.3. "Mobility" of [RIFT] specifies an hybrid method that combines a sequence counter from the mobile node and a timestamp from the network taken at the leaf when the route is injected. If the timestamps of the concurrent advertisements are comparable (i.e., more distant than the precision of the timing protocol), then the timestamp alone is used to determine the relative freshness of the routes. Otherwise, the sequence counter from the mobile node, if available, is used. One caveat is that the sequence counter must not wrap within the precision of the timing protocol. Another is that the mobile node may not even provide a sequence counter, in which case the mobility itself must be slower than the precision of the timing.

Mobility must not be confused with anycast. In both cases, a same address is injected in RIFT at different leaves. In the case of mobility, only the freshest route must be conserved, since mobile node changed its point of attachment for a leaf to the next. In the case of anycast, the node may be either multihomed (attached to multiple leaves in parallel) or reachable beyond the fabric via multiple routes that are redistributed to different leaves; either way, in the case of anycast, the multiple routes are equally valid and should be conserved. Without further information from the redistributed routing protocol, it is impossible to sort out a movement from a redistribution that happens asynchronously on different leaves. [RIFT] expects that anycast addresses are advertised within the timing precision, which is typically the case with a low-precision timing and a multihomed node. Beyond that time interval, RIFT interprets the lag as a mobility and only the freshest route is retained.

When using IPv6 [RFC8200], RIFT suggests to leverage "Registration Extensions for IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Neighbor Discovery (ND)" [RFC8505] as the IPv6 ND interaction between the mobile node and the leaf. This provides not only a sequence counter but also a lifetime and a security token that may be used to protect the ownership of an address [RFC8928]. When using [RFC8505], the parallel registration of an anycast address to multiple leaves is done with the same sequence counter, whereas the sequence counter is incremented when the point of attachement changes. This way, it is possible to differentiate a mobile node from a multihomed node, even when the mobility happens within the timing precision. It is also possible for a mobile node to be multihomed as well, e.g., to change only one of its points of attachement.

5.8. IPv4 over IPv6

RIFT allows advertising IPv4 prefixes over IPv6 RIFT network. IPv6 Address Family (AF) configures via the usual Neighbor Discovery (ND) mechanisms and then V4 can use V6 nexthops analogous to [RFC8950]. It is expected that the whole fabric supports the same type of forwarding of address families on all the links. RIFT provides an indication whether a node is v4 forwarding capable and implementations are possible where different routing tables are computed per address family as long as the computation remains loop-free.

```
           +-----+        +-----+
+---+---+  | ToF |        | ToF |
    ^      +--+--+        +-----+
    |      |  |           |     |
    |      |  +-------------+   |
    |      |     +--------+ |   |
    +      |     |          |   |
   V6      +-----+        +-+---+
Forwarding |Spine|        |Spine|
    +      +--+--+        +-----+
    |      |  |           |     |
    |      |  +-------------+   |
    |      |     +--------+ |   |
    |      |     |          |   |
    v      +-----+        +-+---+
+---+---+  |Leaf |        | Leaf|
           +--+--+        +--+--+
              |              |
 IPv4 prefixes|              |IPv4 prefixes
              |              |
          +---+----+     +---+----+
          |   V4   |     |   V4   |
          | subnet |     | subnet |
          +--------+     +--------+
```

Figure 9: IPv4 over IPv6

5.9. In-Band Reachability of Nodes

RIFT doesn't precondition that nodes of the fabric have reachable addresses. But the operational reasons to reach the internal nodes may exist. Figure 10 shows an example that the network management station (NMS) attaches to leaf1.

```
+-------+      +-------+
| ToF1  |      | ToF2  |
++---- ++      ++-----++
 |     |        |     |
 |     +----------+   |
 |     +--------+ |   |
 |     |          |   |
++-----++      +--+---++
|Spine1 |      |Spine2 |
++-----++      ++-----++
 |     |        |     |
 |     +----------+   |
 |     +--------+ |   |
 |     |          |   |
++-----++      +--+---++
| Leaf1 |      | Leaf2 |
+---+---+      +-------+
    |
    |NMS
```

Figure 10: In-Band reachability of node

If NMS wants to access Leaf2, it simply works. Because loopback address of Leaf2 is flooded in its Prefix North TIE.

If NMS wants to access Spine2, it simply works too. Because spine node always advertises its loopback address in the Prefix North TIE. NMS may reach Spine2 from Leaf1-Spine2 or Leaf1-Spine1-ToF1/ToF2-Spine2.

If NMS wants to access ToF2, ToF2's loopback address needs to be injected into its Prefix South TIE. This TIE must be seen by all nodes at the level below - the spine nodes in Figure 10 - that must form a ceiling for all the traffic coming from below (south). Otherwise, the traffic from NMS may follow the default route to the wrong ToF Node, e.g., ToF1.

In case of failure between ToF2 and spine nodes, ToF2's loopback address must be disaggregated recursively all the way to the leaves. In a partitioned ToF, even with recursive disaggregation a ToF node is only reachable within its plane.

A possible alternative to recursive disaggregation is to use a ring that interconnects the ToF nodes to transmit packets between them for their loopback addresses only. The idea is that this is mostly control traffic and should not alter the load balancing properties of the fabric.

5.10. Dual Homing Servers

Each RIFT node may operate in Zero Touch Provisioning (ZTP) mode. It has no configuration (unless it is a Top-of-Fabric at the top of the topology or the must operate in the topology as leaf and/or support leaf-2-leaf procedures) and it will fully configure itself after being attached to the topology.

```
                +---+         +---+         +---+
                |ToF|         |ToF|         |ToF|      ToF
                +---+         +---+         +---+
                |   |         |   |         |   |
                |   +----------------+      |   |
                |          +----------------+   |
                |          |  |   |  |          |
                +----------+--+   +--+----------+
                |     ToR1    |   |     ToR2    |      Spine
                +--+------+---+   +--+-------+--+
            +---+  |      |   |   |  |       |  +---+
            |   +-----------------+  |       |      |
            |   |  |   +-------------+       |      |
            |   |  |   |  |   +-----------------+   |
            |   |  |   |  +--------------+   |  |   |
            |   |  |   |                 |   |  |   |
            +---+  +---+                 +---+  +---+
            |   |  |   |                 |   |  |   |
            +---+  +---+  .............  +---+  +---+
            SV(1) SV(2)                 SV(n-1) SV(n)  Leaf
```

Figure 11: Dual-homing servers

Sometimes, people may prefer to disaggregate from ToR to servers from start on, i.e. the servers have couple tens of routes in FIB from start on beside default routes to avoid breakages at rack level. Full disaggregation of the fabric could be achieved by configuration supported by RIFT.

5.11. Fabric With A Controller

There are many different ways to deploy the controller. One possibility is attaching a controller to the RIFT domain from ToF and another possibility is attaching a controller from the leaf.

```
                                    +------------+
                                    | Controller |
                                    ++----------++
                                     |          |
                                     |          |
                                +----++        ++----+
                    -------     | ToF |        | ToF |
                       |        +--+--+        +-----+
                       |        |  |           |     |
                       |        |  +-------------+   |
                       |        |     +--------+ |   |
                       |        |     |          |   |
                                +-----+        +-+---+
                   RIFT domain  |Spine|        |Spine|
                                +--+--+        +-----+
                       |        |  |           |     |
                       |        |  +-------------+   |
                       |        |     +--------+ |   |
                       |        |     |          |   |
                       |        +-----+        +-+---+
                    -------     |Leaf |        | Leaf|
                                +-----+        +-----+
```

Figure 12: Fabric with a controller

5.11.1. Controller Attached to ToFs

If a controller is attaching to the RIFT domain from ToF, it usually uses dual-homing connections. The loopback prefix of the controller should be advertised down by the ToF and spine to leaves. If the controller loses link to ToF, make sure the ToF withdraw the prefix of the controller.

5.11.2. Controller Attached to Leaf

If the controller is attaching from a leaf to the fabric, no special provisions are needed.

5.12. Internet Connectivity Within Underlay

If global addressing is running without overlay, an external default route needs to be advertised through RIFT fabric to achieve internet connectivity. For the purpose of forwarding of the entire RIFT fabric, an internal fabric prefix needs to be advertised in the South Prefix TIE by ToF and spine nodes.

5.12.1. Internet Default on the Leaf

In case that the internet gateway is a leaf, the leaf node as the internet gateway needs to advertise a default route in its Prefix North TIE.

5.12.2. Internet Default on the ToFs

In case that the internet gateway is a ToF, the ToF and spine nodes need to advertise a default route in the Prefix South TIE.

5.13. Subnet Mismatch and Address Families

```
+--------+                     +--------+
|        |  LIE         LIE    |        |
|   A    | +---->       <----+ |   B    |
|        +---------------------+        |
+--------+                     +--------+
   X/24                           Y/24
```

Figure 13: subnet mismatch

LIEs are exchanged over all links running RIFT to perform Link (Neighbor) Discovery. A node must NOT originate LIEs on an address family if it does not process received LIEs on that family. LIEs on same link are considered part of the same negotiation independent on the address family they arrive on. An implementation must be ready to accept TIEs on all addresses it used as source of LIE frames.

As shown in the above figure, without further checks adjacency of node A and B may form, but the forwarding between node A and node B may fail because subnet X mismatches with subnet Y.

To prevent this a RIFT implementation should check for subnet mismatch just like e.g. ISIS does. This can lead to scenarios where an adjacency, despite exchange of LIEs in both address families may end up having an adjacency in a single AF only. This is a consideration especially in Section 5.8 scenarios.

5.14. Anycast Considerations

```
                               + traffic
                               |
                               v
                        +------+------+
                        |     ToF     |
                        +---+-----+---+
                        |   |     |   |
           +------------+   |     |   +------------+
           |                |     |                |
       +---+---+    +--------+     +--------+    +---+---+
       |       |    |       |     |       |    |       |
       |Spine11|    |Spine12|     |Spine21|    |Spine22| LEVEL 1
       +-+---+-+    ++----+-+     +-+---+-+    ++----+-+
         |   |       |    |         |   |       |    |
         |   +---------+  |         |   +---------+  |
         |   +-------+ |  |         |   +-------+ |  |
         |   |         |  |         |   |         |  |
       +-+---+-+    +--+--+-+     +-+---+-+    +--+--+-+
       |       |    |       |     |       |    |       |
       |Leaf111|    |Leaf112|     |Leaf121|    |Leaf122| LEVEL 0
       +-+-----+    ++------+     +-----+-+    +-----+-+
         +           +                  +      ^     +
       PrefixA      PrefixB         PrefixA    | PrefixC
                                               |
                                               + traffic
```

Figure 14: Anycast

If the traffic comes from ToF to Leaf111 or Leaf121 which has anycast prefix PrefixA, RIFT can deal with this case well. But if the traffic comes from Leaf122, it arrives Spine21 or Spine22 at level 1. But Spine21 or Spine22 doesn't know another PrefixA attaching Leaf111. So it will always get to Leaf121 and never get to Leaf111. If the intension is that the traffic should been offloaded to Leaf111, then use policy guided prefixes defined in RIFT [RIFT].

5.15. IoT Applicability

The design of RIFT inherits from RPL [RFC6550] the anisotropic design of a default route upwards (northwards); it also inherits the capability to inject external host routes at the Leaf level using Wireless ND (WiND) [RFC8505][RFC8928] between a RIFT-agnostic host and a RIFT router. Both the RPL and the RIFT protocols are meant for large scale, and WiND enables device mobility at the edge the same way in both cases.

The main difference between RIFT and RPL is that with RPL, there's a single Root, whereas RIFT has many ToF nodes. This adds huge capabilities for leaf-2-leaf ECMP paths, but additional complexity with the need to disaggregate. Also RIFT uses Link State flooding northwards, and is not designed for low-power operation.

Still nothing prevents that the IP devices connected at the Leaf are IoT (Internet of Things) devices, which typically expose their address using WiND - which is an upgrade from 6LoWPAN ND [RFC6775].

A network that serves high speed/ high power IoT devices should typically provide deterministic capabilities for applications such as high speed control loops or movement detection. The Fat Tree is highly reliable, and in normal condition provides an equivalent multipath operation; but the ECMP doesn't provide hard guarantees for either delivery or latency. As long as the fabric is non-blocking the result is the same; but there can be load unbalances resulting in incast and possibly congestion loss that will prevent the delivery within bounded latency.

This could be alleviated with Packet Replication, Elimination and Reordering (PREOF) [RFC8655] leaf-2-leaf but PREOF is hard to provide at the scale of all flows, and the replication may increase the probability of the overload that it attempts to solve.

Note that the load balancing is not RIFT's problem, but it is key to serve IoT adequately.

5.16. Key Management

As outlined in Section "Security Considerations" of [RIFT], either a private shared key or a public/private key pair is used to authenticate the adjacency. Both the key distribution and key synchronization methods are out of scope for this document. Both nodes in the adjacency must share the same keys, key type, and algorithm for a given key ID. Mismatched keys will not inter-operate as their security envelopes will be unverifiable.

Key roll-over while the adjacency is active may be supported. The specific mechanism is well documented in [RFC6518].

6. Security Considerations

This document presents applicability of RIFT. As such, it does not introduce any security considerations. However, there are a number of security concerns at [RIFT].

7. IANA Considerations

This document has no IANA actions.

8. Acknowledgments

The authors would like to thank Jaroslaw Kowalczyk and Jeffrey Zhang for providing invaluable concepts and content for this document.

9. Contributors

The following people (listed in alphabetical order) contributed significantly to the content of this document and should be considered co-authors:

Tom Verhaeg

Juniper Networks

Email: tverhaeg@juniper.net

10. Normative References

[ISO10589-Second-Edition] International Organization for Standardization, "Intermediate system to Intermediate system intra-domain routeing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode Network Service (ISO 8473)", November 2002.

[TR-384] Broadband Forum Technical Report, "TR-384 Cloud Central Office Reference Architectural Framework", January 2018.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <https://www.rfc-editor.org/info/rfc4861>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC7130] Bhatia, M., Ed., Chen, M., Ed., Boutros, S., Ed., Binderberger, M., Ed., and J. Haas, Ed., "Bidirectional Forwarding Detection (BFD) on Link Aggregation Group

(LAG) Interfaces", RFC 7130, DOI 10.17487/RFC7130, February 2014, <https://www.rfc-editor.org/info/rfc7130>.

[RFC8950] Litkowski, S., Agrawal, S., Ananthamurthy, K., and K. Patel, "Advertising IPv4 Network Layer Reachability Information (NLRI) with an IPv6 Next Hop", RFC 8950, DOI 10.17487/RFC8950, November 2020, <https://www.rfc-editor.org/info/rfc8950>.

[RFC6518] Lebovitz, G. and M. Bhatia, "Keying and Authentication for Routing Protocols (KARP) Design Guidelines", RFC 6518, DOI 10.17487/RFC6518, February 2012, <https://www.rfc-editor.org/info/rfc6518>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <https://www.rfc-editor.org/info/rfc6550>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <https://www.rfc-editor.org/info/rfc6775>.

[RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <https://www.rfc-editor.org/info/rfc8655>.

[RIFT] Sharma, A., Thubert, P., Rijsman, B., and D. Afanasiev, "RIFT: Routing in Fat Trees", Work in Progress, Internet-Draft, draft-ietf-rift-rift-13, 12 July 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-rift-rift-13>.

11. Informative References

[IEEEstd1588] IEEE standard for Information Technology, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems", <https://standards.ieee.org/standard/1588-2019.html>.

[CLOS] Yuan, X., "On Nonblocking Folded-Clos Networks in Computer Communication Environments", IEEE International Parallel & Distributed Processing Symposium, 2011.

[FATTREE] Leiserson, C. E., "Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing", 1985.

[RFC3626] Clausen, T., Ed. and P. Jacquet, Ed., "Optimized Link State Routing Protocol (OLSR)", RFC 3626, DOI 10.17487/RFC3626, October 2003, <https://www.rfc-editor.org/info/rfc3626>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <https://www.rfc-editor.org/info/rfc5905>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[RFC8505] Thubert, P., Ed., Nordmark, E., Chakrabarti, S., and C. Perkins, "Registration Extensions for IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Neighbor Discovery", RFC 8505, DOI 10.17487/RFC8505, November 2018, <https://www.rfc-editor.org/info/rfc8505>.

[RFC8928] Thubert, P., Ed., Sarikaya, B., Sethi, M., and R. Struik, "Address-Protected Neighbor Discovery for Low-Power and Lossy Networks", RFC 8928, DOI 10.17487/RFC8928, November 2020, <https://www.rfc-editor.org/info/rfc8928>.

Authors' Addresses

Yuehua Wei (editor)
ZTE Corporation
No.50, Software Avenue
Nanjing
210012
China

Email: wei.yuehua@zte.com.cn

Zheng Zhang
ZTE Corporation
No.50, Software Avenue

Nanjing
210012
China

Email: zhang.zheng@zte.com.cn

Dmitry Afanasiev
Yandex

Email: fl0w@yandex-team.ru

Pascal Thubert
Cisco Systems, Inc
Building D
45 Allee des Ormes - BP1200
06254 MOUGINS - Sophia Antipolis
France

Phone: +33 497 23 26 34
Email: pthubert@cisco.com

Tony Przygienda
Juniper Networks
1194 N. Mathilda Ave
Sunnyvale, CA, 94089
United States of America

Email: prz@juniper.net