

RTGWG
Internet-Draft
Intended status: Informational
Expires: May 3, 2018

J. Tantsura
Individual
D. Afanasiev
YANDEX
K. Patel
Arccus
P. Lapukhov
Facebook
P. Przygienda
Juniper
R. White
LinkedIn
Y. Qu
Huawei
J. Uttaro
ATT
October 30, 2017

Requirements for the DataCenter Routing
draft-dt-rtgwg-dcrouting-requirements-00

Abstract

This document describes list of functional requirments towards a Routing Protocol for Data Center Networks.

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 May 3, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. Recommended Reading	3
4. Goals and Requirements	3
5. For further study	5
6. Network Topologies	5
7. IANA Considerations	5
8. Security Considerations	5
9. Acknowledgements	5
10. Change Log	5
11. References	5
11.1. Normative References	5
11.2. Informative References	5
Authors' Addresses	6

1. Introduction

It is common to host and build a network of more than tens of thousands of end points inside a data center. Data Center Networks of such size have unique set of requirements with emphasis on scale, convergence, network stability and operational simplicity. Existing or new set of protocols and routing infrastructure needs to be augmented to support higher scale, faster convergence with increased optional simplicity in order to address the requirements of these networks.

This document describes list of functional requirements that are required towards addressing scale, convergence and operational maintainance of such scaled networks. The requirements described in this document can be used to augment existing solutions or used to design a new set of solutions.

2. 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] only when they appear in all upper case. They may also appear in lower or mixed case as English words, without any normative meaning.

3. Recommended Reading

This document assumes knowledge of existing data center networks and data center network topologies [CLOS],[RFC7938]. This document also assumes knowledge of data center routing protocols like BGP [RFC4271], OSPF [RFC2328] and BFD [RFC5880] as well as data center layer 2 link layer protocols like LLDP [RFC4957].

4. Goals and Requirements

Following are general requirements for the Data Center Network Fabric and its Routing Protocols:

- o The Fabric provides basic connectivity, with possibility to carry one or more overlays.
The Fabric provides no domain separation, if needed, to be handled by an overlay.
The Fabric MAY provide interconnect facility for other fabrics.
The Fabric MUST support non equidistant end-points.
- o The Fabric MUST support Spine and Leaf [CLOS] + isomorphic topologies within its network.
The Fabric MAY support non Spine and Leaf topologies
- o The KPI's below are single-dimensional and expected to be changed, as the document progresses, based on more feedback, we ask community to communicate their views. Combination of # of routes vs # of paths vs desired convergence time will be discussed in a later version.

The Fabric SHOULD support 250k routes @ 5k fabric nodes with convergence time below 250ms.

The Fabric SHOULD support 500k routes @ 7.5k fabric nodes with convergence time below 500ms.

The Fabric SHOULD support 1M routes @ 10k fabric nodes with convergence time below 1s.

- o The Fabric routing protocol MUST support load balancing using ECMP, wECMP and UCMP.

The Fabric routing protocol MAY support any custom or adaptive load balancing algorithms.

The Fabric routing protocol MUST support and provide facility for topology-specific algorithms that enable correct operations in that specific topology.

- o The Fabric routing protocol SHOULD support route scale and convergence times of a Fabric mentioned above.
The Fabric routing protocol SHOULD support ECMP as wide as 256 paths.
- o The Fabric routing protocol MUST support various address families that covers IP as well as MPLS forwarding.
The Fabric routing protocol MUST support extensions to carry 3rd party data and Opaque data.
- o The Fabric routing protocol MUST support Traffic Engineering paths that are host and/or router based paths.
The Fabric routing protocol MUST provide facility to address constituents in an ECMP bundle.
- o The Fabric routing protocol MUST support inband as well as out of band management.
- o The Fabric routing protocol MUST support Zero Touch Provisioning (ZTP).
The Fabric routing protocol MUST support Neighbor Discovery to facilitate ZTP.
- o The Fabric routing protocol MUST be able to leverage BFD [RFC5880] for neighbor state.
The Fabric routing protocol SHOULD be capable of bootstrapping a BFD session [RFC5882].
- o The Fabric routing protocol MUST be able to support real time state notifications of routes and its neighbors state to facilitate control plane telemetry.
The Fabric routing protocol MUST be able to support on-demand snapshots of protocol state and real time state notifications of routes and its neighbors state to remote node(s) to facilitate control plane telemetry.
- o The Fabric routing protocol MUST be able handle commission/decommission of a node as well as any node restart with a minimal data plane impact.

5. For further study

Following topics have been identified to be studied at a later time:

- o gRPC/THRIFT/similar encodings.
- o Ability to function as an overlay.
- o Flowlets signaling.
- o Multicast.
- o State representation NB.
- o Integration with PCE/SDNc.

6. Network Topologies

7. IANA Considerations

8. Security Considerations

This document describes list of functional requirements towards a routing protocol for Data Center Networks. It does not raise any security concerns or issues in addition to ones common to a routing protocol for Data Center Networks.

9. Acknowledgements

10. Change Log

Initial Version: October 31 2017

11. References

11.1. Normative References

- [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/info/rfc2119>>.

11.2. Informative References

- [CLOS] "A Study of Non-Blocking Switching Networks", The Bell System Technical Journal, Vol. 32(2), DOI 10.1002/j.1538-7305.1953.tb01433.x, March 1953.

- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<https://www.rfc-editor.org/info/rfc2328>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [RFC4957] Krishnan, S., Ed., Montavont, N., Njedjou, E., Veerepalli, S., and A. Yegin, Ed., "Link-Layer Event Notifications for Detecting Network Attachments", RFC 4957, DOI 10.17487/RFC4957, August 2007, <<https://www.rfc-editor.org/info/rfc4957>>.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <<https://www.rfc-editor.org/info/rfc5880>>.
- [RFC5882] Katz, D. and D. Ward, "Generic Application of Bidirectional Forwarding Detection (BFD)", RFC 5882, DOI 10.17487/RFC5882, June 2010, <<https://www.rfc-editor.org/info/rfc5882>>.
- [RFC7938] Lapukhov, P., Premji, A., and J. Mitchell, Ed., "Use of BGP for Routing in Large-Scale Data Centers", RFC 7938, DOI 10.17487/RFC7938, August 2016, <<https://www.rfc-editor.org/info/rfc7938>>.

Authors' Addresses

Jeff Tantsura
Individual
USA

Email: jefftant.ietf@gmail.com

Dmitry Afanasiev
YANDEX
RU

Email: dmitry.afanasiev@gmail.com

Keyur Patel
Arccus
San Jose
USA

Email: keyur@arccus.com

Petr Lapukhov
Facebook
USA

Email: petr@fb.com

Tony Przygienda
Juniper
1137 Innovation Way
Sunnyvale, CA
USA

Email: prz@juniper.net

Russ White
LinkedIn
USA

Email: russ@riw.us

Yingzhen Qu
Huawei
Santa Clara
USA

Email: yingzhen.ietf@gmail.com

Jim Uttaro
ATT
USA

Email: juttaro@ieee.org

INTERNET-DRAFT

Intended Status: Informational
Expires: April 24, 2018

N. Elkins
Inside Products
M. Ackermann
BCBS Michigan
October 21, 2017

Common Network Architecture for Brick and Mortar Enterprises
draft-elkins-brickmortar-architecture-00

Abstract

The network architecture and topology for "brick and mortar" enterprises differ in significant aspects from those of Internet-based companies. This has implications for protocol implementations.

By and large, the network connects to sites spread throughout a geographic region. The architecture is not flat. There may be multiple hops - routers, middle boxes and the like. There may also be multiple carriers or ISPs involved (including internally built infrastructure). The number, nature and amount of applications also dictate a complex topology which then dictates a complex protocol implementation. Lastly, a number of these enterprises are in industries which are regulated. Such regulations impact the nature of network design. These considerations are discussed in this document.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

Copyright and License Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

IETF Trust Legal Provisions of 28-dec-2009, Section 6.b(i), paragraph 3: This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1	Background	3
1.1	Middle Box Usage	3
1.2	Routing and Other Protocols Used	4
1.3	"Home-grown" Infrastructure	4
1.4	Connections to Business Partners	4
2	Regulatory Requirements	4
2.1	End-to-End Encryption	4
3	Applications	4
4	Security Considerations	5
5	IANA Considerations	5
6	References	5
6.1	Normative References	5
6.2	Informative References	5
7	Acknowledgments	5
	Authors' Addresses	6

1 Background

The network architecture and topology for "brick and mortar" enterprises differ in significant aspects from those of Internet-based companies. This has implications for protocol implementations. By and large, the network connects to sites spread throughout a geographic region. The architecture is not flat. For example, for an oil and gas company, the network may connect refineries, gas stations, storage depots, oil fields, and the like. The architecture is not flat.

Within the data center as well as to the end location, there will be multiple hops - routers, firewall, load balancers and the like. Often multi-homing is done for fall back and disaster recovery. Hence, multiple carriers or ISPs will be involved. Thus, the architecture is inherently complex -- some have routes with 10, 15 or even over 50 hops.

The number, nature and amount of applications also dictate a complex topology which then dictates a complex protocol implementation. Lastly, a number of these enterprises are in industries which are regulated. This means that some of the control over their architecture is not in their own hands.

1.1 Middle Box Usage

Such large enterprises use Content Delivery Networks (CDNs) and NATs. One might wish that IPv6 was used to avoid NAT but this is not likely

to be the case inside the enterprise for many years.

Other type of middle boxes are frequently used by the data center infrastructure. This includes firewalls, load balancers, web servers, app servers, and middleware servers. A multi-tiered route is very common.

1.2 Routing and Other Protocols Used

Within the data center, such enterprises often use OSPF, EIGRP, BGP, and even RIP and static routes.

1.3 "Home-grown" Infrastructure

What is "home-grown"? For the "brick and mortars", if they do anything on their own, it will be to put up hardware infrastructure. For example, the connectivity in the swamps (which may have oil) or mining locations may be quite bad. Some companies put up, for example, their own microwave towers throughout the region.

What such enterprises do NOT do was to rewrite the code for the routers, middle boxes, etc.

1.4 Connections to Business Partners

Some of the most critical connections of large enterprises are to their business partners or regulatory bodies. For example, many financial institutions in the United States connect to the Federal Reserve; many insurance companies connect to the Medicare or Social Security systems.

2 Regulatory Requirements

Many of the "brick and mortar" enterprises are regulated by various legal structures such as HIPAA or PCI. These have an impact on the type of architecture which can be supported.

2.1 End-to-End Encryption

At times, there are regulatory requirements which enforce end-to-end encryption. For diagnostic and security purposes, it is important to be able to have visibility into the packets, routing and otherwise, so as to be able to manage the network.

If there is a protocol which does not allow for visibility, this can be quite problematic.

3 Applications

One of the advantages that large brick and mortar enterprises had in the dawn of the computer age is that they began to computerize early. Forty or fifty years later, what was once a competitive advantage now carries with it some burdens.

The number and nature of applications has multiplied greatly. Hundreds, if not thousands, of different applications are used. These range from the Stone Age (of computing) to the Space Age (of computing). That is, applications from those written in the 1960's to those using the most current technology must be supported.

Change can come at a glacial pace.

Having said that, many brick and mortars still see technology as their competitive advantage and are trying to keep pace.

4 Security Considerations

There are no security considerations.

5 IANA Considerations

There are no IANA considerations.

6 References

6.1 Normative References

6.2 Informative References

7 Acknowledgments

The authors would like to thank Steve Fenter for his comments.

Authors' Addresses

Nalini Elkins
Inside Products, Inc.
36A Upper Circle
Carmel Valley, CA 93924
United States
Phone: +1 831 659 8360
Email: nalini.elkins@insidethestack.com
<http://www.insidethestack.com>

Michael S. Ackermann
Blue Cross Blue Shield of Michigan
P.O. Box 2888
Detroit, Michigan 48231
United States
Phone: +1 310 460 4080
Email: mackermann@bcbsm.com
<http://www.bcbsm.com>

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 21, 2017

K. Patel
Arrcus, Inc.
A. Lindem
Cisco Systems
S. Zandi
Linkedin
G. Van de Velde
Nokia
June 19, 2017

Shortest Path Routing Extensions for BGP Protocol
draft-keyupate-idr-bgp-spf-03.txt

Abstract

Many Massively Scaled Data Centers (MSDCs) have converged on simplified layer 3 routing. Furthermore, requirements for operational simplicity have lead many of these MSDCs to converge on BGP as their single routing protocol for both their fabric routing and their Data Center Interconnect (DCI) routing. This document describes a solution which leverages BGP Link-State distribution and the Shortest Path First algorithm similar to Internal Gateway Protocols (IGPs) such as OSPF.

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 December 21, 2017.

Copyright Notice

Copyright (c) 2017 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 (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. BGP Shortest Path First (SPF) Motivation	3
1.2. Requirements Language	5
2. BGP Peering Models	5
2.1. BGP Single-Hop Peering on Network Node Connections	5
2.2. BGP Peering Between Directly Connected Network Nodes	5
2.3. BGP Peering in Route-Reflector or Controller Topology	5
3. BGP-LS Shortest Path Routing (SPF) SAFI	6
4. Extensions to BGP-LS	6
4.1. Node NLRI Usage and Modifications	6
4.2. Link NLRI Usage	7
4.3. Prefix NLRI Usage	7
5. Decision Process with SPF Algorithm	8
5.1. Phase-1 BGP NLRI Selection	8
5.2. Dual Stack Support	9
5.3. NEXT_HOP Manipulation	9
5.4. Error Handling	9
6. IANA Considerations	10
7. Security Considerations	10
7.1. Acknowledgements	10
7.2. Contributorss	10
8. References	11
8.1. Normative References	11
8.2. Information References	11
Authors' Addresses	12

1. Introduction

Many Massively Scaled Data Centers (MSDCs) have converged on simplified layer 3 routing. Furthermore, requirements for operational simplicity have lead many of these MSDCs to converge on BGP [RFC4271] as their single routing protocol for both their fabric routing and their Data Center Interconnect (DCI) routing. Requirements and procedures for using BGP are described in [RFC7938]. This document describes an alternative solution which leverages BGP-

LS [RFC7752] and the Shortest Path First algorithm similar to Internal Gateway Protocols (IGPs) such as OSPF [RFC2328].

[RFC4271] defines the Decision Process that is used to select routes for subsequent advertisement by applying the policies in the local Policy Information Base (PIB) to the routes stored in its Adj-RIBs-In. The output of the Decision Process is the set of routes that are announced by a BGP speaker to its peers. These selected routes are stored by a BGP speaker in the speaker's Adj-RIBs-Out according to policy.

[RFC7752] describes a mechanism by which link-state and TE information can be collected from networks and shared with external components using BGP. This is achieved by defining NLRI carried within BGP-LS AFI and BGP-LS SAFIs. The BGP-LS extensions defined in [RFC7752] makes use of the Decision Process defined in [RFC4271].

This document augments [RFC7752] by replacing its use of the existing Decision Process. The BGP-LS-SPF and BGP-LS-SPF-VPN AFI/SAFI are introduced to insure backward compatibility. The Phase 1 and 2 decision functions of the Decision Process are replaced with the Shortest Path Algorithm (SPF) also known as the Dijkstra Algorithm. The Phase 3 decision function is also simplified since it is no longer dependent on the previous phases. This solution avails the benefits of both BGP and SPF-based IGPs. These include TCP based flow-control, no periodic link-state refresh, and completely incremental NLRI advertisement. These advantages can reduce the overhead in MSDCs where there is a high degree of Equal Cost Multi-Path (ECMPs) and the topology is very stable. Additionally, using a SPF-based computation can support fast convergence and the computation of Loop-Free Alternatives (LFAs) [RFC5286] in the event of link failures. Furthermore, a BGP based solution lends itself to multiple peering models including those incorporating route-reflectors [RFC4456] or controllers.

Support for Multiple Topology Routing (MTR) as described in [RFC4915] is an area for further study dependent on deployment requirements.

1.1. BGP Shortest Path First (SPF) Motivation

Given that [RFC7938] already describes how BGP could be used as the sole routing protocol in an MSDC, one might question the motivation for defining an alternate BGP deployment model when a mature solution exists. For both alternatives, BGP offers the operational benefits of a single routing protocol. However, BGP SPF offers some unique advantages above and beyond standard BGP distance-vector routing.

A primary advantage is that all BGP speakers in the BGP SPF routing domain will have a complete view of the topology. This will allow support of ECMP, IP fast-reroute (e.g., Loop-Free Alternatives), Shared Risk Link Groups (SRLGs), and other routing enhancements without advertisement of addition BGP paths or other extensions. In short, the advantages of an IGP such as OSPF [RFC2328] are availed in BGP.

With the simplified BGP decision process as defined in Section 5.1, NLRI changes can be disseminated throughout the BGP routing domain much more rapidly (equivalent to IGPs with the proper implementation).

Another primary advantage is a potential reduction in NLRI advertisement. With standard BGP distance-vector routing, a single link failure may impact 100s or 1000s prefixes and result in the withdrawal or re-advertisement of the attendant NLRI. With BGP SPF, only the BGP speakers corresponding to the link NLRI need withdraw the corresponding BGP-LS Link NLRI. This advantage will contribute to both faster convergence and better scaling.

With controller and route-reflector peering models, BGP SPF advertisement and distributed computation require a minimal number of sessions and copies of the NLRI since only the latest version of the NLRI from the originator is required. Given that verification of the adjacencies is done outside of BGP (see Section 2), each BGP speaker will only need as many sessions and copies of the NLRI as required for redundancy (e.g., one for SPF computation and another for backup). Functions such as Optimized Route Reflection (ORR) are supported without extension by virtue of the primary advantages. Additionally, a controller could inject topology that is learned outside the BGP routing domain.

Given that controllers are already consuming BGP-LS NLRI [RFC7752], reusing for the BGP-LS SPF leverages the existing controller implementations.

Another potential advantage of BGP SPF is that both IPv6 and IPv4 can be supported in the same address family using the same topology. Although not described in this version of the document, multi-topology extensions can be used to support separate IPv4, IPv6, unicast, and multicast topologies while sharing the same NLRI.

Finally, the BGP SPF topology can be used as an underlay for other BGP address families (using the existing model) and realize all the above advantages. A simplified peering model using IPv6 link-local addresses as next-hops can be deployed similar to [RFC5549].

1.2. 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].

2. BGP Peering Models

Depending on the requirements, scaling, and capabilities of the BGP speakers, various peering models are supported. The only requirement is that all BGP speakers in the BGP SPF routing domain receive link-state NLRI on a timely basis, run an SPF calculation, and update their data plane appropriately. The content of the Link NLRI is described in Section 4.2.

2.1. BGP Single-Hop Peering on Network Node Connections

The simplest peering model is the one described in section 5.2.1 of [RFC7938]. In this model, EBGP single-hop sessions are established over direct point-to-point links interconnecting the network nodes. For the purposes of BGP SPF, Link NLRI is only advertised if a single-hop BGP session has been established and the Link-State/SPF address family capability has been exchanged [RFC4790] on the corresponding session. If the session goes down, the NLRI will be withdrawn.

2.2. BGP Peering Between Directly Connected Network Nodes

In this model, BGP speakers peer with all directly connected network nodes but the sessions may be multi-hop and the direct connection discovery and liveness detection for those connections are independent of the BGP protocol. How this is accomplished is outside the scope of this document. Consequently, there will be a single session even if there are multiple direct connections between BGP speakers. For the purposes of BGP SPF, Link NLRI is advertised as long as a BGP session has been established, the Link-State/SPF address family capability has been exchanged [RFC4790] and the corresponding link is up and considered operational.

2.3. BGP Peering in Route-Reflector or Controller Topology

In this model, BGP speakers peer solely with one or more Route Reflectors [RFC4456] or controllers. As in the previous model, direct connection discovery and liveness detection for those connections are done outside the BGP protocol. For the purposes of BGP SPF, Link NLRI is advertised as long as the corresponding link is up and considered operational.

3. BGP-LS Shortest Path Routing (SPF) SAFI

In order to replace the Phase 1 and 2 decision functions of the existing Decision Process with an SPF-based Decision Process and streamline the Phase 3 decision functions in a backward compatible manner, this draft introduces a couple AFI/SAFIs for BGP LS SPF operation. The BGP-LS-SPF (AF 16388 / SAFI TBD1) and BGP-LS-SPF-VPN (AFI 16388 / SAFI TBD2) [RFC4790] are allocated by IANA as specified in the Section 6.

4. Extensions to BGP-LS

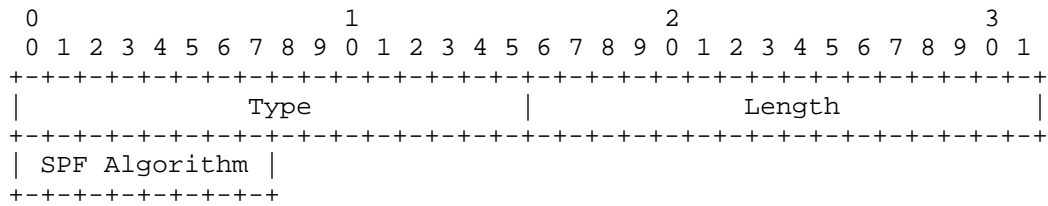
[RFC7752] describes a mechanism by which link-state and TE information can be collected from networks and shared with external components using BGP protocol. It contains two parts: definition of a new BGP NLRI that describes links, nodes, and prefixes comprising IGP link-state information and definition of a new BGP path attribute (BGP-LS attribute) that carries link, node, and prefix properties and attributes, such as the link and prefix metric or auxiliary Router-IDs of nodes, etc.

The BGP protocol will be used in the Protocol-ID field specified in table 1 of [I-D.ietf-idr-bgpls-segment-routing-epe]. The local and remote node descriptors for all NLRI will be the BGP Router-ID (TLV 516) and either the AS Number (TLV 512) [RFC7752] or the BGP Confederation Member (TLV 517)

[I-D.ietf-idr-bgpls-segment-routing-epe]. However, if the BGP Router-ID is known to be unique within the BGP Routing domain, it can be used as the sole descriptor.

4.1. Node NLRI Usage and Modifications

The SPF capability is a new Node Attribute TLV that will be added to those defined in table 7 of [RFC7752]. The new attribute TLV will only be applicable when BGP is specified in the Node NLRI Protocol ID field. The TBD TLV type will be defined by IANA. The new Node Attribute TLV will contain a single octet SPF algorithm field:



The SPF Algorithm may take the following values:

- 1 - Normal SPF
- 2 - Strict SPF

When computing the SPF for a given BGP routing domain, only BGP nodes advertising the SPF capability attribute will be included the Shortest Path Tree (SPT).

4.2. Link NLRI Usage

The criteria for advertisement of Link NLRI are discussed in Section 2.

Link NLRI is advertised with local and remote node descriptors as described above and unique link identifiers dependent on the addressing. For IPv4 links, the links local IPv4 (TLV 259) and remote IPv4 (TLV 260) addresses will be used. For IPv6 links, the local IPv6 (TLV 261) and remote IPv6 (TLV 262) addresses will be used. For unnumbered links, the link local/remote identifiers (TLV 258) will be used. For links supporting having both IPv4 and IPv6 addresses, both sets of descriptors may be included in the same Link NLRI. The link identifiers are described in table 5 of [RFC7752].

The link IGP metric attribute TLV (TLV 1095) as well as any others required for non-SPF purposes SHOULD be advertised. Algorithms such as setting the metric inversely to the link speed as done in the OSPF MIB [RFC4750] may be supported. However, this is beyond the scope of this document.

4.3. Prefix NLRI Usage

Prefix NLRI is advertised with a local descriptor as described above and the prefix and length used as the descriptors (TLV 265) as described in [RFC7752]. The prefix metric attribute TLV (TLV 1155) as well as any others required for non-SPF purposes SHOULD be advertised. For loopback prefixes, the metric should be 0. For non-loopback, the setting of the metric is beyond the scope of this document.

5. Decision Process with SPF Algorithm

The Decision Process described in [RFC4271] takes place in three distinct phases. The Phase 1 decision function of the Decision Process is responsible for calculating the degree of preference for each route received from a Speaker's peer. The Phase 2 decision function is invoked on completion of the Phase 1 decision function and is responsible for choosing the best route out of all those available for each distinct destination, and for installing each chosen route into the Loc-RIB. The combination of the Phase 1 and 2 decision functions is also known as a Path vector algorithm.

When BGP-LS-SPF NLRI is received, all that is required is to determine whether it is the best-path by examining the Node-ID as described in Section 5.1. If the best-path NLRI had changed, it will be advertised to other BGP-LS-SPF peers. If the attributes have changed a BGP SPF calculation will be scheduled. However, a changed best-path can be advertised to other peer immediately and propagation of changes can approach IGP convergence times.

The SPF based Decision process starts with selecting only those Node NLRI whose SPF capability TLV matches with the local BGP speaker's SPF capability TLV value. Since Link-State NLRI always contains the local descriptor [RFC7752], it will only be originated by a single BGP speaker in the BGP routing domain. These selected Node NLRI and their Link/Prefix NLRI are used to build a directed graph during the SPF computation. The best paths for BGP prefixes are installed as a result of the SPF process.

The Phase 3 decision function of the Decision Process [RFC4271] is also simplified since under normal SPF operation, a BGP speaker would advertise the NLRI selected for the SPF to all BGP peers with the BGP-LS/BGP-SPF AFI/SAFI. Application of policy would not be prevented but would normally not be necessary.

5.1. Phase-1 BGP NLRI Selection

The rules for NLRI selection are greatly simplified from [RFC4271].

1. If the NLRI is received from the BGP speaker originating the NLRI (as determined by the comparing BGP Router ID in the NLRI Node identifiers with the BGP speaker Router ID), then it is preferred over the same NLRI from non-originators.
2. The final tie-breaker is the NLRI from the BGP Speaker with the numerically largest BGP Router ID.

The modified Decision Process with SPF algorithm uses the metric from Link and Prefix NLRI Attribute TLVs [RFC7752]. As a result, any attributes that would influence the Decision process defined in [RFC4271] like ORIGIN, MULTI_EXIT_DISC, and LOCAL_PREF attributes are ignored by the SPF algorithm. Furthermore, the NEXT_HOP attribute value is preserved and validated but otherwise ignored during the SPF or best-path.

5.2. Dual Stack Support

The SPF based decision process operates on Node, Link, and Prefix NLRIs that support both IPv4 and IPv6 addresses. Whether to run a single SPF instance or multiple SPF instances for separate AFs is a matter of a local implementation. Normally, IPv4 next-hops are calculated for IPv4 prefixes and IPv6 next-hops are calculated for IPv6 prefixes. However, an interesting use-case is deployment of [RFC5549] where IPv6 link-local next-hops are calculated for both IPv4 and IPv6 prefixes. As stated in Section 1, support for Multiple Topology Routing (MTR) is an area for future study.

5.3. NEXT_HOP Manipulation

A BGP speaker that supports SPF extensions MAY interact with peers that don't support SPF extensions. If the BGP Link-State address family is advertised to a peer not supporting the SPF extensions described herein, then the BGP speaker MUST conform to the NEXT_HOP rules mentioned in [RFC4271] when announcing the Link-State address family routes to those peers.

All BGP peers that support SPF extensions would locally compute the NEXT_HOP values as result of the SPF process. As a result, the NEXT_HOP attribute is always ignored on receipt. However BGP speakers should set the NEXT_HOP address according to the NEXT_HOP attribute rules mentioned in [RFC4271].

5.4. Error Handling

When a BGP speaker receives a BGP Update containing a malformed SPF Capability TLV in the Node NLRI BGP-LS Attribute [RFC7752], it MUST ignore the received TLV and the Node NLRI and not pass it to other BGP peers as specified in [RFC7606]. When discarding a Node NLRI with malformed TLV, a BGP speaker SHOULD log an error for further analysis.

6. IANA Considerations

This document defines a couple AFI/SAFIs for BGP LS SPF operation and requests IANA to assign the BGP-LS-SPF AFI 16388 / SAFI TBD1 and the BGP-LS-SPF-VPN AFI 16388 / SAFI TBD2 as described in [RFC4750].

This document also defines two attribute TLV for BGP LS NLRI. We request IANA to assign TLVs for the SPF capability from the "BGP-LS Node Descriptor, Link Descriptor, Prefix Descriptor, and Attribute TLVs" Registry. Additionally, IANA is requested to create a new registry for "BGP-LS SPF Capability Algorithms" for the value of the algorithm both in the BGP-LS Node Attribute TLV and the BGP SPF Capability. The initial assignments are:

Value(s)	Assignment Policy
0	Reserved (not to be assigned)
1	SPF
2	Strict SPF
3-254	Unassigned (IETF Review)
255	Reserved (not to be assigned)

BGP-LS SPF Capability Algorithms

7. Security Considerations

This extension to BGP does not change the underlying security issues inherent in the existing [RFC4724] and [RFC4271].

7.1. Acknowledgements

The authors would like to thank for the review and comments.

7.2. Contributorss

In addition to the authors listed on the front page, the following co-authors have contributed to the document.

Derek Yeung
Arrcus, Inc.
derek@arrcus.com

Abhay Roy
Cisco Systems
akr@cisco.com

Venu Venugopal
Cisco Systems
venuv@cisco.com

8. References

8.1. Normative References

- [I-D.ietf-idr-bgpls-segment-routing-epe]
Previdi, S., Filsfils, C., Patel, K., Ray, S., and J. Dong, "BGP-LS extensions for Segment Routing BGP Egress Peer Engineering", draft-ietf-idr-bgpls-segment-routing-epe-12 (work in progress), April 2017.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<http://www.rfc-editor.org/info/rfc4271>>.
- [RFC7606] Chen, E., Ed., Scudder, J., Ed., Mohapatra, P., and K. Patel, "Revised Error Handling for BGP UPDATE Messages", RFC 7606, DOI 10.17487/RFC7606, August 2015, <<http://www.rfc-editor.org/info/rfc7606>>.
- [RFC7752] Gredler, H., Ed., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP", RFC 7752, DOI 10.17487/RFC7752, March 2016, <<http://www.rfc-editor.org/info/rfc7752>>.

8.2. Information References

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

- [RFC4456] Bates, T., Chen, E., and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)", RFC 4456, DOI 10.17487/RFC4456, April 2006, <<http://www.rfc-editor.org/info/rfc4456>>.
- [RFC4724] Sangli, S., Chen, E., Fernando, R., Scudder, J., and Y. Rekhter, "Graceful Restart Mechanism for BGP", RFC 4724, DOI 10.17487/RFC4724, January 2007, <<http://www.rfc-editor.org/info/rfc4724>>.
- [RFC4750] Joyal, D., Ed., Galecki, P., Ed., Giacalone, S., Ed., Coltun, R., and F. Baker, "OSPF Version 2 Management Information Base", RFC 4750, DOI 10.17487/RFC4750, December 2006, <<http://www.rfc-editor.org/info/rfc4750>>.
- [RFC4790] Newman, C., Duerst, M., and A. Gulbrandsen, "Internet Application Protocol Collation Registry", RFC 4790, DOI 10.17487/RFC4790, March 2007, <<http://www.rfc-editor.org/info/rfc4790>>.
- [RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF", RFC 4915, DOI 10.17487/RFC4915, June 2007, <<http://www.rfc-editor.org/info/rfc4915>>.
- [RFC5286] Atlas, A., Ed. and A. Zinin, Ed., "Basic Specification for IP Fast Reroute: Loop-Free Alternates", RFC 5286, DOI 10.17487/RFC5286, September 2008, <<http://www.rfc-editor.org/info/rfc5286>>.
- [RFC5549] Le Faucheur, F. and E. Rosen, "Advertising IPv4 Network Layer Reachability Information with an IPv6 Next Hop", RFC 5549, DOI 10.17487/RFC5549, May 2009, <<http://www.rfc-editor.org/info/rfc5549>>.
- [RFC7938] Lapukhov, P., Premji, A., and J. Mitchell, Ed., "Use of BGP for Routing in Large-Scale Data Centers", RFC 7938, DOI 10.17487/RFC7938, August 2016, <<http://www.rfc-editor.org/info/rfc7938>>.

Authors' Addresses

Keyur Patel
Arrcus, Inc.

Email: keyur@arrcus.com

Acee Lindem
Cisco Systems
170 W. Tasman Drive
San Jose, CA 95134
USA

Email: acee@cisco.com

Shawn Zandi
Linkedin
222 2nd Street
San Francisco, CA 94105
USA

Email: szandi@linkedin.com

Gunter Van de Velde
Nokia
Antwerp
Belgium

Email: gunter.van_de_velde@nokia.com

Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 1, 2018

T. Przygienda
Juniper Networks
A. Sharma
Comcast
J. Drake
A. Atlas
Juniper Networks
October 28, 2017

RIFT: Routing in Fat Trees
draft-przygienda-rift-03

Abstract

This document outlines a specialized, dynamic routing protocol for Clos and fat-tree network topologies. The protocol (1) deals with automatic construction of fat-tree topologies based on detection of links, (2) minimizes the amount of routing state held at each level, (3) automatically prunes the topology distribution exchanges to a sufficient subset of links, (4) supports automatic disaggregation of prefixes on link and node failures to prevent black-holing and suboptimal routing, (5) allows traffic steering and re-routing policies and ultimately (6) provides mechanisms to synchronize a limited key-value data-store that can be used after protocol convergence to e.g. bootstrap higher levels of functionality on nodes.

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 May 1, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Requirements Language	4
2. Reference Frame	4
2.1. Terminology	4
2.2. Topology	6
3. Requirement Considerations	8
4. RIFT: Routing in Fat Trees	10
4.1. Overview	10
4.2. Specification	10
4.2.1. Transport	10
4.2.2. Link (Neighbor) Discovery (LIE Exchange)	11
4.2.3. Topology Exchange (TIE Exchange)	11
4.2.3.1. Topology Information Elements	12
4.2.3.2. South- and Northbound Representation	12
4.2.3.3. Flooding	14
4.2.3.4. TIE Flooding Scopes	15
4.2.3.5. Initial and Periodic Database Synchronization	17
4.2.3.6. Purging	17
4.2.3.7. Southbound Default Route Origination	18
4.2.3.8. Optional Automatic Flooding Reduction and Partitioning	18
4.2.4. Policy-Guided Prefixes	19
4.2.4.1. Ingress Filtering	20
4.2.4.2. Applying Policy	21
4.2.4.3. Store Policy-Guided Prefix for Route Computation and Regeneration	22
4.2.4.4. Re-origination	22
4.2.4.5. Overlap with Disaggregated Prefixes	23
4.2.5. Reachability Computation	23
4.2.5.1. Northbound SPF	23
4.2.5.2. Southbound SPF	24

4.2.5.3. East-West Forwarding Within a Level	24
4.2.6. Attaching Prefixes	24
4.2.7. Attaching Policy-Guided Prefixes	26
4.2.8. Automatic Disaggregation on Link & Node Failures	27
4.2.9. Optional Autoconfiguration	30
4.3. Further Mechanisms	30
4.3.1. Overload Bit	30
4.3.2. Optimized Route Computation on Leafs	31
4.3.3. Key/Value Store	31
4.3.4. Interactions with BFD	31
4.3.5. Leaf to Leaf Procedures	32
4.3.6. Other End-to-End Services	32
4.3.7. Address Family and Multi Topology Considerations	33
4.3.8. Reachability of Internal Nodes in the Fabric	33
4.3.9. One-Hop Healing of Levels with East-West Links	33
5. Examples	33
5.1. Normal Operation	33
5.2. Leaf Link Failure	35
5.3. Partitioned Fabric	36
5.4. Northbound Partitioned Router and Optional East-West Links	37
6. Implementation and Operation: Further Details	38
6.1. Considerations for Leaf-Only Implementation	39
6.2. Adaptations to Other Proposed Data Center Topologies	39
6.3. Originating Non-Default Route Southbound	40
7. Security Considerations	40
8. Information Elements Schema	40
8.1. common.thrift	41
8.2. encoding.thrift	44
9. IANA Considerations	49
10. Security Considerations	49
11. Acknowledgments	49
12. References	49
12.1. Normative References	49
12.2. Informative References	51
Authors' Addresses	52

1. Introduction

Clos [CLOS] and Fat-Tree [FATTREE] have gained prominence in today's networking, primarily as a result of a the paradigm shift towards a centralized data-center based architecture that is poised to deliver a majority of computation and storage services in the future. The existing set of dynamic routing protocols was geared originally towards a network with an irregular topology and low degree of connectivity and consequently several attempts to adapt those have been made. Most successfully BGP [RFC4271] [RFC7938] has been extended to this purpose, not as much due to its inherent suitability

to solve the problem but rather because the perceived capability to modify it "quicker" and the immanent difficulties with link-state [DIJKSTRA] based protocols to fulfill certain of the resulting requirements.

In looking at the problem through the very lens of its requirements an optimal approach does not seem to be a simple modification of either a link-state (distributed computation) or distance-vector (diffused computation) approach but rather a mixture of both, colloquially best described as 'link-state towards the spine' and 'distance vector towards the leafs'. The balance of this document details the resulting protocol.

1.1. 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].

2. Reference Frame

2.1. Terminology

This section presents the terminology used in this document. It is assumed that the reader is thoroughly familiar with the terms and concepts used in OSPF [RFC2328] and IS-IS [RFC1142], [ISO10589] as well as the according graph theoretical concepts of shortest path first (SPF) [DIJKSTRA] computation and directed acyclic graphs (DAG).

Level: Clos and Fat Tree networks are trees and 'level' denotes the set of nodes at the same height in such a network, where the bottom level is level 0. A node has links to nodes one level down and/or one level up. Under some circumstances, a node may have links to nodes at the same level. As footnote: Clos terminology uses often the concept of "stage" but due to the folded nature of the Fat Tree we do not use it to prevent misunderstandings.

Spine/Aggregation/Edge Levels: Traditional names for Level 2, 1 and 0 respectively. Level 0 is often called leaf as well.

Point of Delivery (PoD): A self-contained vertical slice of a Clos or Fat Tree network containing normally only level 0 and level 1 nodes. It communicates with nodes in other PoDs via the spine.

Spine: The set of nodes that provide inter-PoD communication. These nodes are also organized into levels (typically one, three, or five levels). Spine nodes do not belong to any PoD and are assigned the PoD value 0 to indicate this.

Leaf: A node at level 0.

Connected Spine: In case a spine level represents a connected graph (discounting links terminating at different levels), we call it a "connected spine", in case a spine level consists of multiple partitions, we call it a "disconnected" or "partitioned spine". In other terms, a spine without east-west links is disconnected and is the typical configuration for Clos and Fat Tree networks.

South/Southbound and North/Northbound (Direction): When describing protocol elements and procedures, we will be using in different situations the directionality of the compass. I.e., 'south' or 'southbound' mean moving towards the bottom of the Clos or Fat Tree network and 'north' and 'northbound' mean moving towards the top of the Clos or Fat Tree network.

Northbound Link: A link to a node one level up or in other words, one level further north.

Southbound Link: A link to a node one level down or in other words, one level further south.

East-West Link: A link between two nodes at the same level. East-west links are normally not part of Clos or "fat-tree" topologies.

Leaf shortcuts (L2L): East-west links at leaf level will need to be differentiated from East-west links at other levels.

Southbound representation: Information sent towards a lower level representing only limited amount of information.

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. It can be thought of as largely equivalent to ISIS LSPs or OSPF LSA. We will talk about N-TIEs when talking about TIEs in the northbound representation and S-TIEs for the southbound equivalent.

Node TIE: This is an acronym for a "Node Topology Information Element", largely equivalent to OSPF Node LSA, i.e. it contains all neighbors the node discovered and information about node itself.

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 N-TIE and in case of S-TIE the necessary default and de-aggregated prefixes the node passes southbound.

Policy-Guided Information: Information that is passed in either southbound direction or north-bound direction by the means of diffusion and can be filtered via policies. Policy-Guided Prefixes and KV Ties are examples of Policy-Guided Information.

Key Value TIE: A S-TIE that is carrying a set of key value pairs [DYNAMO]. It can be used to distribute information in the southbound direction within the protocol.

TIDE: Topology Information Description Element, equivalent to CSNP in ISIS.

TIRE: Topology Information Request Element, equivalent to PSNP in ISIS. It can both confirm received and request missing TIEs.

PGP: Policy-Guided Prefixes allow to support traffic engineering that cannot be achieved by the means of SPF computation or normal node and prefix S-TIE origination. S-PGPs are propagated in south direction only and N-PGPs follow northern direction strictly.

De-aggregation/Disaggregation: Process in which a node decides to advertise certain prefixes it received in N-TIEs to prevent black-holing and suboptimal routing upon link failures.

LIE: This is an acronym for a "Link Information Element", largely equivalent to HELLOs in IGPs and exchanged over all the links between systems running RIFT to form adjacencies.

FL: Flooding Leader for a specific system has a dedicated role to flood TIEs of that system.

2.2. Topology

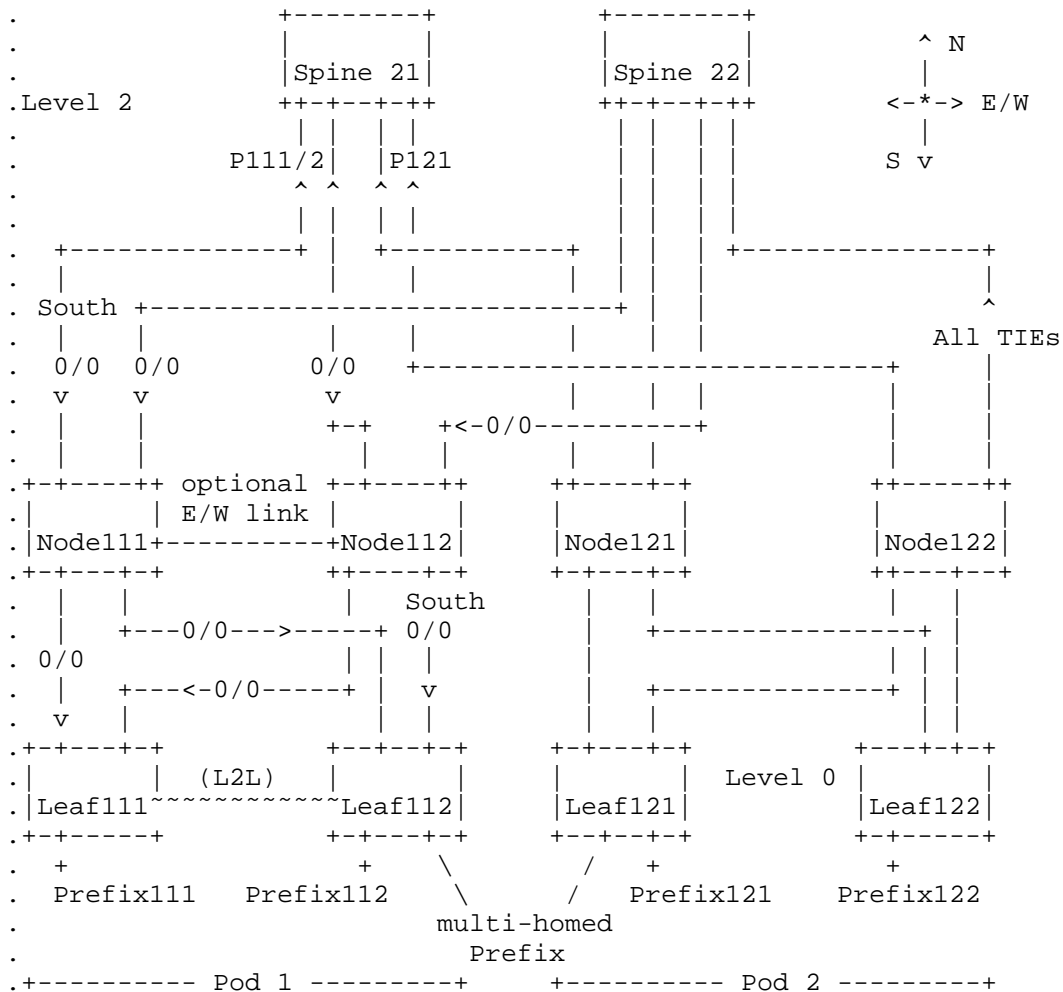


Figure 1: A two level spine-and-leaf topology

We will use this topology (called commonly a fat tree/network in modern DC considerations [VAHDAT08] as homonym to the original definition of the term [FATTREE]) in all further considerations. It depicts a generic "fat-tree" and the concepts explained in three levels here carry by induction for further levels and higher degrees of connectivity.

3. Requirement Considerations

[RFC7938] gives the original set of requirements augmented here based upon recent experience in the operation of fat-tree networks.

- REQ1: The control protocol should discover the physical links automatically and be able to detect cabling that violates fat-tree topology constraints. It must react 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. E.g. connecting a leaf to a spine at level 2 should be detected and ideally prevented.
- REQ2: A node without any configuration beside default values should come up as leaf in any PoD it is introduced into. Optionally, it must be possible to configure nodes to restrict their participation to the PoD(s) targeted at any level.
- REQ3: Optionally, the protocol should allow to provision data centers where the individual switches carry no configuration information and are all deriving their level from a "seed". Observe that this requirement may collide with the desire to detect cabling misconfiguration and with that only one of the requirements can be fully met in a chosen configuration mode.
- REQ4: The solution should allow for minimum size routing information base and forwarding tables at leaf level for speed, cost and simplicity reasons. Holding excessive amount of information away from leaf nodes simplifies operation and lowers cost of the underlay.
- REQ5: Very high degree of ECMP (and ideally non equal cost) must be supported. Maximum ECMP is currently understood as the most efficient routing approach to maximize the throughput of switching fabrics [MAKSIC2013].
- REQ6: Traffic engineering should be allowed by modification of prefixes and/or their next-hops.
- REQ7: The solution should allow for access to link states of the whole topology to enable efficient support for modern control architectures like SPRING [RFC7855] or PCE [RFC4655].

- REQ8: The solution should easily accommodate opaque data to be carried throughout the topology to subsets of nodes. This can be used for many purposes, one of them being a key-value store that allows bootstrapping of nodes based right at the time of topology discovery.
- REQ9: Nodes should be taken out and introduced into production with minimum wait-times and minimum of "shaking" of the network, i.e. radius of propagation (often called "blast radius") of changed information should be as small as feasible.
- REQ10: The protocol should allow for maximum aggregation of carried routing information while at the same time automatically de-aggregating the prefixes to prevent black-holing in case of failures. The de-aggregation should support maximum possible ECMP/N-ECMP remaining after failure.
- REQ11: Reducing the scope of communication needed throughout the network on link and state failure, as well as reducing advertisements of repeating, idiomatic or policy-guided information in stable state is highly desirable since it leads to better stability and faster convergence behavior.
- REQ12: Once a packet traverses a link in a "southbound" direction, it must not take any further "northbound" steps along its path to delivery to its destination under normal conditions. Taking a path through the spine in cases where a shorter path is available is highly undesirable.
- REQ13: Parallel links between same set of nodes must be distinguishable for SPF, failure and traffic engineering purposes.
- REQ14: The protocol must not rely on interfaces having discernible unique addresses, i.e. it must operate in presence of unnumbered links (even parallel ones) or links of a single node having same addresses.

Following list represents possible requirements and requirements under discussion:

- PEND1: Supporting anything but point-to-point links is a non-requirement. Questions remain: for connecting to the leaves, is there a case where multipoint is desirable? One could still model it as point-to-point links; it seems there is no need for anything more than a NBMA-type construct.

PEND2: What is the maximum scale of number leaf prefixes we need to carry. Is 500'000 enough ?

Finally, following are the non-requirements:

NONREQ1: Broadcast media support is unnecessary.

NONREQ2: Purging is unnecessary given its fragility and complexity and today's large memory size on even modest switches and routers.

NONREQ3: Special support for layer 3 multi-hop adjacencies is not part of the protocol specification. Such support can be easily provided by using tunneling technologies the same way IGP today are solving the problem.

4. RIFT: Routing in Fat Trees

Derived from the above requirements we present a detailed outline of a protocol optimized for Routing in Fat Trees (RIFT) that in most abstract terms has many properties of a modified link-state protocol [RFC2328][RFC1142] when "pointing north" and path-vector [RFC4271] protocol when "pointing south". Albeit an unusual combination, it does quite naturally exhibit the desirable properties we seek.

4.1. Overview

The novel property of RIFT is that it floods northbound "flat" link-state information so that each level understands the full topology of levels south of it. In contrast, in the southbound direction the protocol operates like a path vector protocol or rather a distance vector with implicit split horizon since the topology constraints make a diffused computation front propagating in all directions unnecessary.

4.2. Specification

4.2.1. Transport

All protocol elements are carried over UDP. Once QUIC [QUIC] achieves the desired stability in deployments it may prove a valuable candidate for TIE transport.

All packet formats are defined in Thrift models in Section 8.

4.2.2. Link (Neighbor) Discovery (LIE Exchange)

LIE exchange happens over well-known administratively locally scoped IPv4 multicast address [RFC2365] or link-local multicast scope for IPv6 [RFC4291] and SHOULD be sent with a TTL of 1 to prevent RIFT information reaching beyond a single link in the topology. LIEs are exchanged over all links running RIFT.

Each node is provisioned with the level at which it is operating and its PoD. A default level and PoD of zero are assumed, meaning that leafs do not need to be configured with a level (or even PoD). Nodes in the spine are configured with a PoD of zero. This information is propagated in the LIEs exchanged. Adjacencies are formed if and only if

1. the node is in the same PoD or either the node or the neighbor advertises "any" PoD membership (PoD# = 0) AND
2. the neighboring node is at most one level away AND
3. the neighboring node is running the same MAJOR schema version AND
4. the neighbor is not member of some PoD while the node has a northbound adjacency already joining another PoD AND
5. the advertised MTUs match on both sides.

A node configured with "any" PoD membership MUST, after building first northbound adjacency making it participant in a PoD, advertise that PoD as part of its LIEs.

LIEs arriving with a TTL larger than 1 MUST be ignored.

LIE exchange uses three-way handshake mechanism [RFC5303]. Precise final state machines will be provided in later versions of this specification. LIE packets contain nonces and may contain a SHA-1 [RFC6234] over nonces and some of the LIE data which prevents corruption and replay attacks. TIE flooding reuses those nonces to prevent mismatches and can use those for security purposes in case it is using QUIC [QUIC]. Section 7 will address the precise security mechanisms in the future.

4.2.3. Topology Exchange (TIE Exchange)

4.2.3.1. Topology Information Elements

Topology and reachability information in RIFT is conveyed by the means of TIEs which have good amount of commonalities with LSAs in OSPF.

TIE exchange mechanism uses port indicated by each node in the LIE exchange and the interface on which the adjacency has been formed as destination. It SHOULD use TTL of 1 as well.

TIEs contain sequence numbers, lifetimes and a type. Each type has a large identifying number space and information is spread across possibly many TIEs of a certain type by the means of a hash function that a node or deployment can individually determine. One extreme side of the scale is a prefix per TIE which leads to BGP-like behavior vs. dense packing into few TIEs leading to more traditional IGP trade-off with fewer TIEs. An implementation may even rehash at the cost of significant amount of re-advertisements of TIEs.

More information about the TIE structure can be found in the schema in Section 8.

4.2.3.2. South- and Northbound Representation

As a central concept to RIFT, each node represents itself differently depending on the direction in which it is advertising information. More precisely, a spine node represents two different databases to its neighbors depending whether it advertises TIEs to the north or to the south/sideways. We call those differing TIE databases either south- or northbound (S-TIEs and N-TIEs) depending on the direction of distribution.

The N-TIEs hold all of the node's adjacencies, local prefixes and northbound policy-guided prefixes while the S-TIEs hold only all of the node's neighbors and the default prefix with necessary disaggregated prefixes and southbound policy-guided prefixes. We will explain this in detail further in Section 4.2.8 and Section 4.2.4.

The TIEs are symmetric in both directions and Table 1 provides a quick reference to the different TIE types including direction and their function.

TIE-Type	Content
node N-TIE	node properties and adjacencies
node S-TIE	same content as node N-TIE
Prefix N-TIE	contains nodes' directly reachable prefixes
Prefix S-TIE	contains originated defaults and de-aggregated prefixes
PGP N-TIE	contains nodes north PGPs
PGP S-TIE	contains nodes south PGPs
KV N-TIE	contains nodes northbound KVs
KV S-TIE	contains nodes southbound KVs

Table 1: TIE Types

As an example illustrating a databases holding both representations, consider the topology in Figure 1 with the optional link between node 111 and node 112 (so that the flooding on an east-west link can be shown). This example assumes unnumbered interfaces. First, here are the TIEs generated by some nodes. For simplicity, the key value elements and the PGP elements which may be included in their S-TIEs or N-TIEs are not shown.

Spine21 S-TIEs:

Node S-TIE:

```
NodeElement(layer=2, neighbors((Node111, layer 1, cost 1),
    (Node112, layer 1, cost 1), (Node121, layer 1, cost 1),
    (Node122, layer 1, cost 1)))
```

Prefix S-TIE:

```
SouthPrefixesElement(prefixes(0/0, cost 1), (::/0, cost 1))
```

Node111 S-TIEs:

Node S-TIE:

```
NodeElement(layer=1, neighbors((Spine21, layer 2, cost 1),
    (Spine22, layer 2, cost 1), (Node112, layer 1, cost 1),
    (Leaf111, layer 0, cost 1), (Leaf112, layer 0, cost 1)))
```

Prefix S-TIE:

```
SouthPrefixesElement(prefixes(0/0, cost 1), (::/0, cost 1))
```

Node111 N-TIEs:

Node N-TIE:

```
NodeLinkElement(layer=1,
    neighbors((Spine21, layer 2, cost 1, links(...)),
    (Spine22, layer 2, cost 1, links(...)),
    (Node112, layer 1, cost 1, links(...)),
    (Leaf111, layer 0, cost 1, links(...)),
    (Leaf112, layer 0, cost 1, links(...))))
```

Prefix N-TIE:

```
NorthPrefixesElement(prefixes(Node111.loopback)
```

```

Node121 S-TIEs:
Node S-TIE:
  NodeElement(layer=1, neighbors((Spine21,layer 2,cost 1),
    (Spine22, layer 2, cost 1), (Leaf121, layer 0, cost 1),
    (Leaf122, layer 0, cost 1)))
Prefix S-TIE:
  SouthPrefixesElement(prefixes(0/0, cost 1), (::/0, cost 1))

Node121 N-TIEs:
Node N-TIE:
  NodeLinkElement(layer=1,
    neighbors((Spine21, layer 2, cost 1, links(...)),
    (Spine22, layer 2, cost 1, links(...)),
    (Leaf121, layer 0, cost 1, links(...)),
    (Leaf122, layer 0, cost 1, links(...))))
Prefix N-TIE:
  NorthPrefixesElement(prefixes(Node121.loopback)

Leaf112 N-TIEs:
Node N-TIE:
  NodeLinkElement(layer=0,
    neighbors((Node111, layer 1, cost 1, links(...)),
    (Node112, layer 1, cost 1, links(...))))
Prefix N-TIE:
  NorthPrefixesElement(prefixes(Leaf112.loopback, Prefix112,
    Prefix_MH))

```

Figure 2: example TIES generated in a 2 level spine-and-leaf topology

4.2.3.3. Flooding

The mechanism used to distribute TIES is the well-known (albeit modified in several respects to address fat tree requirements) flooding mechanism used by today's link-state protocols. Albeit initially more demanding to implement it avoids many problems with diffused computation update style used by path vector. As described before, TIES themselves are transported over UDP with the ports indicates in the LIE exchanges and using the destination address (for unnumbered IPv4 interfaces same considerations apply as in equivalent OSPF case) on which the LIE adjacency has been formed.

Precise final state machines and procedures will be provided in later versions of this specification.

4.2.3.4. TIE Flooding Scopes

In a somewhat analogous fashion to link-local, area and domain flooding scopes, RIFT defines several complex "flooding scopes" depending on the direction and type of TIE propagated.

Every N-TIE is flooded northbound, providing a node at a given level with the complete topology of the Clos or Fat Tree network underneath it, including all specific prefixes. This means that a packet received from a node at the same or lower level whose destination is covered by one of those specific prefixes may be routed directly towards the node advertising that prefix rather than sending the packet to a node at a higher level.

A node's node S-TIEs, consisting of all node's adjacencies and prefix S-TIEs with default IP prefix and disaggregated prefixes, are flooded southbound in order to allow the nodes one level down to see connectivity of the higher level as well as reachability to the rest of the fabric. In order to allow a E-W disconnected node in a given level to receive the S-TIEs of other nodes at its level, every *NODE* S-TIE is "reflected" northbound to level from which it was received. It should be noted that east-west links are included in South TIE flooding; those TIEs need to be flooded to satisfy algorithms in Section 4.2.5. In that way nodes at same level can learn about each other without a more southern level, e.g. in case of leaf level. The precise flooding scopes are given in Table 2. Those rules govern in a symmetric fashion what SHOULD be included in TIEs towards neighbors.

Node S-TIE "reflection" allows to support disaggregation on failures describes in Section 4.2.8 and flooding reduction in Section 4.2.3.8.

Packet Type vs. Peer Direction	South	North	East-West
node S-TIE	flood own only	flood if TIE originator's level is higher than own level	same as South
other S-TIE	flood own only	flood only if TIE originator is equal peer	same as South
all N-TIES	never flood	flood always	same as South
TIDE	include TIEs in flooding scope	include TIEs in flooding scope	same as South
TIRE	include all N-TIES and all peer's self- originated TIEs and all node S-TIEs	include only if TIE originator is equal peer	same as South

Table 2: Flooding Scopes

As an example to illustrate these rules, consider using the topology in Figure 1, with the optional link between node 111 and node 112, and the associated TIEs given in Figure 2. The flooding from particular nodes of the TIEs is given in Table 3.

Router floods to	Neighbor	TIEs
-----	-----	-----
Leaf111	Node112	Leaf111 N-TIEs, Node111 node S-TIE
Leaf111	Node111	Leaf111 N-TIEs, Node112 node S-TIE
Node111	Leaf111	Node111 S-TIEs
Node111	Leaf112	Node111 S-TIEs
Node111	Node112	Node111 S-TIEs
Node111	Spine21	Node111 N-TIEs, Leaf111 N-TIEs, Leaf112 N-TIEs, Spine22 node S-TIE
Node111	Spine22	Node111 N-TIEs, Leaf111 N-TIEs, Leaf112 N-TIEs, Spine21 node S-TIE
...
Spine21	Node111	Spine21 S-TIEs
Spine21	Node112	Spine21 S-TIEs
Spine21	Node121	Spine21 S-TIEs
Spine21	Node122	Spine21 S-TIEs
...

Table 3: Flooding some TIEs from example topology

4.2.3.5. Initial and Periodic Database Synchronization

The initial exchange of RIFT is modeled after ISIS with TIDE being equivalent to CSNP and TIRE playing the role of PSNP. The content of TIDEs and TIREs is governed by Table 2.

4.2.3.6. Purging

RIFT does not purge information that has been distributed by the protocol. Purging mechanisms in other routing protocols have proven through many years of experience to be complex and fragile. Abundant amounts of memory are available today even on low-end platforms. The information will age out and all computations will deliver correct results if a node leaves the network due to the new information distributed by its adjacent nodes.

Once a RIFT node issues a TIE with an ID, it MUST preserve the ID as long as feasible (also when the protocol restarts), even if the TIE loses all content. The re-advertisement of empty TIE fulfills the purpose of purging any information advertised in previous versions. The originator is free to not re-originate the according empty TIE again or originate an empty TIE with relatively short lifetime to prevent large number of long-lived empty stubs polluting the network. Each node will timeout and clean up the according empty TIEs independently.

4.2.3.7. Southbound Default Route Origination

Under certain conditions nodes issue a default route in their South Prefix TIEs.

A node X that

1. is NOT overloaded AND
2. has southbound or east-west adjacencies

originates in its south prefix TIE such a default route IIF

1. all other nodes at X's' level are overloaded OR
2. all other nodes at X's' level have NO northbound adjacencies OR
3. X has computed reachability to a default route during N-SPF.

The term "all other nodes at X's' level" describes obviously just the nodes at the same level in the POD with a viable lower layer (otherwise the node S-TIEs cannot be reflected and the nodes in e.g. POD 1 nad POD 2 are "invisible" to each other).

A node originating a southbound default route MUST install a default discard route if it did not compute a default route during N-SPF.

4.2.3.8. Optional Automatic Flooding Reduction and Partitioning

Several nodes can, but strictly only under conditions defined below, run a hashing function based on TIE originator value and partition flooding between them.

Steps for flooding reduction and partitioning:

1. select all nodes in the same level for which node S-TIEs have been received and which have precisely the same non-empty sets of respectively north and south neighbor adjacencies and support flooding reduction (overload bits are ignored) and then
2. run on the chosen set a hash algorithm using nodes flood priorities and IDs to select flooding leader and backup per TIE originator ID, i.e. each node floods immediately through to all its necessary neighbors TIEs that it received with an originator ID that makes it the flooding leader or backup for this originator. The preference (higher is better) is computed as $\text{XOR}(\text{TIE-ORIGINATOR-ID} \ll 1, \sim \text{OWN-SYSTEM-ID})$, whereas \ll is a non-circular shift and \sim is bit-wise NOT.

3. In the very unlikely case of hash collisions on either of the two nodes with highest values (i.e. either does NOT produce unique hashes as compared to all other hash values), the node running the election does not attempt to reduce flooding.

Additional rules for flooding reduction and partitioning:

1. A node always floods its own TIEs
2. A node generates TIDEs as usual but when receiving TIREs with requests for TIEs for a node for which it is not a flooding leader or backup it ignores such TIDEs on first request only. Normally, the flooding leader should satisfy the requestor and with that no further TIREs for such TIEs will be generated. Otherwise, the next set of TIDEs and TIREs will lead to flooding independent of the flooding leader status.
3. A node receiving a TIE originated by a node for which it is not a flooding leader floods such TIEs only when receiving an out-of-date TIDE for them, except for the first one.

The mechanism can be implemented optionally in each node. The capability is carried in the node S-TIE (and for symmetry purposes in node N-TIE as well but it serves no purpose there currently).

Obviously flooding reduction does NOT apply to self originated TIEs. Observe further that all policy-guided information consists of self-originated TIEs.

4.2.4. Policy-Guided Prefixes

In a fat tree, it can be sometimes desirable to guide traffic to particular destinations or keep specific flows to certain paths. In RIFT, this is done by using policy-guided prefixes with their associated communities. Each community is an abstract value whose meaning is determined by configuration. It is assumed that the fabric is under a single administrative control so that the meaning and intent of the communities is understood by all the nodes in the fabric. Any node can originate a policy-guided prefix.

Since RIFT uses distance vector concepts in a southbound direction, it is straightforward to add a policy-guided prefix to an S-TIE. For easier troubleshooting, the approach taken in RIFT is that a node's southbound policy-guided prefixes are sent in its S-TIE and the receiver does inbound filtering based on the associated communities (an egress policy is imaginable but would lead to different S-TIEs per neighbor possibly which is not considered in RIFT protocol procedures). A southbound policy-guided prefix can only use links in

the south direction. If an PGP S-TIE is received on an east-west or northbound link, it must be discarded by ingress filtering.

Conceptually, a southbound policy-guided prefix guides traffic from the leaves up to at most the north-most layer. It is also necessary to have northbound policy-guided prefixes to guide traffic from the north-most layer down to the appropriate leaves. Therefore, RIFT includes northbound policy-guided prefixes in its N PGP-TIE and the receiver does inbound filtering based on the associated communities. A northbound policy-guided prefix can only use links in the northern direction. If an N PGP TIE is received on an east-west or southbound link, it must be discarded by ingress filtering.

By separating southbound and northbound policy-guided prefixes and requiring that the cost associated with a PGP is strictly monotonically increasing at each hop, the path cannot loop. Because the costs are strictly increasing, it is not possible to have a loop between a northbound PGP and a southbound PGP. If east-west links were to be allowed, then looping could occur and issues such as counting to infinity would become an issue to be solved. If complete generality of path - such as including east-west links and using both north and south links in arbitrary sequence - then a Path Vector protocol or a similar solution must be considered.

If a node has received the same prefix, after ingress filtering, as a PGP in an S-TIE and in an N-TIE, then the node determines which policy-guided prefix to use based upon the advertised cost.

A policy-guided prefix is always preferred to a regular prefix, even if the policy-guided prefix has a larger cost. Section 8 provides normative indication of prefix preferences.

The set of policy-guided prefixes received in a TIE is subject to ingress filtering and then re-originated to be sent out in the receiver's appropriate TIE. Both the ingress filtering and the re-origination use the communities associated with the policy-guided prefixes to determine the correct behavior. The cost on re-advertisement MUST increase in a strictly monotonic fashion.

4.2.4.1. Ingress Filtering

When a node X receives a PGP S-TIE or a PGP N-TIE that is originated from a node Y which does not have an adjacency with X, all PGPs in such a TIE MUST be filtered. Similarly, if node Y is at the same layer as node X, then X MUST filter out PGPs in such S- and N-TIEs to prevent loops.

Next, policy can be applied to determine which policy-guided prefixes to accept. Since ingress filtering is chosen rather than egress filtering and per-neighbor PGPs, policy that applies to links is done at the receiver. Because the RIFT adjacency is between nodes and there may be parallel links between the two nodes, the policy-guided prefix is considered to start with the next-hop set that has all links to the originating node Y.

A policy-guided prefix has or is assigned the following attributes:

cost: This is initialized to the cost received

community_list: This is initialized to the list of the communities received.

next_hop_set: This is initialized to the set of links to the originating node Y.

4.2.4.2. Applying Policy

The specific action to apply based upon a community is deployment specific. Here are some examples of things that can be done with communities. The length of a community is a 64 bits number and it can be written as a single field M or as a multi-field ($S = M[0-31]$, $T = M[32-63]$) in these examples. For simplicity, the policy-guided prefix is referred to as P, the processing node as X and the originator as Y.

Prune Next-Hops: Community Required: For each next-hop in P.next_hop_set, if the next-hop does not have the community, prune that next-hop from P.next_hop_set.

Prune Next-Hops: Avoid Community: For each next-hop in P.next_hop_set, if the next-hop has the community, prune that next-hop from P.next_hop_set.

Drop if Community: If node X has community M, discard P.

Drop if not Community: If node X does not have the community M, discard P.

Prune to ifIndex T: For each next-hop in P.next_hop_set, if the next-hop's ifIndex is not the value T specified in the community (S,T), then prune that next-hop from P.next_hop_set.

Add Cost T: For each appearance of community S in P.community_list, if the node X has community S, then add T to P.cost.

Accumulate Min-BW T: Let bw be the sum of the bandwidth for P.next_hop_set. If that sum is less than T, then replace (S,T) with (S, bw).

Add Community T if Node matches S: If the node X has community S, then add community T to P.community_list.

4.2.4.3. Store Policy-Guided Prefix for Route Computation and Regeneration

Once a policy-guided prefix has completed ingress filtering and policy, it is almost ready to store and use. It is still necessary to adjust the cost of the prefix to account for the link from the computing node X to the originating neighbor node Y.

There are three different policies that can be used:

Minimum Equal-Cost: Find the lowest cost C next-hops in P.next_hop_set and prune to those. Add C to P.cost.

Minimum Unequal-Cost: Find the lowest cost C next-hop in P.next_hop_set. Add C to P.cost.

Maximum Unequal-Cost: Find the highest cost C next-hop in P.next_hop_set. Add C to P.cost.

The default policy is Minimum Unequal-Cost but well-known communities can be defined to get the other behaviors.

Regardless of the policy used, a node MUST store a PGP cost that is at least 1 greater than the PGP cost received. This enforces the strictly monotonically increasing condition that avoids loops.

Two databases of PGPs - from N-TIEs and from S-TIEs are stored. When a PGP is inserted into the appropriate database, the usual tie-breaking on cost is performed. Observe that the node retains all PGP TIEs due to normal flooding behavior and hence loss of the best prefix will lead to re-evaluation of TIEs present and re-advertisement of a new best PGP.

4.2.4.4. Re-origination

A node must re-originate policy-guided prefixes and retransmit them. The node has its database of southbound policy-guided prefixes to send in its S-TIE and its database of northbound policy-guided prefixes to send in its N-TIE.

Of course, a leaf does not need to re-originate southbound policy-guided prefixes.

4.2.4.5. Overlap with Disaggregated Prefixes

PGPs may overlap with prefixes introduced by automatic de-aggregation. The topic is under further discussion. The break in connectivity that leads to infeasibility of a PGP is mirrored in adjacency tear-down and according removal of such PGPs. Nevertheless, the underlying link-state flooding will be likely reacting significantly faster than a hop-by-hop redistribution and with that the preference for PGPs may cause intermittent black-holes.

4.2.5. Reachability Computation

A node has three sources of relevant information. A node knows the full topology south from the received N-TIEs. A node has the set of prefixes with associated distances and bandwidths from received S-TIEs. A node can also have a set of PGPs.

To compute reachability, a node runs conceptually a northbound and a southbound SPF. We call that N-SPF and S-SPF.

Since neither computation can "loop" (with due considerations given to PGPs), it is possible to compute non-equal-cost or even k-shortest paths [EPPSTEIN] and "saturate" the fabric to the extent desired.

4.2.5.1. Northbound SPF

N-SPF uses northbound and east-west adjacencies in North Node TIEs when progressing Dijkstra. Observe that this is really just a one hop variety since South Node TIEs are not re-flooded southbound beyond a single level (or east-west) and with that the computation cannot progress beyond adjacent nodes.

Default route found when crossing an E-W link is used IIF

1. the node itself does NOT have any northbound adjacencies AND
2. the adjacent node has one or more northbound adjacencies

This rule forms a "one-hop default route split-horizon" and prevents looping over default routes while allowing for "one-hop protection" of nodes that lost all northbound adjacencies.

Other south prefixes found when crossing E-W link MAY be used IIF

1. no north neighbors are advertising same or supersuming prefix AND

2. the node does not originate a supersuming prefix itself.

i.e. the E-W link can be used as the gateway of last resort for a specific prefix only. Using south prefixes across E-W link can be beneficial e.g. on automatic de-aggregation in pathological fabric partitioning scenarios.

A detailed example can be found in Section 5.4.

For N-SPF we are using the South Node TIEs to find according adjacencies to verify backlink connectivity. Just as in case of IS-IS or OSPF, two unidirectional links are associated together to confirm bidirectional connectivity.

4.2.5.2. Southbound SPF

S-SPF uses only the southbound adjacencies in the south node TIEs, i.e. progresses towards nodes at lower levels. Observe that E-W adjacencies are NEVER used in the computation. This enforces the requirement that a packet traversing in a southbound direction must never change its direction.

S-SPF uses northbound adjacencies in north node TIEs to verify backlink connectivity.

4.2.5.3. East-West Forwarding Within a Level

Ultimately, it should be observed that in presence of a "ring" of E-W links in a level neither SPF will provide a "ring protection" scheme since such a computation would have to deal necessarily with breaking of "loops" in generic Dijkstra sense; an application for which RIFT is not intended. It is outside the scope of this document how an underlay can be used to provide a full-mesh connectivity between nodes in the same layer that would allow for N-SPF to provide protection for a single node loosing all its northbound adjacencies (as long as any of the other nodes in the level are northbound connected).

Using south prefixes over horizontal links is optional and can protect against pathological fabric partitioning cases that leave only paths to destinations that would necessitate multiple changes of forwarding direction between north and south.

4.2.6. Attaching Prefixes

After the SPF is run, it is necessary to attach according prefixes. For S-SPF, prefixes from an N-TIE are attached to the originating node with that node's next-hop set and a distance equal to the

prefix's cost plus the node's minimized path distance. The RIFT route database, a set of (prefix, type=spf, path_distance, next-hop set), accumulates these results. Obviously, the prefix retains its type which is used to tie-break between the same prefix advertised with different types.

In case of N-SPF prefixes from each S-TIE need to also be added to the RIFT route database. The N-SPF is really just a stub so the computing node needs simply to determine, for each prefix in an S-TIE that originated from adjacent node, what next-hops to use to reach that node. Since there may be parallel links, the next-hops to use can be a set; presence of the computing node in the associated Node S-TIE is sufficient to verify that at least one link has bidirectional connectivity. The set of minimum cost next-hops from the computing node X to the originating adjacent node is determined.

Each prefix has its cost adjusted before being added into the RIFT route database. The cost of the prefix is set to the cost received plus the cost of the minimum cost next-hop to that neighbor. Then each prefix can be added into the RIFT route database with the next_hop_set; ties are broken based upon type first and then distance. RIFT route preferences are normalized by the according thrift model type.

An exemplary implementation for node X follows:

```

for each S-TIE
  if S-TIE.layer > X.layer
    next_hop_set = set of minimum cost links to the S-TIE.originator
    next_hop_cost = minimum cost link to S-TIE.originator
  end if
  for each prefix P in the S-TIE
    P.cost = P.cost + next_hop_cost
    if P not in route_database:
      add (P, type=DistVector, P.cost, next_hop_set) to route_database
    end if
    if (P in route_database) and
      (route_database[P].type is not PolicyGuided):
      if route_database[P].cost > P.cost:
        update route_database[P] with (P, DistVector, P.cost, next_hop_set)
      else if route_database[P].cost == P.cost
        update route_database[P] with (P, DistVector, P.cost,
          merge(next_hop_set, route_database[P].next_hop_set))
      else
        // Not preferred route so ignore
      end if
    end if
  end for
end for

```

Figure 3: Adding Routes from S-TIE Prefixes

4.2.7. Attaching Policy-Guided Prefixes

Each policy-guided prefix P has its cost and next_hop_set already stored in the associated database, as specified in Section 4.2.4.3; the cost stored for the PGP is already updated to considering the cost of the link to the advertising neighbor. By definition, a policy-guided prefix is preferred to a regular prefix.

```

for each policy-guided prefix P:
  if P not in route_database:
    add (P, type=PolicyGuided, P.cost, next_hop_set)
  end if
  if P in route_database :
    if (route_database[P].type is not PolicyGuided) or
      (route_database[P].cost > P.cost):
      update route_database[P] with (P, PolicyGuided, P.cost, next_hop_set
)
    else if route_database[P].cost == P.cost
      update route_database[P] with (P, PolicyGuided, P.cost,
        merge(next_hop_set, route_database[P].next_hop_set))
    else
      // Not preferred route so ignore
    end if
  end if
end for

```

Figure 4: Adding Routes from Policy-Guided Prefixes

4.2.8. Automatic Disaggregation on Link & Node Failures

Under normal circumstances, node's S-TIEs contain just the adjacencies, a default route and policy-guided prefixes. However, if a node detects that its default IP prefix covers one or more prefixes that are reachable through it but not through one or more other nodes at the same level, then it MUST explicitly advertise those prefixes in an S-TIE. Otherwise, some percentage of the northbound traffic for those prefixes would be sent to nodes without according reachability, causing it to be black-holed. Even when not black-holing, the resulting forwarding could 'backhaul' packets through the higher level spines, clearly an undesirable condition affecting the blocking probabilities of the fabric.

We refer to the process of advertising additional prefixes as 'de-aggregation' or 'dis-aggregation'.

A node determines the set of prefixes needing de-aggregation using the following steps:

1. A DAG computation in the southern direction is performed first, i.e. the N-TIEs are used to find all of prefixes it can reach and the set of next-hops in the lower level for each. Such a computation can be easily performed on a fat tree by e.g. setting all link costs in the southern direction to 1 and all northern directions to infinity. We term set of those prefixes $|R$, and for each prefix, r , in $|R$, we define its set of next-hops to

be $|H(r)$. Observe that policy-guided prefixes are NOT affected since their scope is controlled by configuration.

2. The node uses reflected S-TIEs to find all nodes at the same level in the same PoD and the set of southbound adjacencies for each. The set of nodes at the same level is termed $|N$ and for each node, n , in $|N$, we define its set of southbound adjacencies to be $|A(n)$.
3. For a given r , if the intersection of $|H(r)$ and $|A(n)$, for any n , is null then that prefix r must be explicitly advertised by the node in an S-TIE.
4. Identical set of de-aggregated prefixes is flooded on each of the node's southbound adjacencies. In accordance with the normal flooding rules for an S-TIE, a node at the lower level that receives this S-TIE will not propagate it south-bound. Neither is it necessary for the receiving node to reflect the disaggregated prefixes back over its adjacencies to nodes at the level from which it was received.

To summarize the above in simplest terms: if a node detects that its default route encompasses prefixes for which one of the other nodes in its level has no possible next-hops in the level below, it has to disaggregate it to prevent black-holing or suboptimal routing. Hence a node X needs to determine if it can reach a different set of south neighbors than other nodes at the same level, which are connected to it via at least one common south or east-west neighbor. If it can, then prefix disaggregation may be required. If it can't, then no prefix disaggregation is needed. An example of disaggregation is provided in Section 5.3.

A possible algorithm is described last:

1. Create `partial_neighbors = (empty)`, a set of neighbors with partial connectivity to the node X 's layer from X 's perspective. Each entry is a list of south neighbor of X and a list of nodes of X .layer that can't reach that neighbor.
2. A node X determines its set of southbound neighbors `X.south_neighbors`.
3. For each S-TIE originated from a node Y that X has which is at X .layer, if `Y.south_neighbors` is not the same as `X.south_neighbors` but the nodes share at least one southern neighbor, for each neighbor N in `X.south_neighbors` but not in `Y.south_neighbors`, add $(N, (Y))$ to `partial_neighbors` if N isn't there or add Y to the list for N .

4. If `partial_neighbors` is empty, then node X does not to disaggregate any prefixes. If node X is advertising disaggregated prefixes in its S-TIE, X SHOULD remove them and re-advertise its according S-TIEs.

A node X computes its SPF based upon the received N-TIEs. This results in a set of routes, each categorized by (`prefix`, `path_distance`, `next-hop-set`). Alternately, for clarity in the following procedure, these can be organized by `next-hop-set` as (`next-hops`), `{(prefix, path_distance)}`). If `partial_neighbors` isn't empty, then the following procedure describes how to identify prefixes to disaggregate.

```

disaggregated_prefixes = {empty }
nodes_same_layer = { empty }
for each S-TIE
  if (S-TIE.layer == X.layer and
      X shares at least one S-neighbor with X)
    add S-TIE.originator to nodes_same_layer
  end if
end for

for each next-hop-set NHS
  isolated_nodes = nodes_same_layer
  for each NH in NHS
    if NH in partial_neighbors
      isolated_nodes = intersection(isolated_nodes,
                                   partial_neighbors[NH].nodes)
    end if
  end for

  if isolated_nodes is not empty
    for each prefix using NHS
      add (prefix, distance) to disaggregated_prefixes
    end for
  end if
end for

copy disaggregated_prefixes to X's S-TIE
if X's S-TIE is different
  schedule S-TIE for flooding
end if

```

Figure 5: Computation to Disaggregate Prefixes

Each disaggregated prefix is sent with the accurate path_distance. This allows a node to send the same S-TIE to each south neighbor. The south neighbor which is connected to that prefix will thus have a shorter path.

Finally, to summarize the less obvious points partially omitted in the algorithms to keep them more tractable:

1. all neighbor relationships MUST perform backlink checks.
2. overload bits as introduced in Section 4.3.1 have to be respected during the computation.
3. all the lower level nodes are flooded the same disaggregated prefixes since we don't want to build an S-TIE per node and complicate things unnecessarily. The PoD containing the prefix will prefer southbound anyway.
4. disaggregated prefixes do NOT have to propagate to lower levels. With that the disturbance in terms of new flooding is contained to a single level experiencing failures only.
5. disaggregated prefix S-TIEs are not "reflected" by the lower layer, i.e. nodes within same level do NOT need to be aware which node computed the need for disaggregation.
6. The fabric is still supporting maximum load balancing properties while not trying to send traffic northbound unless necessary.

4.2.9. Optional Autoconfiguration

RIFT nodes can operate in an optional mode where the levels in the fabric are being elected fully automatically. Future version of this document will render more detailed specification of the necessary protocol extensions.

4.3. Further Mechanisms

4.3.1. Overload Bit

Overload Bit MUST be respected in all according reachability computations. A node with overload bit set SHOULD NOT advertise any reachability prefixes southbound except locally hosted ones.

The leaf node SHOULD set the 'overload' bit on its node TIEs, since if the spine nodes were to forward traffic not meant for the local node, the leaf node does not have the topology information to prevent a routing/forwarding loop.

4.3.2. Optimized Route Computation on Leafs

Since the leafs do see only "one hop away" they do not need to run a full SPF but can simply gather prefix candidates from their neighbors and build the according routing table.

A leaf will have no N-TIEs except its own and optionally from its east-west neighbors. A leaf will have S-TIEs from its neighbors.

Instead of creating a network graph from its N-TIEs and neighbor's S-TIEs and then running an SPF, a leaf node can simply compute the minimum cost and next_hop_set to each leaf neighbor by examining its local interfaces, determining bi-directionality from the associated N-TIE, and specifying the neighbor's next_hop_set set and cost from the minimum cost local interfaces to that neighbor.

Then a leaf attaches prefixes as in Section 4.2.6 as well as the policy-guided prefixes as in Section 4.2.7.

4.3.3. Key/Value Store

The protocol supports a southbound distribution of key-value pairs that can be used to e.g. distribute configuration information during topology bring-up. The KV TIEs (which are always S-TIEs) can arrive from multiple nodes and need tie-breaking per key uses the following rules

1. Only KV TIEs originated by a node to which the receiver has an adjacency are considered.
2. Within all valid KV S-TIEs containing the key, the value of the S-TIE with the highest level and within the same level highest originator ID is preferred.

Observe that if a node goes down, the node south of it loses adjacencies to it and with that the KVs will be disregarded and on tie-break changes new KV re-advertised to prevent stale information being used by nodes further south. KV information is not result of independent computation of every node but a diffused computation.

4.3.4. Interactions with BFD

RIFT MAY incorporate BFD [RFC5881] to react quickly to link failures. In such case following procedures are introduced:

After RIFT 3-way hello adjacency convergence a BFD session MAY be formed automatically between the RIFT endpoints without further configuration.

In case RIFT loses 3-way hello adjacency, the BFD session should be brought down until 3-way adjacency is formed again.

In case established BFD session goes Down after it was Up, RIFT adjacency should be re-initialized from scratch.

In case of parallel links between nodes each link may run its own independent BFD session.

In case RIFT changes link identifiers both the hello as well as the BFD sessions will be brought down and back up again.

4.3.5. Leaf to Leaf Procedures

RIFT can be optionally relaxed to allow leaf East-West adjacencies under additional set of rules. The leaf supporting those procedures MUST:

Only nodes supporting Leaf to Leaf Procedures CAN advertise LIEs on E-W links at level 0 and MUST in such a case advertise the according flag in node capabilities as "true".

The overload bit MUST be set on all leaf's node TIEs.

Only node's own north and south TIEs are flooded over E-W leaf adjacency.

E-W leaf adjacency is always used in both north as well as south computation.

Any advertised aggregate in leaf's south TIE MUST install a discard route.

This will allow the E-W leaf nodes to exchange traffic strictly for the prefixes advertised in each other's north prefix TIEs (since the southbound computation will find the reverse direction in the other node's TIE and install its north prefixes).

4.3.6. Other End-to-End Services

Losing full, flat topology information at every node will have an impact on some of the end-to-end network services. This is the price paid for minimal disturbance in case of failures and reduced flooding and memory requirements on nodes lower south in the level hierarchy.

4.3.7. Address Family and Multi Topology Considerations

Multi-Topology (MT)[RFC5120] and Multi-Instance (MI)[RFC6822] is used today in link-state routing protocols to support several domains on the same physical topology. RIFT supports this capability by carrying transport ports in the LIE protocol exchanges. Multiplexing of LIEs can be achieved by either choosing varying multicast addresses or ports on the same address.

BFD interactions in Section 4.3.4 are implementation dependent when multiple RIFT instances run on the same link.

4.3.8. Reachability of Internal Nodes in the Fabric

RIFT does not precondition that its nodes have reachable addresses albeit for operational purposes this is clearly desirable. Under normal operating conditions this can be easily achieved by e.g. injecting the node's loopback address into North Prefix TIEs.

Things get more interesting in case a node loses all its northbound adjacencies but is not at the top of the fabric. In such a case a node that detects that some other members at its level are advertising northbound adjacencies MAY inject its loopback address into southbound PGP TIE and become reachable "from the south" that way. Further, a solution may be implemented where based on e.g. a "well known" community such a southbound PGP is reflected at level 0 and advertised as northbound PGP again to allow for "reachability from the north" at the cost of additional flooding.

4.3.9. One-Hop Healing of Levels with East-West Links

Based on the rules defined in Section 4.2.5, Section 4.2.3.7 and given presence of E-W links, RIFT can provide a one-hop protection of nodes that lost all their northbound links or in other complex link set failure scenarios. Section 5.4 explains the resulting behavior based on one such example.

5. Examples

5.1. Normal Operation

This section describes RIFT deployment in the example topology without any node or link failures. We disregard flooding reduction for simplicity's sake.

As first step, the following bi-directional adjacencies will be created (and any other links that do not fulfill LIE rules in Section 4.2.2 disregarded):

1. Spine 21 (PoD 0) to Node 111, Node 112, Node 121, and Node 122
2. Spine 22 (PoD 0) to Node 111, Node 112, Node 121, and Node 122
3. Node 111 to Leaf 111, Leaf 112
4. Node 112 to Leaf 111, Leaf 112
5. Node 121 to Leaf 121, Leaf 122
6. Node 122 to Leaf 121, Leaf 122

Consequently, N-TIEs would be originated by Node 111 and Node 112 and each set would be sent to both Spine 21 and Spine 22. N-TIEs also would be originated by Leaf 111 (w/ Prefix 111) and Leaf 112 (w/ Prefix 112 and the multi-homed prefix) and each set would be sent to Node 111 and Node 112. Node 111 and Node 112 would then flood these N-TIEs to Spine 21 and Spine 22.

Similarly, N-TIEs would be originated by Node 121 and Node 122 and each set would be sent to both Spine 21 and Spine 22. N-TIEs also would be originated by Leaf 121 (w/ Prefix 121 and the multi-homed prefix) and Leaf 122 (w/ Prefix 122) and each set would be sent to Node 121 and Node 122. Node 121 and Node 122 would then flood these N-TIEs to Spine 21 and Spine 22.

At this point both Spine 21 and Spine 22, as well as any controller to which they are connected, would have the complete network topology. At the same time, Node 111/112/121/122 hold only the N-ties of level 0 of their respective PoD. Leafs hold only their own N-TIEs.

S-TIEs with adjacencies and a default IP prefix would then be originated by Spine 21 and Spine 22 and each would be flooded to Node 111, Node 112, Node 121, and Node 122. Node 111, Node 112, Node 121, and Node 122 would each send the S-TIE from Spine 21 to Spine 22 and the S-TIE from Spine 22 to Spine 21. (S-TIEs are reflected up to level from which they are received but they are NOT propagated southbound.)

An S Tie with a default IP prefix would be originated by Node 111 and Node 112 and each would be sent to Leaf 111 and Leaf 112. Leaf 111 and Leaf 112 would each send the S-TIE from Node 111 to Node 112 and the S-TIE from Node 112 to Node 111.

Similarly, an S Tie with a default IP prefix would be originated by Node 121 and Node 122 and each would be sent to Leaf 121 and Leaf 122. Leaf 121 and Leaf 122 would each send the S-TIE from Node 121

to Node 122 and the S-TIE from Node 122 to Node 121. At this point IP connectivity with maximum possible ECMP has been established between the leafs while constraining the amount of information held by each node to the minimum necessary for normal operation and dealing with failures.

5.2. Leaf Link Failure

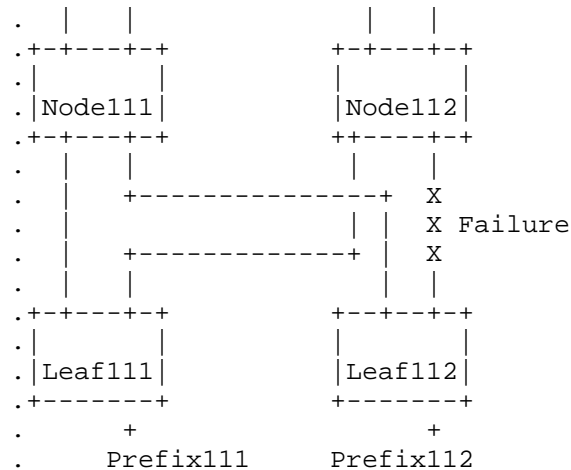


Figure 6: Single Leaf link failure

In case of a failing leaf link between node 112 and leaf 112 the link-state information will cause re-computation of the necessary SPF and the higher levels will stop forwarding towards prefix 112 through node 112. Only nodes 111 and 112, as well as both spines will see control traffic. Leaf 111 will receive a new S-TIE from node 112 and reflect back to node 111. Node 111 will de-aggregate prefix 111 and prefix 112 but we will not describe it further here since de-aggregation is emphasized in the next example. It is worth observing however in this example that if leaf 111 would keep on forwarding traffic towards prefix 112 using the advertised south-bound default of node 112 the traffic would end up on spine 21 and spine 22 and cross back into pod 1 using node 111. This is arguably not as bad as black-holing present in the next example but clearly undesirable. Fortunately, de-aggregation prevents this type of behavior except for a transitory period of time.

5.3. Partitioned Fabric

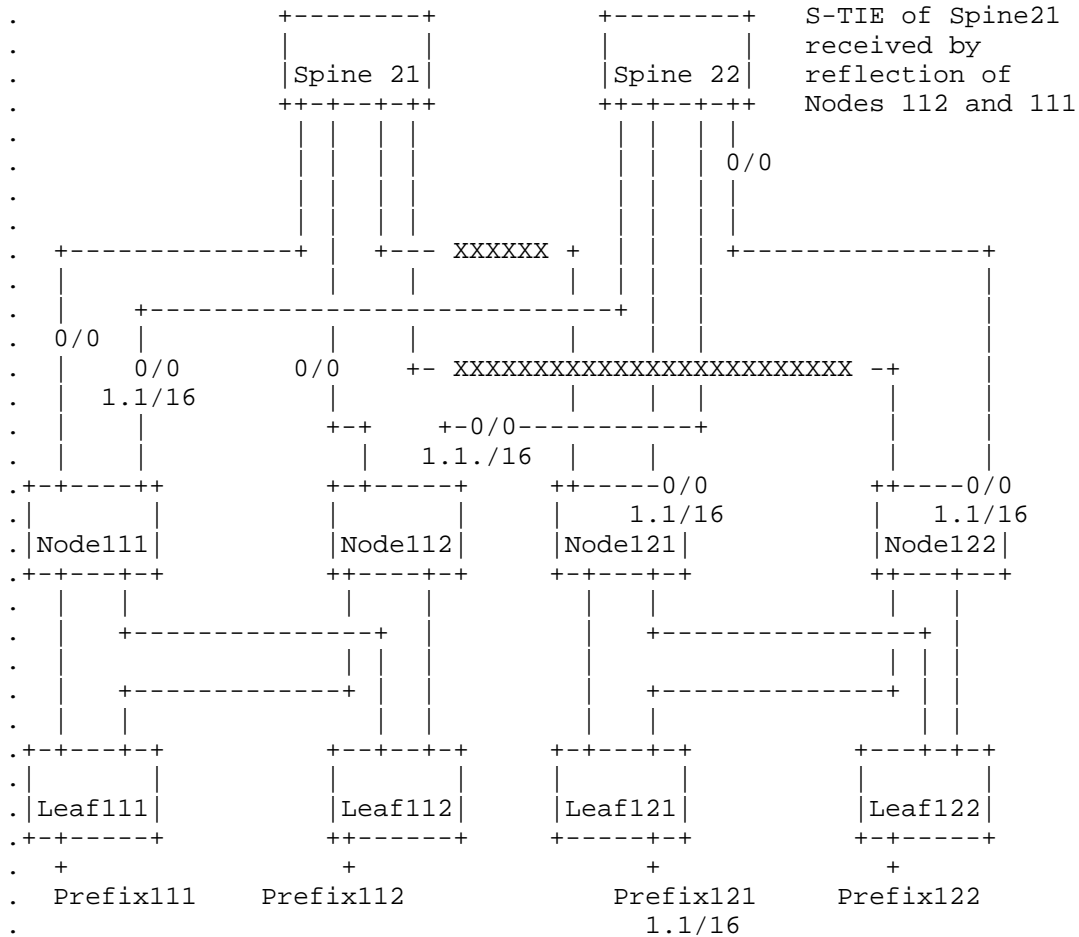


Figure 7: Fabric partition

Figure 7 shows the arguably most catastrophic but also the most interesting case. Spine 21 is completely severed from access to Prefix 121 (we use in the figure 1.1/16 as example) by double link failure. However unlikely, if left unresolved, forwarding from leaf 111 and leaf 112 to prefix 121 would suffer 50% black-holing based on pure default route advertisements by spine 21 and spine 22.

The mechanism used to resolve this scenario is hinging on the distribution of southbound representation by spine 21 that is reflected by node 111 and node 112 to spine 22. Spine 22, having

computed reachability to all prefixes in the network, advertises with the default route the ones that are reachable only via lower level neighbors that spine 21 does not show an adjacency to. That results in node 111 and node 112 obtaining a longest-prefix match to prefix 121 which leads through spine 22 and prevents black-holing through spine 21 still advertising the 0/0 aggregate only.

The prefix 121 advertised by spine 22 does not have to be propagated further towards leafs since they do no benefit from this information. Hence the amount of flooding is restricted to spine 21 reissuing its S-TIEs and reflection of those by node 111 and node 112. The resulting SPF in spine 22 issues a new prefix S-TIEs containing 1.1/16. None of the leafs become aware of the changes and the failure is constrained strictly to the level that became partitioned.

To finish with an example of the resulting sets computed using notation introduced in Section 4.2.8, spine 22 constructs the following sets:

|R = Prefix 111, Prefix 112, Prefix 121, Prefix 122

|H (for r=Prefix 111) = Node 111, Node 112

|H (for r=Prefix 112) = Node 111, Node 112

|H (for r=Prefix 121) = Node 121, Node 122

|H (for r=Prefix 122) = Node 121, Node 122

|A (for Spine 21) = Node 111, Node 112

With that and |H (for r=prefix 121) and |H (for r=prefix 122) being disjoint from |A (for spine 21), spine 22 will originate an S-TIE with prefix 121 and prefix 122, that is flooded to nodes 112, 121 and 122.

5.4. Northbound Partitioned Router and Optional East-West Links

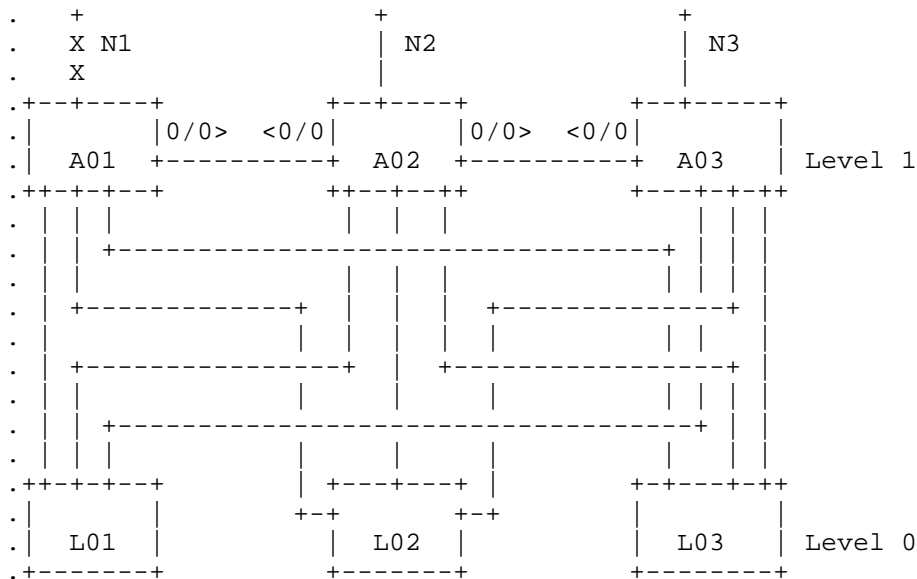


Figure 8: North Partitioned Router

Figure 8 shows a part of a fabric where level 1 is horizontally connected and A01 lost its only northbound adjacency. Based on N-SPF rules in Section 4.2.5.1 A01 will compute northbound reachability by using the link A01 to A02 (whereas A02 will NOT use this link during N-SPF). Hence A01 will still advertise the default towards level 0 and route unidirectionally using the horizontal link. Moreover, based on Section 4.3.8 it may advertise its loopback address as south PGP to remain reachable "from the south" for operational purposes. This is necessary since A02 will NOT route towards A01 using the E-W link (doing otherwise may form routing loops).

As further consideration, the moment A02 loses link N2 the situation evolves again. A01 will have no more northbound reachability while still seeing A03 advertising northbound adjacencies in its south node tie. With that it will stop advertising a default route due to Section 4.2.3.7. Moreover, A02 may now inject its loopback address as south PGP.

6. Implementation and Operation: Further Details

6.1. Considerations for Leaf-Only Implementation

Ideally RIFT can be stretched out to the lowest level in the IP fabric to integrate ToRs or even servers. Since those entities would run as leafs only, it is worth to observe that a leaf only version is significantly simpler to implement and requires much less resources:

1. Under normal conditions, the leaf needs to support a multipath default route only. In worst partitioning case it has to be capable of accommodating all the leaf routes in its own POD to prevent black-holing.
2. Leaf nodes hold only their own N-TIEs and S-TIEs of Level 1 nodes they are connected to; so overall few in numbers.
3. Leaf node does not have to support flooding reduction and de-aggregation.
4. Unless optional leaf-2-leaf procedures are desired default route origination, S-TIE origination is unnecessary.

6.2. Adaptations to Other Proposed Data Center Topologies

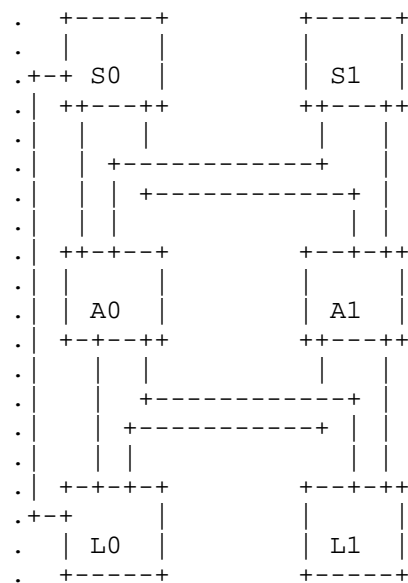


Figure 9: Level Shortcut

Strictly speaking, RIFT is not limited to Clos variations only. The protocol preconditions only a sense of 'compass rose direction' achieved by configuration of levels and other topologies are possible within this framework. So, conceptually, one could include leaf to leaf links and even shortcut between layers but certain requirements in Section 3 will not be met anymore. As an example, shortcutting levels illustrated in Figure 9 will lead either to suboptimal routing when L0 sends traffic to L1 (since using S0's default route will lead to the traffic being sent back to A0 or A1) or the leafs need each other's routes installed to understand that only A0 and A1 should be used to talk to each other.

Whether such modifications of topology constraints make sense is dependent on many technology variables and the exhausting treatment of the topic is definitely outside the scope of this document.

6.3. Originating Non-Default Route Southbound

Obviously, an implementation may choose to originate southbound instead of a strict default route (as described in Section 4.2.3.7) a shorter prefix P' but in such a scenario all addresses carried within the RIFT domain must be contained within P'.

7. Security Considerations

The protocol has provisions for nonces and can include authentication mechanisms in the future comparable to [RFC5709] and [RFC7987].

One can consider additionally attack vectors where a router may reboot many times while changing its system ID and pollute the network with many stale TIEs or TIEs are sent with very long lifetimes and not cleaned up when the routes vanishes. Those attack vectors are not unique to RIFT. Given large memory footprints available today those attacks should be relatively benign. Otherwise a node can implement a strategy of e.g. discarding contents of all TIEs of nodes that were not present in the SPF tree over a certain period of time. Since the protocol, like all modern link-state protocols, is self-stabilizing and will advertise the presence of such TIEs to its neighbors, they can be re-requested again if a computation finds that it sees an adjacency formed towards the system ID of the discarded TIEs.

8. Information Elements Schema

This section introduces the schema for information elements.

On schema changes that

1. change field numbers or
2. add new required fields or
3. change lists into sets, unions into structures or
4. change multiplicity of fields or
5. change datatypes of any field or
6. changes default value of any field

major version of the schema MUST increase. All other changes MUST increase minor version within the same major.

Thrift serializer/deserializer MUST not discard optional, unknown fields but preserve and serialize them again when re-flooding.

All signed integer as forced by Thrift support must be cast for internal purposes to equivalent unsigned values without discarding the signedness bit. An implementation SHOULD try to avoid using the signedness bit when generating values.

The schema is normative.

8.1. common.thrift

```
/**
 * Thrift file for common definitions in RIFT
 */

namespace * models

typedef i64      SystemIDType
typedef i32      IPv4Address
/** this has to be of length long enough to accommodate prefix */
typedef binary  IPv6Address
typedef i16      UDPPortType
typedef i32      TIENrType
typedef i32      MTUSizeType
typedef i32      SeqNrType
/** lifetime in seconds */
typedef i32      LifeTimeInSecType
typedef i16      LevelType
typedef i16      PodType
typedef i16      VersionType
typedef i32      MetricType
```

```
typedef string KeyIDType
/** node local, unique identification for a link (interface/tunnel
 * etc. Basically anything RIFT runs on). This is kept
 * at 32 bits so it aligns with BFD [RFC5880] discriminator size.
 */
typedef i32 LinkIDType
typedef string KeyNameType
typedef i8 PrefixLenType
/** timestamp in seconds since the epoch */
typedef i64 TimestampInSecsType
/** security nonce */
typedef i64 NonceType
/** adjacency holdtime */
typedef i16 HoldTimeInSecType

/** fixed leaf level that will not participate in election */
const LevelType leaf_level = 0
const LevelType default_level = 0
const PodType default_pod = 0
const LinkIDType undefined_linkid = 0
const MetricType default_distance = 1
/** any distance larger than this will be considered infinity */
const MetricType infinite_distance = 0x70000000
/** any element with 0 distance will be ignored,
 * missing metrics will be replaced with default_distance
 */
const MetricType invalid_distance = 0
const bool overload_default = false
const bool flood_reduction_default = true
const bool leaf_2_leaf_procedures_default = false
const HoldTimeInSecType default_holdtime = 3
/** 0 is illegal for SystemID */
const SystemIDType IllegalSystemID = 0

/** indicates whether the direction is northbound/east-west
 * or southbound */
enum TieDirectionType {
    Illegal = 0,
    South = 1,
    North = 2,
    DirectionMaxValue = 3,
}

enum AddressFamilyType {
    Illegal = 0,
    AddressFamilyMinValue = 1,
    IPv4 = 2,
    IPv6 = 3,
```

```
    AddressFamilyMaxValue = 4,
}

struct IPv4PrefixType {
    1: required IPv4Address    address;
    2: required PrefixLenType  prefixlen;
}

struct IPv6PrefixType {
    1: required IPv6Address    address;
    2: required PrefixLenType  prefixlen;
}

union IPAddressType {
    1: optional IPv4Address    ipv4address;
    2: optional IPv6Address    ipv6address;
}

union IPPrefixType {
    1: optional IPv4PrefixType  ipv4prefix;
    2: optional IPv6PrefixType  ipv6prefix;
}

enum TIETypeType {
    Illegal                = 0,
    TIETypeMinValue        = 1,
    /** first legal value */
    NodeTIEType            = 2,
    PrefixTIEType          = 3,
    PGPrefixTIEType        = 4,
    KeyValueTIEType        = 5,
    TIETypeMaxValue        = 6,
}

/** @note: route types which MUST be ordered on preference
 *  * PGP prefixes are most preferred attracting
 *  * traffic north (towards spine)
 *  * normal prefixes are attracting traffic south (towards leafs),
 *  * i.e. prefix in NORTH PREFIX TIE is preferred
 */
enum RouteType {
    Illegal                = 0,
    RouteTypeMinValue      = 1,
    /** First legal value. */
    /** Discard routes are most preferred */
    Discard                = 2,

    /** Local prefixes are directly attached prefixes on the
```

```
    * system such as e.g. interface routes.
    */
    LocalPrefix      = 3,
    /** advertised in S-TIEs */
    SouthPGPPrefix   = 4,
    /** advertised in N-TIEs */
    NorthPGPPrefix   = 5,
    /** advertised in N-TIEs */
    NorthPrefix      = 6,
    /** advertised in S-TIEs */
    SouthPrefix      = 7,
    RouteTypeMaxValue = 8
}
```

8.2. encoding.thrift

```
/**
    Thrift file for packet encodings for RIFT
*/

include "common.thrift"

namespace * models

/**
    Thrift file for packet encodings for RIFT
*/

include "common.thrift"

namespace rs models
namespace py encoding

/** represents protocol major version */
const i32 protocol_major_version = 3
/** represents protocol minor version */
const i32 protocol_minor_version = 0

/** common RIFT packet header */
struct PacketHeader {
    1: required common.VersionType major_version = protocol_major_version;
    2: required common.VersionType minor_version = protocol_minor_version;
    /** this is the node sending the packet, in case of LIE/TIRE/TIDE
        also the originator of it */
    3: required common.SystemIDType sender;
    /** level of the node sending the packet, required on anything but LIEs.
        * Lack of presence on LIEs indicates auto-election of level.
```

```

        */
        4: optional common.LevelType      level;
    }

    /** Community serves as community for PGP purposes */
    struct Community {
        1: required i32          top;
        2: required i32          bottom;
    }

    /** Neighbor structure */
    struct Neighbor {
        1: required common.SystemIDType    originator;
        2: required common.LinkIDType      remote_id;
    }

    /** Capabilities the node supports */
    struct NodeCapabilities {
        /** can this node participate in flood reduction,
         * only relevant at level > 0 */
        1: optional bool      flood_reduction = common.flood_reduction_default;
        /** does this node support leaf-2-leaf procedures,
         * only relevant in case node has no southbound
         * adjacencies, otherwise ignored */
        2: optional bool      leaf_2_leaf_procedures = common.leaf_2_leaf_procedure_default;
    }

    /** RIFT LIE packet

        @note this node's level is already included on the packet header */
    struct LIEPacket {
        /** optional node or adjacency name */
        1: optional string      name;
        /** local link ID */
        2: required common.LinkIDType    local_id;
        /** UDP port to which we can flood TIEs, same address
         * as the hello TX this hello has been received on
         */
        3: required common.UDPPortType    flood_port;
        /** layer 3 MTU */
        4: required common.MTUSizeType    link_mtu_size;
        /** this will reflect the neighbor once received */
        5: optional Neighbor              neighbor;
        6: optional common.PodType        pod = common.default_pod;
        /** optional nonce used for security computations */
        7: optional common.NonceType      nonce;
        /** optional node capabilities shown in the LIE. The capabilities
         * MUST match the capabilities shown in the Node TIEs, otherwise

```

```

        the behavior is unspecified. A node detecting the mismatch
        SHOULD generate according error.
    */
    8: optional NodeCapabilities      capabilities;
    /** required holdtime of the adjacency, i.e. how much time
        MUST expire without LIE for the adjacency to drop
    */
    9: required common.HoldTimeInSecType holdtime = common.default_holdtime;
}

/** LinkID pair describes one of parallel links between two nodes */
struct LinkIDPair {
    /** node-wide unique value for the local link */
    1: required common.LinkIDType      local_id;
    /** received remote link ID for this link */
    2: required common.LinkIDType      remote_id;
    /** more properties of the link can go in here */
}

/** ID of a TIE

    @note: TIEID space is a total order achieved by comparing the elements
           in sequence defined and comparing each value as an unsigned integer
           of according length
*/
struct TIEID {
    /** indicates direction of the TIE */
    1: required common.TieDirectionType direction;
    /** indicates originator of the TIE */
    2: required common.SystemIDType      originator;
    3: required common.TIETimeTypeType   tietype;
    4: required common.TIENrType         tie_nr;
}

/** Header of a TIE */
struct TIEHeader {
    2: required TIEID                      tieid;
    3: required common.SeqNrType           seq_nr;
    /** lifetime expires down to 0 just like in ISIS */
    4: required common.LifeTimeInSecType   lifetime;
}

/** A sorted TIDE packet, if unsorted, behavior is undefined */
struct TIDEPacket {
    /** all 00s marks starts */
    1: required TIEID                      start_range;
    /** all FFs mark end */
    2: required TIEID                      end_range;
}

```



```

    /** _sorted_ list of headers */
    3: required list<TIEHeader> headers;
}

/** A TIRE packet */
struct TIREPacket {
    1: required set<TIEHeader> headers;
}

/** Neighbor of a node */
struct NodeNeighborsTIEElement {
    2: required common.LevelType      level;
    /** Cost to neighbor.

        @note: All parallel links to same node
        incur same cost, in case the neighbor has multiple
        parallel links at different cost, the largest distance
        (highest numerical value) MUST be advertised */
    3: optional common.MetricType      cost = common.default_distance;
    /** can carry description of multiple parallel links in a TIE */
    4: optional set<LinkIDPair>        link_ids;
}

/** Flags the node sets */
struct NodeFlags {
    /** node is in overload, do not transit traffic through it */
    1: optional bool                  overload = common.overload_default;
}

/** Description of a node.

    It may occur multiple times in different TIEs but if either
    * capabilities values do not match or
    * flags values do not match
    * neighbors repeat with different values
    the behavior is undefined and a warning
    SHOULD be generated.

    @note: observe that absence of fields implies defined defaults
*/
struct NodeTIEElement {
    1: required common.LevelType      level;
    2: optional NodeCapabilities      capabilities;
    3: optional NodeFlags              flags;
    /** optional node name for easier operations */
    4: optional string                 name;
    /** if neighbor systemID repeats in other node TIEs of same node
        the behavior is undefined */

```

```
    5: required map<common.SystemIDType,NodeNeighborsTIEElement> neighbors;
}

/** multiple prefixes */
struct PrefixTIEElement {
    /** prefixes with the associated cost.
        if the same prefix repeats in multiple TIEs of same node
        or with different metrics, behavior is unspecified */
    1: required map<common.IPPrefixType,common.MetricType> prefixes;
}

/** keys with their values */
struct KeyValueTIEElement {
    /** if the same key repeats in multiple TIEs of same node
        or with different values, behavior is unspecified */
    1: required map<common.KeyIDType,string> keyvalues;
}

/** single element in a TIE. enum common.TIETypeType
    in TIEID indicates which elements MUST be present
    in the TIEElement. In case of mismatch the unexpected
    elements MUST be ignored.
    */
union TIEElement {
    /** in case of enum common.TIETypeType.NodeTIEType */
    1: optional NodeTIEElement node;
    /** in case of enum common.TIETypeType.PrefixTIEType */
    2: optional PrefixTIEElement prefixes;
    3: optional KeyValueTIEElement keyvalues;
    /** @todo: policy guided prefixes */
}

/** @todo: flood header separately in UDP to allow caching to TIEs
    while changing lifetime?
    */
struct TIEPacket {
    1: required TIEHeader header;
    2: required TIEElement element;
}

union PacketContent {
    1: optional LIEPacket lie;
    2: optional TIDEPacket tide;
    3: optional TIREPacket tire;
    4: optional TIEPacket tie;
}

/** protocol packet structure */
```

```
struct ProtocolPacket {  
    1: required PacketHeader  header;  
    2: required PacketContent content;  
}
```

9. IANA Considerations

This specification will request at an opportune time multiple registry points to exchange protocol packets in a standardized way, amongst them multicast address assignments and standard port numbers. The schema itself defines many values and codepoints which can be considered registries themselves.

10. Security Considerations

Security mechanisms will be addressed in upcoming versions of this specification.

11. Acknowledgments

Many thanks to Naiming Shen for some of the early discussions around the topic of using IGP for routing in topologies related to Clos. Adrian Farrel, Joel Halpern and Jeffrey Zhang provided thoughtful comments that improved the readability of the document and found good amount of corners where the light failed to shine. Kris Price was first to mention single router, single arm default considerations. Jeff Tantsura helped out with some initial thoughts on BFD interactions while Jeff Haas corrected several misconceptions about BFD's finer points. Artur Makutunowicz pointed out many possible improvements and acted as sounding board in regard to modern protocol implementation techniques RIFT is exploring.

12. References

12.1. Normative References

- [ISO10589] ISO "International Organization for Standardization", "Intermediate system to Intermediate system intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode Network Service (ISO 8473), ISO/IEC 10589:2002, Second Edition.", Nov 2002.
- [RFC1142] Oran, D., Ed., "OSI IS-IS Intra-domain Routing Protocol", RFC 1142, DOI 10.17487/RFC1142, February 1990, <<https://www.rfc-editor.org/info/rfc1142>>.

- [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/info/rfc2119>>.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<https://www.rfc-editor.org/info/rfc2328>>.
- [RFC2365] Meyer, D., "Administratively Scoped IP Multicast", BCP 23, RFC 2365, DOI 10.17487/RFC2365, July 1998, <<https://www.rfc-editor.org/info/rfc2365>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.
- [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <<https://www.rfc-editor.org/info/rfc4291>>.
- [RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <<https://www.rfc-editor.org/info/rfc4655>>.
- [RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <<https://www.rfc-editor.org/info/rfc5120>>.
- [RFC5303] Katz, D., Saluja, R., and D. Eastlake 3rd, "Three-Way Handshake for IS-IS Point-to-Point Adjacencies", RFC 5303, DOI 10.17487/RFC5303, October 2008, <<https://www.rfc-editor.org/info/rfc5303>>.
- [RFC5709] Bhatia, M., Manral, V., Fanto, M., White, R., Barnes, M., Li, T., and R. Atkinson, "OSPFv2 HMAC-SHA Cryptographic Authentication", RFC 5709, DOI 10.17487/RFC5709, October 2009, <<https://www.rfc-editor.org/info/rfc5709>>.
- [RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, DOI 10.17487/RFC5881, June 2010, <<https://www.rfc-editor.org/info/rfc5881>>.

- [RFC6234] Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, DOI 10.17487/RFC6234, May 2011, <<https://www.rfc-editor.org/info/rfc6234>>.
- [RFC6822] Previdi, S., Ed., Ginsberg, L., Shand, M., Roy, A., and D. Ward, "IS-IS Multi-Instance", RFC 6822, DOI 10.17487/RFC6822, December 2012, <<https://www.rfc-editor.org/info/rfc6822>>.
- [RFC7855] Previdi, S., Ed., Filsfils, C., Ed., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "Source Packet Routing in Networking (SPRING) Problem Statement and Requirements", RFC 7855, DOI 10.17487/RFC7855, May 2016, <<https://www.rfc-editor.org/info/rfc7855>>.
- [RFC7938] Lapukhov, P., Premji, A., and J. Mitchell, Ed., "Use of BGP for Routing in Large-Scale Data Centers", RFC 7938, DOI 10.17487/RFC7938, August 2016, <<https://www.rfc-editor.org/info/rfc7938>>.
- [RFC7987] Ginsberg, L., Wells, P., Decraene, B., Przygienda, T., and H. Gredler, "IS-IS Minimum Remaining Lifetime", RFC 7987, DOI 10.17487/RFC7987, October 2016, <<https://www.rfc-editor.org/info/rfc7987>>.

12.2. Informative References

- [CLOS] Yuan, X., "On Nonblocking Folded-Clos Networks in Computer Communication Environments", IEEE International Parallel & Distributed Processing Symposium, 2011.
- [DIJKSTRA] Dijkstra, E., "A Note on Two Problems in Connexion with Graphs", Journal Numer. Math. , 1959.
- [DYNAMO] De Candia et al., G., "Dynamo: amazon's highly available key-value store", ACM SIGOPS symposium on Operating systems principles (SOSP '07), 2007.
- [EPPSTEIN] Eppstein, D., "Finding the k-Shortest Paths", 1997.
- [FATTREE] Leiserson, C., "Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing", 1985.

[MAKSIC2013]

Maksic et al., N., "Improving Utilization of Data Center Networks", IEEE Communications Magazine, Nov 2013.

[QUIC]

Iyengar et al., J., "QUIC: A UDP-Based Multiplexed and Secure Transport", 2016.

[VAHDAT08]

Al-Fares, M., Loukissas, A., and A. Vahdat, "A Scalable, Commodity Data Center Network Architecture", SIGCOMM , 2008.

Authors' Addresses

Tony Przygienda
Juniper Networks
1194 N. Mathilda Ave
Sunnyvale, CA 94089
US

Email: prz@juniper.net

Alankar Sharma
Comcast
1800 Bishops Gate Blvd
Mount Laurel, NJ 08054
US

Email: Alankar_Sharma@comcast.com

John Drake
Juniper Networks
1194 N. Mathilda Ave
Sunnyvale, CA 94089
US

Email: jdrake@juniper.net

Alia Atlas
Juniper Networks
10 Technology Park Drive
Westford, MA 01886
US

Email: akatlas@juniper.net

Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 19, 2019

N. Shen
L. Ginsberg
Cisco Systems
S. Thyamagundalu
October 16, 2018

IS-IS Routing for Spine-Leaf Topology
draft-shen-isis-spine-leaf-ext-07

Abstract

This document describes a mechanism for routers and switches in a Spine-Leaf type topology to have non-reciprocal Intermediate System to Intermediate System (IS-IS) routing relationships between the leafs and spines. The leaf nodes do not need to have the topology information of other nodes and exact prefixes in the network. This extension also has application in the Internet of Things (IoT).

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 19, 2019.

Copyright Notice

Copyright (c) 2018 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 (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Requirements Language	3
2. Motivations	3
3. Spine-Leaf (SL) Extension	4
3.1. Topology Examples	4
3.2. Applicability Statement	5
3.3. Spine-Leaf TLV	6
3.3.1. Spine-Leaf Sub-TLVs	7
3.3.1.1. Leaf-Set Sub-TLV	7
3.3.1.2. Info-Req Sub-TLV	8
3.3.2. Advertising IPv4/IPv6 Reachability	8
3.3.3. Advertising Connection to RF-Leaf Node	8
3.4. Mechanism	8
3.4.1. Pure CLOS Topology	10
3.5. Implementation and Operation	11
3.5.1. CSNP PDU	11
3.5.2. Overload Bit	11
3.5.3. Spine Node Hostname	11
3.5.4. IS-IS Reverse Metric	11
3.5.5. Spine-Leaf Traffic Engineering	12
3.5.6. Other End-to-End Services	12
3.5.7. Address Family and Topology	12
3.5.8. Migration	13
4. IANA Considerations	13
5. Security Considerations	14
6. Acknowledgments	14
7. Document Change Log	14
7.1. Changes to draft-shen-isis-spine-leaf-ext-05.txt	14
7.2. Changes to draft-shen-isis-spine-leaf-ext-04.txt	14
7.3. Changes to draft-shen-isis-spine-leaf-ext-03.txt	14
7.4. Changes to draft-shen-isis-spine-leaf-ext-02.txt	14
7.5. Changes to draft-shen-isis-spine-leaf-ext-01.txt	15
7.6. Changes to draft-shen-isis-spine-leaf-ext-00.txt	15
8. References	15
8.1. Normative References	15
8.2. Informative References	16
Authors' Addresses	17

1. Introduction

The IS-IS routing protocol defined by [ISO10589] has been widely deployed in provider networks, data centers and enterprise campus environments. In the data center and enterprise switching networks, a Spine-Leaf topology is commonly used. This document describes a mechanism where IS-IS routing can be optimized for a Spine-Leaf topology.

In a Spine-Leaf topology, normally a leaf node connects to a number of spine nodes. Data traffic going from one leaf node to another leaf node needs to pass through one of the spine nodes. Also, the decision to choose one of the spine nodes is usually part of equal cost multi-path (ECMP) load sharing. The spine nodes can be considered as gateway devices to reach destinations on other leaf nodes. In this type of topology, the spine nodes have to know the topology and routing information of the entire network, but the leaf nodes only need to know how to reach the gateway devices to which are the spine nodes they are uplinked.

This document describes the IS-IS Spine-Leaf extension that allows the spine nodes to have all the topology and routing information, while keeping the leaf nodes free of topology information other than the default gateway routing information. The leaf nodes do not even need to run a Shortest Path First (SPF) calculation since they have no topology information.

1.1. 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].

2. Motivations

- o The leaf nodes in a Spine-Leaf topology do not require complete topology and routing information of the entire domain since their forwarding decision is to use ECMP with spine nodes as default gateways
- o The spine nodes in a Spine-Leaf topology are richly connected to leaf nodes, which introduces significant flooding duplication if they flood all Link State PDUs (LSPs) to all the leaf nodes. It saves both spine and leaf nodes' CPU and link bandwidth resources if flooding is blocked to leaf nodes. For small Top of the Rack (ToR) leaf switches in data centers, it is meaningful to prevent full topology routing information and massive database flooding through those devices.

- o When a spine node advertises a topology change, every leaf node connected to it will flood the update to all the other spine nodes, and those spine nodes will further flood them to all the leaf nodes, causing a $O(n^2)$ flooding storm which is largely redundant.
- o Similar to some of the overlay technologies which are popular in data centers, the edge devices (leaf nodes) may not need to contain all the routing and forwarding information on the device's control and forwarding planes. "Conversational Learning" can be utilized to get the specific routing and forwarding information in the case of pure CLOS topology and in the events of link and node down.
- o Small devices and appliances of Internet of Things (IoT) can be considered as leafs in the routing topology sense. They have CPU and memory constrains in design, and those IoT devices do not have to know the exact network topology and prefixes as long as there are ways to reach the cloud servers or other devices.

3. Spine-Leaf (SL) Extension

3.1. Topology Examples

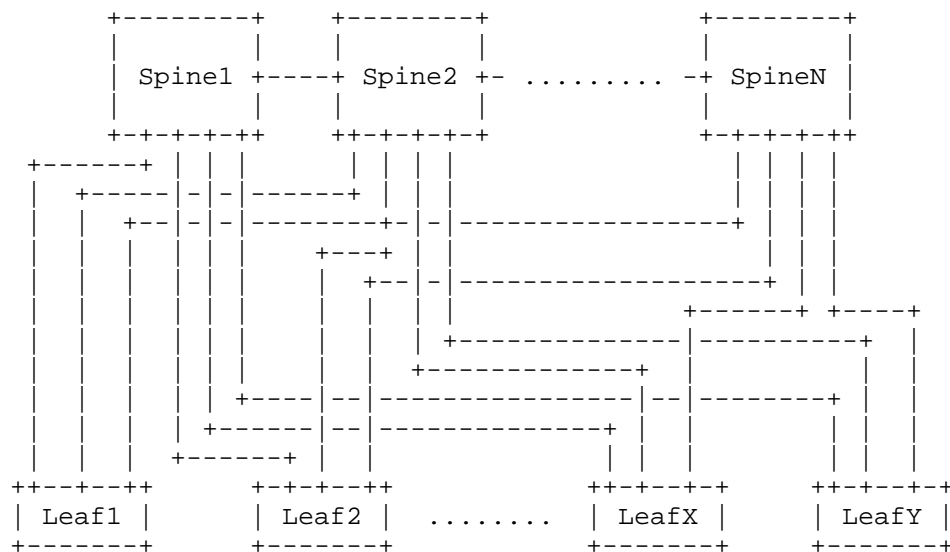


Figure 1: A Spine-Leaf Topology

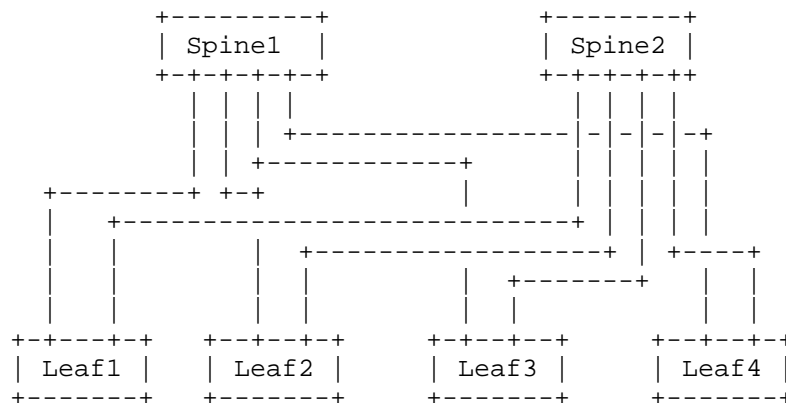


Figure 2: A CLOS Topology

3.2. Applicability Statement

This extension assumes the network is a Spine-Leaf topology, and it should not be applied in an arbitrary network setup. The spine nodes can be viewed as the aggregation layer of the network, and the leaf nodes as the access layer of the network. The leaf nodes use a load sharing algorithm with spine nodes as nexthops in routing and forwarding.

This extension works when the spine nodes are inter-connected, and it works with a pure CLOS or Fat Tree topology based network where the spines are NOT horizontally interconnected.

Although the example diagram in Figure 1 shows a fully meshed Spine-Leaf topology, this extension also works in the case where they are partially meshed. For instance, leaf1 through leaf10 may be fully meshed with spine1 through spine5 while leaf11 through leaf20 is fully meshed with spine4 through spine8, and all the spines are inter-connected in a redundant fashion.

This extension can also work in multi-level spine-leaf topology. The lower level spine node can be a 'leaf' node to the upper level spine node. A spine-leaf 'Tier' can be exchanged with IS-IS hello packets to allow tier X to be connected with tier X+1 using this extension. Normally tier-0 will be the TOR routers and switches if provisioned.

This extension also works with normal IS-IS routing in a topology with more than two layers of spine and leaf. For instance, in example diagrams Figure 1 and Figure 2, there can be another Core layer of routers/switches on top of the aggregation layer. From an IS-IS routing point of view, the Core nodes are not affected by this

extension and will have the complete topology and routing information just like the spine nodes. To make the network even more scalable, the Core layer can operate as a level-2 IS-IS sub-domain while the Spine and Leaf layers operate as stays at the level-1 IS-IS domain.

This extension assumes the link between the spine and leaf nodes are point-to-point, or point-to-point over LAN [RFC5309]. The links connecting among the spine nodes or the links between the leaf nodes can be any type.

3.3. Spine-Leaf TLV

This extension introduces a new TLV, the Spine-Leaf TLV, which may be advertised in IS-IS Hello (IIH) PDUs, LSPs, or in Circuit Scoped Link State PDUs (CS-LSP) [RFC7356]. It is used by both spine and leaf nodes in this Spine-Leaf mechanism.

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-----+-----+-----+-----+-----+-----+-----+-----+
    |          Type          |      Length      |          SL Flag          |
    +-----+-----+-----+-----+-----+-----+-----+-----+
    |          .. Optional Sub-TLVs          |
    +-----+-----+-----+-----+-----+-----+

```

The fields of this TLV are defined as follows:

Type: 1 octet Suggested value 150 (to be assigned by IANA)

Length: 1 octet (2 + length of sub-TLVs).

SL Flags: 16 bits

```

    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
    +-----+-----+-----+-----+-----+-----+
    | Tier |      Reserved      | T | R | L |
    +-----+-----+-----+-----+-----+-----+

```

Tier: A value from 0 to 15. It represents the spine-leaf tier level. The value 15 is reserved to indicate the tier level is unknown. This value is only valid when the 'T' bit (see below) is set. If the 'T' bit is clear, this value MUST be set to zero on transmission, and it MUST be ignored on receipt.

L bit (0x01): Only leaf node sets this bit. If the L bit is set in the SL flag, the node indicates it is in 'Leaf-Mode'.

R bit (0x02): Only Spine node sets this bit. If the R bit is set, the node indicates to the leaf neighbor that it can be used as the default route gateway.

T bit (0x04): If set, the value in the "Tier" field (see above) is valid.

Optional Sub-TLV: Not defined in this document, for future extension

sub-TLVs MAY be included when the TLV is in a CS-LSP.
sub-TLVs MUST NOT be included when the TLV is in an IIH

3.3.1. Spine-Leaf Sub-TLVs

If the data center topology is a pure CLOS or Fat Tree, there are no link connections among the spine nodes. If we also assume there is not another Core layer on top of the aggregation layer, then the traffic from one leaf node to another may have a problem if there is a link outage between a spine node and a leaf node. For instance, in the diagram of Figure 2, if Leaf1 sends data traffic to Leaf3 through Spine1 node, and the Spine1-Leaf3 link is down, the data traffic will be dropped on the Spine1 node.

To address this issue spine and leaf nodes may send/request specific reachability information via the sub-TLVs defined below.

Two Spine-Leaf sub-TLVs are defined. The Leaf-Set sub-TLV and the Info-Req sub-TLV.

3.3.1.1. Leaf-Set Sub-TLV

This sub-TLV is used by spine nodes to optionally advertise Leaf neighbors to other Leaf nodes. The fields of this sub-TLV are defined as follows:

Type: 1 octet Suggested value 1 (to be assigned by IANA)

Length: 1 octet MUST be a multiple of 6 octets.

Leaf-Set: A list of IS-IS System-ID of the leaf node neighbors of this spine node.

3.3.1.2. Info-Req Sub-TLV

This sub-TLV is used by leaf nodes to request the advertisement of more specific prefix information from a selected spine node. The list of leaf nodes in this sub-TLV reflects the current set of leaf-nodes for which not all spine node neighbors have indicated the presence of connectivity in the Leaf-Set sub-TLV (See Section 3.3.1.1). The fields of this sub-TLV are defined as follows:

Type: 1 octet Suggested value 2 (to be assigned by IANA)

Length: 1 octet. It MUST be a multiple of 6 octets.

Info-Req: List of IS-IS System-IDs of leaf nodes for which connectivity information is being requested.

3.3.2. Advertising IPv4/IPv6 Reachability

In cases where connectivity between a leaf node and a spine node is down, the leaf node MAY request reachability information from a spine node as described in Section 3.3.1.2. The spine node utilizes TLVs 135 [RFC5305] and TLVs 236 [RFC5308] to advertise this information. These TLVs MAY be included either in IIHs or CS-LSPs [RFC7356] sent from the spine to the requesting leaf node. Sending such information in IIHs has limited scale - all reachability information MUST fit within a single IIH. It is therefore recommended that CS-LSPs be used.

3.3.3. Advertising Connection to RF-Leaf Node

For links between Spine and Leaf Nodes on which the Spine Node has set the R-bit and the Leaf node has set the L-bit in their respective Spine-Leaf TLVs, spine nodes may advertise the link with a bit in the "link-attribute" sub-TLV [RFC5029] to express this link is not used for LSP flooding. This information can be used by nodes computing a flooding topology e.g., [DYNAMIC-FLOODING], to exclude the RF-Leaf nodes from the computed flooding topology.

3.4. Mechanism

Leaf nodes in a spine-leaf application using this extension are provisioned with two attributes:

1) Tier level of 0. This indicates the node is a Leaf Node. The value 0 is advertised in the Tier field of Spine-Leaf TLV defined above.

2) Flooding reduction enabled/disabled. If flooding reduction is enabled the L-bit is set to one in the Spine-Leaf TLV defined above

A spine node does not need explicit configuration. Spine nodes can dynamically discover their tier level by computing the number of hops to a leaf node. Until a spine node determines its tier level it MUST advertise level 15 (unknown tier level) in the Spine-Leaf TLV defined above. Each tier level can also be statically provisioned on the node.

When a spine node receives an IIH which includes the Spine-Leaf TLV with Tier level 0 and 'L' bit set, it labels the point-to-point interface and adjacency to be a 'Reduced Flooding Leaf-Peer (RF-Leaf)'. IIHs sent by a spine node on a link to an RF-Leaf include the Spine-Leaf TLV with the 'R' bit set in the flags field. The 'R' bit indicates to the RF-Leaf neighbor that the spine node can be used as a default routing nexthop.

There is no change to the IS-IS adjacency bring-up mechanism for Spine-Leaf peers.

A spine node blocks LSP flooding to RF-Leaf adjacencies, except for the LSP PDUs in which the IS-IS System-ID matches the System-ID of the RF-Leaf neighbor. This exception is needed since when the leaf node reboots, the spine node needs to forward to the leaf node non-purged LSPs from the RF-Leaf's previous incarnation.

Leaf nodes will perform IS-IS LSP flooding as normal over all of its IS-IS adjacencies, but in the case of RF-Leafs only self-originated LSPs will exist in its LSP database.

Spine nodes will receive all the LSP PDUs in the network, including all the spine nodes and leaf nodes. It will perform Shortest Path First (SPF) as a normal IS-IS node does. There is no change to the route calculation and forwarding on the spine nodes.

The LSPs of a node only floods north bound towards the upper layer spine nodes. The default route is generated with loadsharing also towards the upper layer spine nodes.

RF-Leaf nodes do not have any LSP in the network except for its own. Therefore there is no need to perform SPF calculation on the RF-Leaf node. It only needs to download the default route with the nexthops of those Spine Neighbors which have the 'R' bit set in the Spine-Leaf TLV in IIH PDUs. IS-IS can perform equal cost or unequal cost load sharing while using the spine nodes as nexthops. The aggregated metric of the outbound interface and the 'Reverse Metric' [REVERSE-METRIC] can be used for this purpose.

3.4.1. Pure CLOS Topology

In a data center where the topology is pure CLOS or Fat Tree, there is no interconnection among the spine nodes, and there is not another Core layer above the aggregation layer with reachability to the leaf nodes. When flooding reduction to RF-Leafs is in use, if the link between a spine and a leaf goes down, there is then a possibility of black holing the data traffic in the network.

As in the diagram Figure 2, if the link Spine1-Leaf3 goes down, there needs to be a way for Leaf1, Leaf2 and Leaf4 to avoid the Spine1 if the destination of data traffic is to Leaf3 node.

In the above example, the Spine1 and Spine2 are provisioned to advertise the Leaf-Set sub-TLV of the Spine-Leaf TLV. Originally both Spines will advertise Leaf1 through Leaf4 as their Leaf-Set. When the Spine1-Leaf3 link is down, Spine1 will only have Leaf1, Leaf2 and Leaf4 in its Leaf-Set. This allows the other leaf nodes to know that Spine1 has lost connectivity to the leaf node of Leaf3.

Each RF-Leaf node can select another spine node to request for some prefix information associated with the lost leaf node. In this diagram of Figure 2, there are only two spine nodes (Spine-Leaf topology can have more than two spine nodes in general). Each RF-Leaf node can independently select a spine node for the leaf information. The RF-Leaf nodes will include the Info-Req sub-TLV in the Spine-Leaf TLV in hellos sent to the selected spine node, Spine2 in this case.

The spine node, upon receiving the request from one or more leaf nodes, will find the IPv6/IPv4 prefixes advertised by the leaf nodes listed in the Info-Req sub-TLV. The spine node will use the mechanism defined in Section 3.3.2 to advertise these prefixes to the RF-Leaf node. For instance, it will include the IPv4 loopback prefix of leaf3 based on the policy configured or administrative tag attached to the prefixes. When the leaf nodes receive the more specific prefixes, they will install the advertised prefixes towards the other spine nodes (Spine2 in this example).

For instance in the data center overlay scenario, when any IP destination or MAC destination uses the leaf3's loopback as the tunnel nexthop, the overlay tunnel from leaf nodes will only select Spine2 as the gateway to reach leaf3 as long as the Spine1-Leaf3 link is still down.

In cases where multiple links or nodes fail at the same time, the RF-leaf node may need to send the Info-Req to multiple upper layer spine

nodes in order to obtain reachability information for all the partially connected nodes.

This negative routing is more useful between tier 0 and tier 1 spine-leaf levels in a multi-level spine-leaf topology when the reduced flooding extension is in use. Nodes in tiers 1 or greater may have much richer topology information and alternative paths.

3.5. Implementation and Operation

3.5.1. CSNP PDU

In Spine-Leaf extension, Complete Sequence Number PDU (CSNP) does not need to be transmitted over the Spine-Leaf link to an RF-Leaf. Some IS-IS implementations send periodic CSNPs after the initial adjacency bring-up over a point-to-point interface. There is no need for this optimization here since the RF-Leaf does not need to receive any other LSPs from the network, and the only LSPs transmitted across the Spine-Leaf link is the leaf node LSP.

Also in the graceful restart case[RFC5306], for the same reason, there is no need to send the CSNPs over the Spine-Leaf interface to an RF-Leaf. Spine nodes only need to set the SRMflag on the LSPs belonging to the RF-Leaf.

3.5.2. Overload Bit

The leaf node SHOULD set the 'overload' bit on its LSP PDU, since if the spine nodes were to forward traffic not meant for the local node, the leaf node does not have the topology information to prevent a routing/forwarding loop.

3.5.3. Spine Node Hostname

This extension creates a non-reciprocal relationship between the spine node and leaf node. The spine node will receive leaf's LSP and will know the leaf's hostname, but the leaf does not have spine's LSP. This extension allows the Dynamic Hostname TLV [RFC5301] to be optionally included in spine's IIH PDU when sending to a 'Leaf-Peer'. This is useful in troubleshooting cases.

3.5.4. IS-IS Reverse Metric

This metric is part of the aggregated metric for leaf's default route installation with load sharing among the spine nodes. When a spine node is in 'overload' condition, it should use the IS-IS Reverse Metric TLV in IIH [REVERSE-METRIC] to set this metric to maximum to discourage the leaf using it as part of the loadsharing.

In some cases, certain spine nodes may have less bandwidth in link provisioning or in real-time condition, and it can use this metric to signal to the leaf nodes dynamically.

In other cases, such as when the spine node loses a link to a particular leaf node, although it can redirect the traffic to other spine nodes to reach that destination leaf node, but it MAY want to increase this metric value if the inter-spine connection becomes over utilized, or the latency becomes an issue.

In the leaf-leaf link as a backup gateway use case, the 'Reverse Metric' SHOULD always be set to very high value.

3.5.5. Spine-Leaf Traffic Engineering

Besides using the IS-IS Reverse Metric by the spine nodes to affect the traffic pattern for leaf default gateway towards multiple spine nodes, the IPv6/IPv4 Info-Advertise sub-TLVs can be selectively used by traffic engineering controllers to move data traffic around the data center fabric to alleviate congestion and to reduce the latency of a certain class of traffic pairs. By injecting more specific leaf node prefixes, it will allow the spine nodes to attract more traffic on some underutilized links.

3.5.6. Other End-to-End Services

Losing the topology information will have an impact on some of the end-to-end network services, for instance, MPLS TE or end-to-end segment routing. Some other mechanisms such as those described in PCE [RFC4655] based solution may be used. In this Spine-Leaf extension, the role of the leaf node is not too much different from the multi-level IS-IS routing while the level-1 IS-IS nodes only have the default route information towards the node which has the Attach Bit (ATT) set, and the level-2 backbone does not have any topology information of the level-1 areas. The exact mechanism to enable certain end-to-end network services in Spine-Leaf network is outside the scope of this document.

3.5.7. Address Family and Topology

IPv6 Address families[RFC5308], Multi-Topology (MT)[RFC5120] and Multi-Instance (MI)[RFC8202] information is carried over the IIH PDU. Since the goal is to simplify the operation of IS-IS network, for the simplicity of this extension, the Spine-Leaf mechanism is applied the same way to all the address families, MTs and MIs.

3.5.8. Migration

For this extension to be deployed in existing networks, a simple migration scheme is needed. To support any leaf node in the network, all the involved spine nodes have to be upgraded first. So the first step is to migrate all the involved spine nodes to support this extension, then the leaf nodes can be enabled with 'Leaf-Mode' one by one. No flag day is needed for the extension migration.

4. IANA Considerations

A new TLV codepoint is defined in this document and needs to be assigned by IANA from the "IS-IS TLV Codepoints" registry. It is referred to as the Spine-Leaf TLV and the suggested value is 150. This TLV is only to be optionally inserted either in the IIH PDU or in the Circuit Flooding Scoped LSP PDU. IANA is also requested to maintain the SL-flag bit values in this TLV, and 0x01, 0x02 and 0x04 bits are defined in this document.

Value	Name	IIH	LSP	SNP	Purge	CS-LSP
-----	-----	---	---	---	-----	-----
150	Spine-Leaf	y	y	n	n	y

This extension also proposes to have the Dynamic Hostname TLV, already assigned as code 137, to be allowed in IIH PDU.

Value	Name	IIH	LSP	SNP	Purge
-----	-----	---	---	---	-----
137	Dynamic Name	y	y	n	y

Two new sub-TLVs are defined in this document and needs to be added assigned by IANA from the "IS-IS TLV Codepoints". They are referred to in this document as the Leaf-Set sub-TLV and the Info-Req sub-TLV. It is suggested to have the values 1 and 2 respectively.

This document also requests that IANA allocate from the registry of link-attribute bit values for sub-TLV 19 of TLV 22 (Extended IS reachability TLV). This new bit is referred to as the "Connect to RF-Leaf Node" bit.

Value	Name	Reference
-----	-----	-----
0x3	Connect to RF-Leaf Node	This document

5. Security Considerations

Security concerns for IS-IS are addressed in [ISO10589], [RFC5304], [RFC5310], and [RFC7602]. This extension does not raise additional security issues.

6. Acknowledgments

The authors would like to thank Tony Przygienda for his discussion and contributions. The authors also would like to thank Acee Lindem, Russ White and Christian Hopps for their review and comments of this document.

7. Document Change Log

7.1. Changes to draft-shen-isis-spine-leaf-ext-05.txt

- o Submitted January 2018.
- o Just a refresh.

7.2. Changes to draft-shen-isis-spine-leaf-ext-04.txt

- o Submitted June 2017.
- o Added the Tier level information to handle the multi-level spine-leaf topology using this extension.

7.3. Changes to draft-shen-isis-spine-leaf-ext-03.txt

- o Submitted March 2017.
- o Added the Spine-Leaf sub-TLVs to handle the case of data center pure CLOS topology and mechanism.
- o Added the Spine-Leaf TLV and sub-TLVs can be optionally inserted in either IIH PDU or CS-LSP PDU.
- o Allow use of prefix Reachability TLVs 135 and 236 in IIHs/CS-LSPs sent from spine to leaf.

7.4. Changes to draft-shen-isis-spine-leaf-ext-02.txt

- o Submitted October 2016.
- o Removed the 'Default Route Metric' field in the Spine-Leaf TLV and changed to using the IS-IS Reverse Metric in IIH.

7.5. Changes to draft-shen-isis-spine-leaf-ext-01.txt

- o Submitted April 2016.
- o No change. Refresh the draft version.

7.6. Changes to draft-shen-isis-spine-leaf-ext-00.txt

- o Initial version of the draft is published in November 2015.

8. References

8.1. Normative References

[ISO10589]

ISO "International Organization for Standardization",
"Intermediate system to Intermediate system intra-domain
routing information exchange protocol for use in
conjunction with the protocol for providing the
connectionless-mode Network Service (ISO 8473), ISO/IEC
10589:2002, Second Edition.", Nov 2002.

[REVERSE-METRIC]

Shen, N., Amante, S., and M. Abrahamsson, "IS-IS Routing
with Reverse Metric", draft-ietf-isis-reverse-metric-07
(work in progress), 2017.

[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/info/rfc2119>>.

[RFC5029]

Vasseur, JP. and S. Previdi, "Definition of an IS-IS Link
Attribute Sub-TLV", RFC 5029, DOI 10.17487/RFC5029,
September 2007, <<https://www.rfc-editor.org/info/rfc5029>>.

[RFC5120]

Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi
Topology (MT) Routing in Intermediate System to
Intermediate Systems (IS-ISs)", RFC 5120,
DOI 10.17487/RFC5120, February 2008, <<https://www.rfc-editor.org/info/rfc5120>>.

[RFC5301]

McPherson, D. and N. Shen, "Dynamic Hostname Exchange
Mechanism for IS-IS", RFC 5301, DOI 10.17487/RFC5301,
October 2008, <<https://www.rfc-editor.org/info/rfc5301>>.

- [RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, DOI 10.17487/RFC5304, October 2008, <<https://www.rfc-editor.org/info/rfc5304>>.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <<https://www.rfc-editor.org/info/rfc5305>>.
- [RFC5306] Shand, M. and L. Ginsberg, "Restart Signaling for IS-IS", RFC 5306, DOI 10.17487/RFC5306, October 2008, <<https://www.rfc-editor.org/info/rfc5306>>.
- [RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <<https://www.rfc-editor.org/info/rfc5308>>.
- [RFC5310] Bhatia, M., Manral, V., Li, T., Atkinson, R., White, R., and M. Fanto, "IS-IS Generic Cryptographic Authentication", RFC 5310, DOI 10.17487/RFC5310, February 2009, <<https://www.rfc-editor.org/info/rfc5310>>.
- [RFC7356] Ginsberg, L., Previdi, S., and Y. Yang, "IS-IS Flooding Scope Link State PDUs (LSPs)", RFC 7356, DOI 10.17487/RFC7356, September 2014, <<https://www.rfc-editor.org/info/rfc7356>>.
- [RFC7602] Chunduri, U., Lu, W., Tian, A., and N. Shen, "IS-IS Extended Sequence Number TLV", RFC 7602, DOI 10.17487/RFC7602, July 2015, <<https://www.rfc-editor.org/info/rfc7602>>.
- [RFC8202] Ginsberg, L., Previdi, S., and W. Henderickx, "IS-IS Multi-Instance", RFC 8202, DOI 10.17487/RFC8202, June 2017, <<https://www.rfc-editor.org/info/rfc8202>>.

8.2. Informative References

- [DYNAMIC-FLOODING] Li, T., "Dynamic Flooding on Dense Graphs", draft-li-dynamic-flooding (work in progress), 2018.
- [RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <<https://www.rfc-editor.org/info/rfc4655>>.

[RFC5309] Shen, N., Ed. and A. Zinin, Ed., "Point-to-Point Operation over LAN in Link State Routing Protocols", RFC 5309, DOI 10.17487/RFC5309, October 2008, <<https://www.rfc-editor.org/info/rfc5309>>.

Authors' Addresses

Naiming Shen
Cisco Systems
560 McCarthy Blvd.
Milpitas, CA 95035
US

Email: naiming@cisco.com

Les Ginsberg
Cisco Systems
821 Alder Drive
Milpitas, CA 95035
US

Email: ginsberg@cisco.com

Sanjay Thyamagundalu

Email: tsanjay@gmail.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 15, 2018

R. White, Ed.
S. Zandi, Ed.
LinkedIn
October 12, 2017

IS-IS Support for Openfabric
draft-white-openfabric-04

Abstract

Spine and leaf topologies are widely used in hyperscale and cloud scale networks. In most of these networks, configuration is automated, but difficult, and topology information is extracted through broad based connections. Policy is often integrated into the control plane, as well, making configuration, management, and troubleshooting difficult. Openfabric is an adaptation of an existing, widely deployed link state protocol, Intermediate System to Intermediate System (IS-IS) that is designed to:

- o Provide a full view of the topology from a single point in the network to simplify operations
- o Minimize configuration of each Intermediate System (IS) (also called a router or switch) in the network
- o Optimize the operation of IS-IS within a spine and leaf fabric to enable scaling

This document begins with an overview of openfabric, including a description of what may be removed from IS-IS to enable scaling. The document then describes an optimized adjacency formation process; an optimized flooding scheme; some thoughts on the operation of openfabric, metrics, and aggregation; and finally a description of the changes to the IS-IS protocol required for openfabric.

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 April 15, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Goals	3
1.2. Contributors	3
1.3. Simplification	3
1.4. Additions and Requirements	4
1.5. Sample Network	4
2. Modified Adjacency Formation	6
2.1. Level 2 Adjacencies Only	6
2.2. Point-to-point Adjacencies	6
2.3. Three Way Handshake Support	7
2.4. Adjacency Formation Optimization	7
3. Advertisement of Reachability Information	7
4. Determining and Advertising Location on the Fabric	8
4.1. Calculating Tier Number with a Fixed T0	9
4.2. Calculating the Tier Number in a Five Stage Spine and Leaf	10
5. Flooding Optimization	11
5.1. Flooding Failures	13
6. Other Optimizations	13
6.1. Transit Link Reachability	13
6.2. Transiting T0 Intermediate Systems	13
7. Openfabric and Route Aggregation	14
8. Security Considerations	14
9. References	14
9.1. Normative References	14
9.2. Informative References	16

Authors' Addresses	17
--------------------	----

1. Introduction

1.1. Goals

Spine and leaf fabrics are often used in large scale data centers; in this application, they are commonly called a fabric because of their regular structure and predictable forwarding and convergence properties. This document describes modifications to the IS-IS protocol to enable it to run efficiently on a large scale spine and leaf fabric, openfabric. The goals of this control plane are:

- o Provide a full view of the topology from a single point in the network to simplify operations
- o Minimize configuration of each IS in the network
- o Optimize the operation of IS-IS within a spine and leaf fabric to enable scaling

1.2. Contributors

The following people have contributed to this draft: Nikos Triantafyllis (reflected flooding optimization), Ivan Pepelnjak (three stage fabric modifications), Hannes Gredler (do not reflood optimizations), Les Ginsberg (capabilities encoding, circuit local reflooding), Naiming Shen (capabilities encoding, circuit local reflooding), Uma Chunduri (failure mode suggestions, flooding), Nick Russo, and Rodny Molina.

See [RFC5449], [RFC5614], and [RFC7182] for similar solutions in the Mobile Ad Hoc Networking (MANET) solution space.

1.3. Simplification

In building any scalable system, it is often best to begin by removing what is not needed. In this spirit, openfabric implementations MAY remove the following from IS-IS:

- o External metrics. There is no need for external metrics in large scale spine and leaf fabrics; it is assumed that metrics will be properly configured by the operator to account for the correct order of route preference at any route redistribution point.
- o Tags and traffic engineering processing. Openfabric is only designed to provide topology and reachability information. It is not designed to provide for traffic engineering, route preference

through tags, or other policy mechanisms. It is assumed that all routing policy will be provided through an overlay system which communicates directly with each IS in the fabric, such as PCEP [RFC5440] or I2RS [RFC7921]. Traffic engineering is assumed to be provided through Segment Routing (SR) [I-D.ietf-spring-segment-routing].

1.4. Additions and Requirements

To create a scalable link state fabric, openfabric includes the following:

- o A slightly modified adjacency formation process.
- o Mechanisms for determining which tier within a spine and leaf fabric in which the IS is located.
- o A mechanism that reduces flooding to the minimum possible, while still ensuring complete database synchronization among the intermediate systems within the fabric.

Three general requirements are placed here; more specific requirements are considered in the following sections. Openfabric implementations:

- o MUST support [RFC5301] and enable hostname advertisement by default if a hostname is configured on the intermediate system.
- o SHOULD support [RFC6232], purge originator identification for IS-IS.
- o MUST NOT be mixed with standard IS-IS implementations in operational deployments. Openfabric and standard IS-IS implementations SHOULD be treated as two separate protocols.

1.5. Sample Network

The following spine and leaf fabric will be used to describe these modifications.

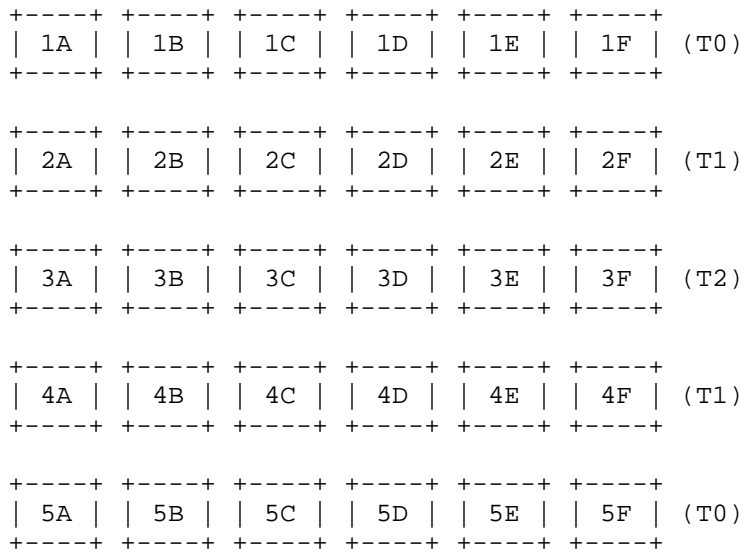


Figure 1

To reduce confusion (spine and leaf fabrics are difficult to draw in plain text art), this diagram does not contain the connections between devices. The reader should assume that each device in a given layer is connected to every device in the layer above it. For instance:

- o 5A is connected to 4A, 4B, 4C, 4D, 4E, and 4F
- o 5B is connected to 4A, 4B, 4C, 4D, 4E, and 4F
- o 4A is connected to 3A, 3B, 3C, 3D, 3E, 3F, 5A, 5B, 5C, 5D, 5E, and 5F
- o 4B is connected to 3A, 3B, 3C, 3D, 3E, 3F, 5A, 5B, 5C, 5D, 5E, and 5F
- o etc.

The tiers or stages of the fabric are also marked for easier reference. T0 is assumed to be connected to application servers, or rather they are Top of Rack (ToR) intermediate systems. The remaining tiers, T1 and T2, are connected only to the fabric itself. Note there are no "cross links," or "east west" links in the illustrated fabric. The fabric locality detection mechanism described here will not work if there are cross links running east/

west through the fabric. Locality detection may be possible in such a fabric; this is an area for further study.

2. Modified Adjacency Formation

Because Openfabric operates in a tightly controlled data center environment, various modifications can be made to the IS-IS neighbor formation process to increase efficiency and simplify the protocol. Specifically, Openfabric implementations SHOULD support [RFC3719], section 4, hello padding for IS-IS. Variable hello padding SHOULD NOT be used, as data center fabrics are built using high speed links on which padded hellos will have little performance impact. Further modifications to the neighbor formation process are considered in the following sections.

2.1. Level 2 Adjacencies Only

Openfabric is designed to work in a single flooding domain over a single data center fabric at the scale of thousands of routers with hundreds of thousands of routes (so a moderate scale in router and route count terms). Because of the way Openfabric optimizes operation in this environment, it is not necessary nor desirable to build multiple flooding domains. For instance, the flooding optimizations described later in this document require a full view of the topology, as does any proposed overlay to inject policy into the forwarding plane. In light of this, the following changes SHOULD BE to IS-IS implementations to support Openfabric:

- o IIH PDU 16 (level 2 broadcast circuit hello) should be the only IIH PDU type transmitted (see section 9.6 of [ISO10589] and section 4.1 of [RFC5309])
- o In IIH PDU 16 (level 2 broadcast circuit hello), the Circuit Type field should be set to 2 (see section 9.6 of [ISO10589])
- o Support for IIH PDU 15 (level 1 broadcast hello) should be removed (see section 9.5 of [ISO10589])
- o Support for IIH PDU 17 (point-to-point hello) should be removed (see section 9.7 of [ISO10589])

2.2. Point-to-point Adjacencies

Data center network fabrics only contain point-to-point links; because of this, there is no reason to support any broadcast link types, nor to support the Designated Intermediate System processing, including pseudonode creation. In light of this, processing related to sections 7.2.3 (broadcast networks), 7.3.8 (generation of level 1

pseudonode LSPs), 7.3.10 (generation of level 2 pseudonode LSPs), and section 8.4.5 (LAN designated intermediate systems) in [ISO10589] SHOULD BE removed.

2.3. Three Way Handshake Support

It is important that two way connectivity be established before synchronizing the link state database, or routing through a link in a data center fabric. To reject optical failures that cause a one way connection between two routers, fabricDC must support the three way handshake mechanism described in [RFC5303].

2.4. Adjacency Formation Optimization

While adjacency formation is not considered particularly burdensome in IS-IS, it is still useful to reduce the amount of state transferred across the network when connecting a new IS to the fabric. Any such optimization is bound to present a tradeoff between several factors; the mechanism described here increases the amount of time required to form adjacencies slightly in order to reduce the total state carried across the network. The process is:

- o An IS connected to the fabric will send hellos on all links.
- o The IS will only complete the three-way handshake with one newly discovered neighbor; this would normally be the first neighbor which sends the newly connected intermediate system's ID back in the three-way handshake process.
- o The IS will complete its database exchange with this one newly adjacent neighbor.
- o Once this process is completed, the IS will continue processing the remaining neighbors as normal.

This process allows each IS newly added to the fabric to exchange a full table once; a very minimal amount of information will be transferred with the remaining neighbors to reach full synchronization.

3. Advertisement of Reachability Information

IS-IS describes the topology in two different sets of TLVs; the first describes the set of neighbors connected to an IS, the second describes the set of reachable destination connected to an IS. There are two different forms of both of these descriptions, one of which carries what are widely called narrow metrics, the other of which carries what are widely called wide metrics. In a tightly controlled

data center fabric implementation, such as the ones Openfabric is designed to support, no IS that supports narrow metrics will ever be deployed or supported; hence there is no reason to support any metric type other than wide metrics.

- o The Level 2 Link State PDU (type 20 in section 9.9 of [ISO10589]) and the scoped flooding PDU (type 10 in section 3.1 of [RFC7356]) SHOULD BE the only PDU types used to carry link state information in a Openfabric implementation
- o Processing related to the Level 1 Link State PDU (type 18) MAY BE removed from Openfabric implementations (see section 9.8 of [ISO10589])
- o Neighbor reachability MUST BE carried in TLV type 22 (see section 3 of [RFC5305])
- o IPv4 reachability SHOULD BE carried in TLV type 135 (see section 4 of [RFC5305]), or TLV type 235 for multitopology implementations (see [RFC5120])
- o IPv6 reachability SHOULD BE carried in TLV type 236 (see [RFC5308]), or TLV type 237 for multitopology implemenations (see [RFC5120])
- o Processing related to the neighbor reachability TLV (type 2, see sections 9.8 and 9.9 of [ISO10589]) SHOULD BE removed
- o Processing related to the narrow metric IP reachability TLV (types 128 and 130) SHOULD BE removed

In order to support segment routing, Openfabric needs to be able to support the advertisement of a Prefix-SID tied to a local loopback address assigned to the IS. The configuration of the label to advertise MAY BE manually configured for the moment or determined through autoconfiguration. A Prefix-SID SHOULD BE advertised if a local label is configured using the Prefix Segment Identifier sub-TLV (see section 2.1 of [I-D.ietf-isis-segment-routing-extensions]).

4. Determining and Advertising Location on the Fabric

The tier to which a IS is connected is useful to enable autoconfiguration of intermediate systems connected to the fabric and to reduce flooding. Once the tier of an intermediate system within the fabric has been determined, it MUST be advertised using the 4 bit Tier field described in section 3.3 of [I-D.shen-isis-spine-leaf-ext]. This section describes two

mechanisms for determining the tier at which a IS is connected in the fabric in several steps.

4.1. Calculating Tier Number with a Fixed T0

The first method begins with one of the T0 intermediate systems advertising its location in the fabric. This information can either be obtained through:

- o A single T0 intermediate system is manually configured to advertise 0x00 in their IS reachability tier sub-TLV, indicating they are at the edge of the fabric (a ToR IS).
- o The T0 intermediate systems detect they are T0 through the presence connected hosts (i.e. through a request for address assignment or some other means). If such detection is used, and the IS determines it is located at T0, it should advertise 0x00 in its IS reachability tier sub-TLV.

The second method above SHOULD be used with care, as it may not be secure, and it may not work in all data center environments. For instance, if a host is mistakenly (or intentionally, as a form of attack) attached to a spine IS, or a request for address assignment is transmitted to a spine IS during the bootup phase of the device or fabric, it is possible to cause a spine IS to advertise itself as a T0. Unless the autodetection of the T0 devices is secured, the manual mechanism SHOULD BE used (configuring at least one T0 device manually).

Given at least one T0 device is advertising its tier number, the remaining intermediate systems calculate their tier number as follows:

- o The local IS calculates an SPT (using SPF) setting the cost of every link to 1; this effectively calculates a topology only view of the network, without considering any configured link costs
- o Find the closest IS advertising a tier number of 0 in the Spine Leaf extension sub-TLV; call this node A, and set FD to this cost
- o Calculate an SPT (using SPF) from the perspective of A (above), and setting the cost of every link to 1; the maximum cost to any node should be 2 for a 3 stage fabric, 4 for a 5 stage fabric, etc.
- o Choose any node that is a maximum metric from A (above); call this IS B

- o Find the cost to B on the locally calculated SPT from the first step; call this TD
- o Calculate the tier number of the local node by subtracting FD from TD

In the example network, assume 5A is manually configured as a T0, and is advertising its tier number. From here:

- o From 1A the path to 5A is 4 hops; this is FD
- o Run SPF from the perspective of 5A with all link metrics set to 1
- o The maximum path length is 4; 1F is one such node; set this node to B, and set TD to 4
- o $TD - FD$ is 0 at 1A, so 1A is T0, or a ToR

This process will work for any spine and leaf fabric without "cross links."

4.2. Calculating the Tier Number in a Five Stage Spine and Leaf

In some fabrics, it is possible to calculate which intermediate systems are at T0 using a modified Shortest Path First (SPF) calculation. Specifically, if the fabric is configured in five stages, as shown in the example network, and is not some form of butterfly, Benes, or a three stage fabric, it is possible to calculate if an IS is at T0 using the following process:

- o Calculate a Shortest Path Tree (SPT) for the entire network with all link metrics set to 1; this has the effect of calculating a tree based only on hop count
- o Find one node that is the farthest from the local node in the resulting tree; call this node F, and the distance to this node FD
- o Calculate an SPT for the entire network with all link metrics set to 1 from the perspective of F; call this TD

If $FD == TD$, and $TD \geq 4$, this is a greater than three stage fabric; the local device SHOULD advertise 0x00 in its IS reachability tier sub-TLV. For instance, in the diagram above, 1A would:

- o Calculate an SPT with all link metrics set to 1; on this SPT, 5A through 5F would all have a distance of 4
- o Select one of these nodes as F; assume 5F is chosen as F

- o Set FD to 4, the distance to 5F
- o Run SPF from the perspective of 5F with all link metrics set to 1
- o Set TD to 4, the cost from 5F to 1A
- o $TD - FD == 0$, so 1A is at T0, and is a ToR

For the remaining intermediate systems to determine which tier they are situated on, they perform the following calculation:

- o Calculate a Shortest Path Tree (SPT) for the entire network with all link metrics set to 1; this has the effect of calculating a tree based only on hop count
- o Find one node that is the farthest from the local node in the resulting tree; call this node F, and the distance to this node FD
- o Calculate an SPT for the entire network with all link metrics set to 1 from the perspective of F; call this TD

The IS SHOULD advertise (TD - FD) in its IS reachability tier sub-TLV.

For example, in the above five stage fabric, 3B would:

- o Calculate an SPT with all link metrics set to 1; on this SPT, 5A through 5F and 1A through 1F would all have a cost of 2
- o Select one of these nodes as F; assume 5F is chosen as F
- o Set FD to 2, the distance to 5F
- o Run SPF from the perspective of 5F with all link metrics set to 1
- o Set TD to 4, the cost from 5F to 1A
- o $TD - FD == 2$, so 1A is at T2, and is a spine switch

5. Flooding Optimization

Flooding is perhaps the most challenging scaling issue for a link state protocol running on a dense, large scale fabric. To reduce the flooding of link state information in the form of Link State Protocol Data Units (LSPs), Openfabric takes advantage of information already available in the link state protocol, the list of the local intermediate system's neighbor's neighbors, and the fabric locality

computed above. The following tables are required to compute a set of reflooders:

- o Neighbor List (NL) list: The set of neighbors
- o Neighbor's Neighbors (NN) list: The set of neighbor's neighbors; this can be calculated by running SPF truncated to two hops
- o Do Not Reflood (DNR) list: The set of neighbors who should have LSPs (or fragments) who should not reflood LSPs
- o Reflood (RF) list: The set of neighbors who should flood LSPs (or fragments) to their adjacent neighbors to ensure synchronization

NL is set to contain all neighbors, and sorted deterministically (for instance, from the highest IS identifier to the lowest). All intermediate systems within a single fabric SHOULD use the same mechanism for sorting the NL list. NN is set to contain all neighbor's neighbors, or all intermediate systems that are two hops away, as determined by performing a truncated SPF. The DNR and RF tables are initially empty. To begin, the following steps are taken to reduce the size of NN and NL:

- o Move any IS in NL with its tier (or fabric location) set to T0 to DNR
- o Remove all intermediate systems from NL and NN that in the shortest path to the IS that originated the LSP

Then, for every IS in NL:

- o If the current entry in NL is connected to any entries in NN:
 - * Move the IS to RF
 - * Remove the intermediate systems connected to the IS from NN
- o Else move the IS to DNR

When flooding, LSPs transmitted to adjacent neighbors on the RF list will be transmitted normally. Adjacent intermediate systems on this list will reflood received LSPs into the next stage of the topology, ensuring database synchronization. LSPs transmitted to adjacent neighbors on the DNR list, however, MUST be transmitted using a circuit scope PDU as described in [RFC7356].

5.1. Flooding Failures

It is possible in some failure modes for flooding to be incomplete because of the flooding optimizations outlined. Specifically, if a reflooder fails, or is somehow disconnected from all the links across which it should be reflooding, it is possible an LSP is only partially flooded through the fabric. To prevent such situations, any IS receiving an LSP transmitted using DNR SHOULD:

- o Set a short timer; the default should be less than one second
- o When the timer expires, send a Complete Sequence Number Packet (CSNP) to all neighbors
- o Process any Partial Sequence Number Packets (PSNPs) as required to resynchronize
- o If a resynchronization is required, notify the network operator through a network management system

6. Other Optimizations

6.1. Transit Link Reachability

In order to reduce the amount of control plane state carried on large scale spine and leaf fabrics, openfabric implementations SHOULD NOT advertise reachability for transit links. These links MAY remain unnumbered, as IS-IS does not require layer 3 IP addresses to operate. Each IS SHOULD be configured with a single loopback address, which is assigned an IPv6 address, to provide reachability to intermediate systems which make up the fabric.

6.2. Transiting T0 Intermediate Systems

In data center fabrics, ToR intermediate systems SHOULD NOT be used to transit between two T1 (or above) spine intermediate systems. The simplest way to prevent this is to set the overload bit [RFC3277] for all the LSPs originated from T0 intermediate systems. However, this solution would have the unfortunate side effect of causing all reachability beyond any T0 IS to have the same metric, and many implementations treat a set overload bit as a metric of 0xFFFF in calculating the Shortest Path Tree (SPT). This document proposes an alternate solution which preserves the leaf node metric, while still avoiding transiting T0 intermediate systems.

Specifically, all T0 intermediate systems SHOULD advertise their metric to reach any T1 adjacent neighbor with a cost of 0XFFE. T1 intermediate systems, on the other hand, will advertise T0

intermediate systems with the actual interface cost used to reach the T0 IS. Hence, links connecting T0 and T1 intermediate systems will be advertised with an asymmetric cost that discourages transiting T0 intermediate systems, while leaving reachability to the destinations attached to T0 devices the same.

7. Openfabric and Route Aggregation

While schemes may be designed so reachability information can be aggregated in Openfabric deployments, this is not a recommended configuration.

8. Security Considerations

This document outlines modifications to the IS-IS protocol for operation on large scale data center fabrics. While it does add new TLVs, and some local processing changes, it does not add any new security vulnerabilities to the operation of IS-IS. However, openfabric implementations SHOULD implement IS-IS cryptographic authentication, as described in [RFC5304], and should enable other security measures in accordance with best common practices for the IS-IS protocol.

If T0 intermediate systems are auto-detected using information outside Openfabric, it is possible to attack the calculations used for flooding reduction and auto-configuration of intermediate systems. For instance, if a request for an address pool is used as an indicator of an attached host, and hence receiving such a request causes an intermediate system to advertise itself as T0, it is possible for an attacker (or a simple mistake) to cause auto-configuration to fail. Any such auto-detection mechanisms SHOULD BE secured using appropriate techniques, as described by any protocols or mechanisms used.

9. References

9.1. Normative References

- [I-D.shen-isis-spine-leaf-ext]
Shen, N., Ginsberg, L., and S. Thyamagundalu, "IS-IS Routing for Spine-Leaf Topology", draft-shen-isis-spine-leaf-ext-04 (work in progress), June 2017.

- [ISO10589] 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)", ISO/IEC 10589:2002, Second Edition, Nov 2002.
- [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/info/rfc2119>>.
- [RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, DOI 10.17487/RFC2629, June 1999, <<https://www.rfc-editor.org/info/rfc2629>>.
- [RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, DOI 10.17487/RFC5120, February 2008, <<https://www.rfc-editor.org/info/rfc5120>>.
- [RFC5301] McPherson, D. and N. Shen, "Dynamic Hostname Exchange Mechanism for IS-IS", RFC 5301, DOI 10.17487/RFC5301, October 2008, <<https://www.rfc-editor.org/info/rfc5301>>.
- [RFC5303] Katz, D., Saluja, R., and D. Eastlake 3rd, "Three-Way Handshake for IS-IS Point-to-Point Adjacencies", RFC 5303, DOI 10.17487/RFC5303, October 2008, <<https://www.rfc-editor.org/info/rfc5303>>.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <<https://www.rfc-editor.org/info/rfc5305>>.
- [RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <<https://www.rfc-editor.org/info/rfc5308>>.
- [RFC5309] Shen, N., Ed. and A. Zinin, Ed., "Point-to-Point Operation over LAN in Link State Routing Protocols", RFC 5309, DOI 10.17487/RFC5309, October 2008, <<https://www.rfc-editor.org/info/rfc5309>>.

- [RFC5311] McPherson, D., Ed., Ginsberg, L., Previdi, S., and M. Shand, "Simplified Extension of Link State PDU (LSP) Space for IS-IS", RFC 5311, DOI 10.17487/RFC5311, February 2009, <<https://www.rfc-editor.org/info/rfc5311>>.
- [RFC5316] Chen, M., Zhang, R., and X. Duan, "ISIS Extensions in Support of Inter-Autonomous System (AS) MPLS and GMPLS Traffic Engineering", RFC 5316, DOI 10.17487/RFC5316, December 2008, <<https://www.rfc-editor.org/info/rfc5316>>.
- [RFC7356] Ginsberg, L., Previdi, S., and Y. Yang, "IS-IS Flooding Scope Link State PDUs (LSPs)", RFC 7356, DOI 10.17487/RFC7356, September 2014, <<https://www.rfc-editor.org/info/rfc7356>>.
- [RFC7981] Ginsberg, L., Previdi, S., and M. Chen, "IS-IS Extensions for Advertising Router Information", RFC 7981, DOI 10.17487/RFC7981, October 2016, <<https://www.rfc-editor.org/info/rfc7981>>.

9.2. Informative References

- [I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and j. jeffrant@gmail.com, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-13 (work in progress), June 2017.
- [I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-12 (work in progress), June 2017.
- [RFC3277] McPherson, D., "Intermediate System to Intermediate System (IS-IS) Transient Blackhole Avoidance", RFC 3277, DOI 10.17487/RFC3277, April 2002, <<https://www.rfc-editor.org/info/rfc3277>>.
- [RFC3719] Parker, J., Ed., "Recommendations for Interoperable Networks using Intermediate System to Intermediate System (IS-IS)", RFC 3719, DOI 10.17487/RFC3719, February 2004, <<https://www.rfc-editor.org/info/rfc3719>>.
- [RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <<https://www.rfc-editor.org/info/rfc4271>>.

- [RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, DOI 10.17487/RFC5304, October 2008, <<https://www.rfc-editor.org/info/rfc5304>>.
- [RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <<https://www.rfc-editor.org/info/rfc5440>>.
- [RFC5449] Baccelli, E., Jacquet, P., Nguyen, D., and T. Clausen, "OSPF Multipoint Relay (MPR) Extension for Ad Hoc Networks", RFC 5449, DOI 10.17487/RFC5449, February 2009, <<https://www.rfc-editor.org/info/rfc5449>>.
- [RFC5614] Ogier, R. and P. Spagnolo, "Mobile Ad Hoc Network (MANET) Extension of OSPF Using Connected Dominating Set (CDS) Flooding", RFC 5614, DOI 10.17487/RFC5614, August 2009, <<https://www.rfc-editor.org/info/rfc5614>>.
- [RFC6232] Wei, F., Qin, Y., Li, Z., Li, T., and J. Dong, "Purge Originator Identification TLV for IS-IS", RFC 6232, DOI 10.17487/RFC6232, May 2011, <<https://www.rfc-editor.org/info/rfc6232>>.
- [RFC7182] Herberg, U., Clausen, T., and C. Dearlove, "Integrity Check Value and Timestamp TLV Definitions for Mobile Ad Hoc Networks (MANETs)", RFC 7182, DOI 10.17487/RFC7182, April 2014, <<https://www.rfc-editor.org/info/rfc7182>>.
- [RFC7921] Atlas, A., Halpern, J., Hares, S., Ward, D., and T. Nadeau, "An Architecture for the Interface to the Routing System", RFC 7921, DOI 10.17487/RFC7921, June 2016, <<https://www.rfc-editor.org/info/rfc7921>>.

Authors' Addresses

Russ White (editor)
LinkedIn

Email: russ@riw.us

Shawn Zandi (editor)
LinkedIn

Email: szandi@linkedin.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2018

X. Xu
Huawei
L. Fang
ebay
J. Tantsura
Individual
October 30, 2017

IS-IS Flooding Reduction in MSDC
draft-xu-isis-flooding-reduction-in-msdc-02

Abstract

IS-IS is commonly used as an underlay routing protocol for MSDC (Massively Scalable Data Center) networks. For a given IS-IS router within the CLOS topology, it would receive multiple copies of exactly the same LSP from multiple IS-IS neighbors. In addition, two IS-IS neighbors may send each other the same LSP simultaneously. The unnecessary link-state information flooding wastes the precious process resource of IS-IS routers greatly due to the fact that there are too many IS-IS neighbors for each IS-IS router within the CLOS topology. This document proposes some extensions to IS-IS so as to reduce the IS-IS flooding within MSDC networks greatly. The reduction of the IS-IS flooding is much beneficial to improve the scalability of MSDC networks.

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 May 3, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	4
2. Terminology	4
3. Modifications to Current IS-IS Behaviors	4
3.1. IS-IS Routers as Non-DIS	4
3.2. Controllers as DIS	5
4. Acknowledgements	5
5. IANA Considerations	5
6. Security Considerations	5
7. References	5
7.1. Normative References	5
7.2. Informative References	6
Authors' Addresses	6

1. Introduction

IS-IS is commonly used as an underlay routing protocol for Massively Scalable Data Center (MSDC) networks where CLOS is the most popular topology. For a given IS-IS router within the CLOS topology, it would receive multiple copies of exactly the same LSP from multiple IS-IS neighbors. In addition, two IS-IS neighbors may send each other the same LSP simultaneously. The unnecessary link-state information flooding wastes the precious process resource of IS-IS routers greatly and therefore IS-IS could not scale very well in MSDC networks.

To simplify the network management task, centralized controllers are becoming fundamental network elements in most MSDCs. One or more controllers are usually connected to all routers within the MSDC network via a Local Area Network (LAN) which is dedicated for network management purpose (called management LAN), as shown in Figure 1.

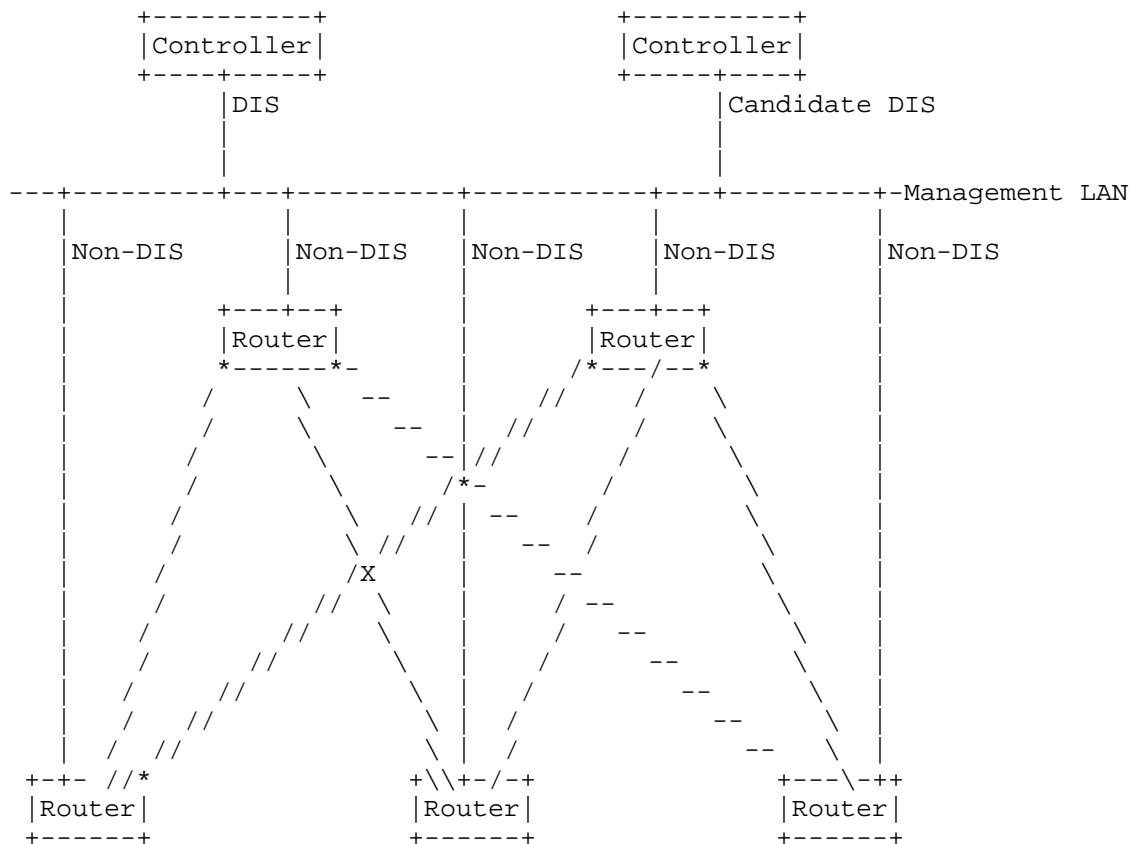


Figure 1

With the assistance of a controller acting as IS-IS Designated Intermediate System (DIS) for the management LAN, IS-IS routers within the MSDC network don't need to exchange any IS-IS Protocol Datagram Units (PDUs) other than Hello packets among them. In order to obtain the full topology information (i.e., the fully synchronized link-state database) of the MSDC's network, these IS-IS routers would exchange the link-state information with the controller being elected as IS-IS DIS for the management LAN instead.

To further suppress the flooding of multicast IS-IS PDUs originated from IS-IS routers over the management LAN, IS-IS routers would not send multicast IS-IS Hello packets over the management LAN. Insteads, they just wait for IS-IS Hello packets originated from the controller being elected as IS-IS DIS initially. Once an IS-IS DIS for the management LAN has been discovered, they start to send IS-IS Hello packets directly (as unicasts) to the IS-IS DIS periodically.

In addition, IS-IS routers would send IS-IS PDUs to the IS-IS DIS for the management LAN as unicasts as well. In contrast, the controller being elected as IS-IS DIS would send IS-IS PDUs as before. As a result, IS-IS routers would not receive IS-IS PDUs from one another unless these IS-IS PDUs are forwarded as unknown unicasts over the management LAN. Through the above modifications to the current IS-IS router behaviors, the IS-IS flooding is greatly reduced, which is much beneficial to improve the scalability of MSDC networks.

1.1. 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].

2. Terminology

This memo makes use of the terms defined in [RFC1195].

3. Modifications to Current IS-IS Behaviors

3.1. IS-IS Routers as Non-DIS

After the bidirectional exchange of IS-IS Hello packets among IS-IS routers, IS-IS routers would originate Link State PDUs (LSPs) accordingly. However, these self-originated LSPs need not to be exchanged directly among them anymore. Instead, these LSPs just need to be sent solely to the controller being elected as IS-IS DIS for the management LAN.

To further reduce the flood of multicast IS-IS PDUs over the management LAN, IS-IS routers SHOULD send IS-IS PDUs as unicasts. More specifically, IS-IS routers SHOULD send unicast IS-IS Hello packets periodically to the controller being elected as IS-IS DIS. In other words, IS-IS routers would not send any IS-IS Hello packet over the management LAN until they have found an IS-IS DIS for the management LAN. Note that IS-IS routers SHOULD NOT be elected as IS-IS DIS for the management LAN (This is done by setting the DIS Priority of those IS-IS routers to zero). As a result, IS-IS routers would not see each other over the management LAN. In other word, IS-IS routers would not establish adjacencies with one other. Furthermore, IS-IS routers SHOULD send all the types of IS-IS PDUs to the controller being elected as IS-IS DIS as unicasts as well.

To avoid the data traffic from being forwarded across the management LAN, the cost of all IS-IS routers' interfaces to the management LAN SHOULD be set to the maximum value.

When a given IS-IS router lost its connection to the management LAN, it SHOULD actively establish adjacency with all of its IS-IS neighbors within the CLOS network. As such, it could obtain the full LSDB of the CLOS network while flooding its self-originated LSPs to the remaining part of the whole CLOS network through these IS-IS neighbor.

3.2. Controllers as DIS

The controller being elected as IS-IS DIS would send IS-IS PDUs as multicasts or unicasts as before. And it SHOULD accept and process those unicast IS-IS PDUs originated from IS-IS routers. Upon receiving any new LSP from a given IS-IS router, the controller being elected as DIS MUST flood it immediately to the management LAN for two purposes: 1) implicitly acknowledging the receipt of that LSP; 2) synchronizing that LSP to all the other IS-IS routers.

Furthermore, to decrease the frequency of advertising Complete Sequence Number PDU (CSNP) on the controller being elected as DIS, it's RECOMMENDED that IS-IS routers SHOULD send an explicit acknowledgement with a Partial Sequence Number PDU (PSNP) upon receiving a new LSP from the controller being elected as DIS.

4. Acknowledgements

The authors would like to thank Peter Lothberg and Erik Auerswald for his valuable comments and suggestions on this document.

5. IANA Considerations

TBD.

6. Security Considerations

TBD.

7. References

7.1. Normative References

- [RFC1195] Callon, R., "Use of OSI IS-IS for routing in TCP/IP and dual environments", RFC 1195, DOI 10.17487/RFC1195, December 1990, <<https://www.rfc-editor.org/info/rfc1195>>.
- [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/info/rfc2119>>.

7.2. Informative References

[RFC4136] Pillay-Esnault, P., "OSPF Refresh and Flooding Reduction in Stable Topologies", RFC 4136, DOI 10.17487/RFC4136, July 2005, <<https://www.rfc-editor.org/info/rfc4136>>.

Authors' Addresses

Xiaohu Xu
Huawei

Email: xuxiaohu@huawei.com

Luyuan Fang
ebay

Email: lufang@ebay.com

Jeff Tantsura
Individual

Email: jefftant@gmail.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2018

X. Xu
Huawei
L. Fang
ebay
J. Tantsura
Individual
October 30, 2017

OSPF Flooding Reduction in MSDC
draft-xu-ospf-flooding-reduction-in-msdc-02

Abstract

OSPF is commonly used as an underlay routing protocol for MSDC (Massively Scalable Data Center) networks. For a given OSPF router within the CLOS topology, it would receive multiple copies of exactly the same LSA from multiple OSPF neighbors. In addition, two OSPF neighbors may send each other the same LSA simultaneously. The unnecessary link-state information flooding wastes the precious process resource of OSPF routers greatly due to the fact that there are too many OSPF neighbors for each OSPF router within the CLOS topology. This document proposes some extensions to OSPF so as to reduce the OSPF flooding within MSDC networks greatly. The reduction of the OSPF flooding is much beneficial to improve the scalability of MSDC networks. These modifications are applicable to both OSPFv2 and OSPFv3.

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 May 3, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Requirements Language	4
2. Terminology	4
3. Modifications to Current OSPF Behaviors	4
3.1. OSPF Routers as Non-DRs	4
3.2. Controllers as DR/BDR	5
4. Acknowledgements	5
5. IANA Considerations	5
6. Security Considerations	6
7. References	6
7.1. Normative References	6
7.2. Informative References	6
Authors' Addresses	6

1. Introduction

OSPF is commonly used as an underlay routing protocol for Massively Scalable Data Center (MSDC) networks where CLOS is the most popular topology. For a given OSPF router within the CLOS topology, it would receive multiple copies of exactly the same LSA from multiple OSPF neighbors. In addition, two OSPF neighbors may send each other the same LSA simultaneously. The unnecessary link-state information flooding wastes the precious process resource of OSPF routers greatly and therefore OSPF could not scale very well in MSDC networks.

To simplify the network management task, centralized controllers are becoming fundamental network elements in most MSDCs. One or more controllers are usually connected to all routers within the MSDC network via a Local Area Network (LAN) which is dedicated for network management purpose (called management LAN), as shown in Figure 1.

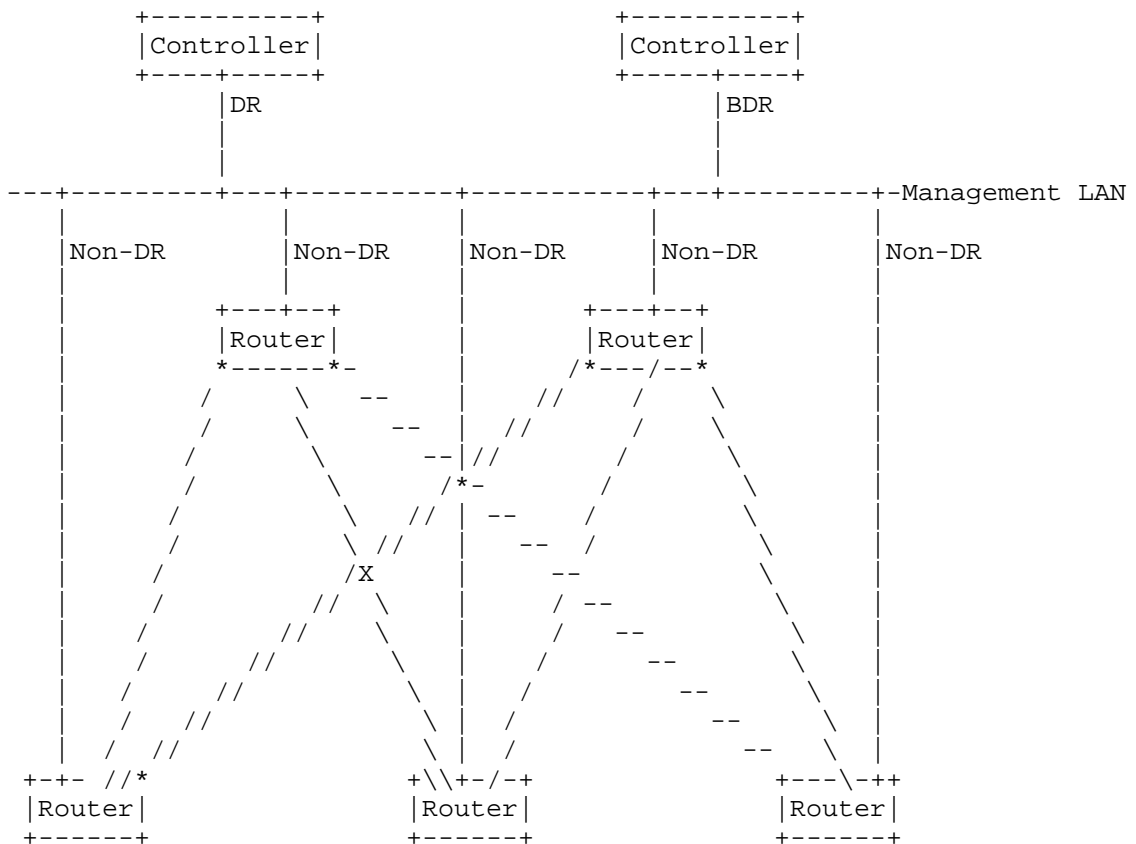


Figure 1

With the assistance of controllers acting as OSPF Designated Router (DR)/Backup Designated Router (BDR) for the management LAN, OSPF routers within the MSDC network don't need to exchange any other types of OSPF packet than the OSPF Hello packet among them. As specified in [RFC2328], these Hello packets are used for the purpose of establishing and maintaining neighbor relationships and ensuring bidirectional communication between OSPF neighbors, and even the DR/BDR election purpose in the case where those OSPF routers are connected to a broadcast network. In order to obtain the full topology information (i.e., the fully synchronized link-state database) of the MSDC's network, these OSPF routers just need to exchange the link-state information with the controllers being elected as OSPF DR/BDR for the management LAN instead.

To further suppress the flooding of multicast OSPF packets originated from OSPF routers over the management LAN, OSPF routers would not

send multicast OSPF Hello packets over the management LAN. Insteads, they just wait for OSPF Hello packets originated from the controllers being elected as OSPF DR/BDR initially. Once OSPF DR/BDR for the management LAN have been discovered, they start to send OSPF Hello packets directly (as unicasts) to OSPF DR/BDR periodically. In addition, OSPF routers would send other types of OSPF packets (e.g., Database Descriptor packet, Link State Request packet, Link State Update packet, Link State Acknowledgment packet) to OSPF DR/BDR for the management LAN as unicasts as well. In contrast, the controllers being elected as OSPF DR/BDR would send OSPF packets as specified in [RFC2328]. As a result, OSPF routers would not receive OSPF packets from one another unless these OSPF packets are forwarded as unknown unicasts over the management LAN. Through the above modifications to the current OSPF router behaviors, the OSPF flooding is greatly reduced, which is much beneficial to improve the scalability of MSDC networks. These modifications are applicable to both OSPFv2 [RFC2328] and OSPFv3 [RFC5340].

Furthermore, the mechanism for OSPF refresh and flooding reduction in stable topologies as described in [RFC4136] could be considered as well.

1.1. 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].

2. Terminology

This memo makes use of the terms defined in [RFC2328].

3. Modifications to Current OSPF Behaviors

3.1. OSPF Routers as Non-DRs

After the exchange of OSPF Hello packets among OSPF routers, the OSPF neighbor relationship among them would transition to and remain in the TWO-WAY state. OSPF routers would originate Router-LSAs and/or Network-LSAs accordingly depending upon the link-types. Note that the neighbors in the TWO-WAY state would be advertised in the Router-LSAs and/or Network-LSA. This is a little bit different from the OSPF router behavior as specified in [RFC2328] where the neighbors in the TWO-WAY state would not be advertised. However, these self-originated LSAs need not to be exchanged directly among them anymore. Instead, these LSAs just need to be sent solely to the controllers being elected as OSPF DR/BDR for the management LAN.

To further reduce the flood of multicast OSPF packets over the management LAN, OSPF routers SHOULD send OSPF packets as unicasts. More specifically, OSPF routers SHOULD send unicast OSPF Hello packets periodically to the controllers being elected as OSPF DR/BDR. In other words, OSPF routers would not send any OSPF Hello packet over the management LAN until they have found OSPF DR/BDR for the management LAN. Note that OSPF routers SHOULD NOT be elected as OSPF DR/BDR for the management LAN (This is done by setting the Router Priority of those OSPF routers to zero). As a result, OSPF routers would not see each other over the management LAN. Furthermore, OSPF routers SHOULD send all other types of OSPF packets than OSPF Hello packets (i.e., Database Descriptor packet, Link State Request packet, Link State Update packet, Link State Acknowledgment packet) to the controllers being elected as OSPF DR/BDR as unicasts as well.

To avoid the data traffic from being forwarded across the management LAN, the cost of all OSPF routers' interfaces to the management LAN SHOULD be set to the maximum value.

When a given OSPF router lost its connection to the management LAN, it SHOULD actively establish FULL adjacency with all of its OSPF neighbors within the CLOS network. As such, it could obtain the full LSDB of the CLOS network while flooding its self-originated LSAs to the remaining part of the whole network. That's to say, for a given OSPF router within the CLOS network, it would not actively establish FULL adjacency with its OSPF neighbor in the TWO-WAY state by default. However, it SHOULD NOT refuse to establish FULL adjacency with a given OSPF neighbors when receiving Database Description Packets from that OSPF neighbor.

3.2. Controllers as DR/BDR

The controllers being elected as OSPF DR/BDR would send OSPF packets as multicasts or unicasts as per [RFC2328]. In addition, Link State Acknowledgment packets are RECOMMENDED to be sent as unicasts rather than multicasts if possible.

4. Acknowledgements

The authors would like to thank Acee Lindem for his valuable comments and suggestions on this document.

5. IANA Considerations

TBD.

6. Security Considerations

TBD.

7. References

7.1. Normative References

- [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/info/rfc2119>>.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <<https://www.rfc-editor.org/info/rfc2328>>.
- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <<https://www.rfc-editor.org/info/rfc5340>>.

7.2. Informative References

- [RFC4136] Pillay-Esnault, P., "OSPF Refresh and Flooding Reduction in Stable Topologies", RFC 4136, DOI 10.17487/RFC4136, July 2005, <<https://www.rfc-editor.org/info/rfc4136>>.
- [RFC5838] Lindem, A., Ed., Mirtorabi, S., Roy, A., Barnes, M., and R. Aggarwal, "Support of Address Families in OSPFv3", RFC 5838, DOI 10.17487/RFC5838, April 2010, <<https://www.rfc-editor.org/info/rfc5838>>.

Authors' Addresses

Xiaohu Xu
Huawei

Email: xuxiaohu@huawei.com

Luyuan Fang
ebay

Email: lufang@ebay.com

Internet-Draft

October 2017

Jeff Tantsura
Individual

Email: jefftant@gmail.com