RIFT
Internet-Draft
Intended status: Standards Track
Expires: 26 August 2021

J. Head, Ed.
T. Przygienda
W. Lin
Juniper Networks
22 February 2021

RIFT Auto-EVPN
draft-head-rift-auto-evpn-00

Abstract

This document specifies procedures that allow an EVPN overlay to be fully and automatically provisioned when using RIFT as underlay and leveraging its no touch ZTP architecture.

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 26 August 2021.

Table of Contents

1. Introduction

RIFT is a protocol that focuses heavily on operational simplicity. [RIFT] natively supports Zero Touch Provisioning (ZTP) functionality that allows each node in an underlay network to automatically derive its place in the topology and configure itself accordingly when properly cabled. RIFT can also disseminate Key-Value information contained in Key-Value Topology Information Elements (KV-TIEs). These KV-TIEs can contain any information and therefore be used for any purpose. Leveraging RIFT to provision EVPN overlays without any need for configuration and leveraging KV capabilities to easily validate correct operation of such overlay without a single point of failure would provide significant benefit to operators in terms of simplicity and robustness of such a solution.

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. Design Considerations

EVPN supports various service models, this document defines a method for the VLAN-Aware service model defined in [RFC7432]. Other service models may be considered in future revisions of this document.

Each model has its own set of requirements for deployment. For example, a functional BGP overlay is necessary to exchange EVPN NLRI regardless of the service model. Furthermore, the requirements are made up of individual variables, such as each node's loopback address and AS number for the BGP session. Some of these variables may be coordinated across each node in a network, but are ultimately locally significant (e.g. route distinguishers). Similarly, calculation of some variables will be local only to each device. RIFT contains currently enough topology information in each node to calculate all those necessary variables automatically.

Once the EVPN overlay is configured and becomes operational KV TIEs can be used to distribute state information to allow for validation of basic operational correctness without need for further tooling.

3. System ID

The 64-bit RIFT System ID that uniquely identifies a node as defined in [RIFT].

4. Fabric ID

RIFT operates on variants of Clos substrate which are commonly called an IP Fabric. Since EVPN VLANs can be either contained within one fabric or span them, Auto-EVPN introduces the concept of a Fabric ID into RIFT.

This section describes an optional extension to LIE packet schema in the form of a 16-bit Fabric ID that identifies a nodes membership within a particular fabric. Auto-EVPN capable nodes MUST support this extension but MAY not advertise it when not participating in Auto-EVPN. A non-present Fabric ID and value of 0 is reserved as ANY_FABRIC and MUST NOT be used for any other purpose.

Fabric ID MUST be considered in existing adjacency FSM rules so nodes that support Auto-EVPN can interoperate with nodes that do not. The LIE validation is extended with following clause and if it is not met, miscabling should be declared:

```
(if fabric_id is not advertised by either node OR
if fabric_id is identical on both nodes)
AND
(if auto_evpn_version is not advertised by either node OR
if auto_evpn_version is identical on both nodes)
```

The appendix details LIE (Appendix A.1.2) and Node-TIE (Appendix A.2.2) schema changes.

5. Auto-EVPN Device Roles

Auto-EVPN requires that each node understand its given role within the scope of the EVPN implementation so each node derives the necessary variables and provides the necessary overlay configuration. For example, a leaf node performing VXLAN gateway functions does not need to derive its own Cluster ID or learn one from the route reflector that it peers with.

5.1. All Participating Nodes

Not all nodes have to participate in Auto-EVPN but when they do they do assume EVPN roles and MUST derive according variables:

IPv6 Loopback Address
: Unique IPv6 loopback address used in BGP sessions.

Router ID
: The BGP Router ID.

Autonomous System Number
: The ASN for IBGP sessions.

Cluster ID
: The Cluster ID for Top-of-Fabric IBGP route reflection.

5.2. ToF Nodes as Route Reflectors

This section defines an Auto-EVPN role whereby some Top-of-Fabric nodes act as EVPN route reflectors. It is expected that route reflectors would establish IBGP sessions with leaf nodes in the same fabric. The typical route reflector requirements do not change, however determining which specific values to use requires further consideration. ToF nodes performing route reflector functionality MUST derive the following variables:

IPv6 RR Loopback Address
: The source address for IBGP sessions with leaf nodes in case ToF won election for one of the route reflectors in the fabric.

IPv6 RR Acceptable Prefix Range
Range of addresses acceptable by the route reflector to form a IBGP session. This range covers ALL possible IPv6 Loopback Addresses derived by other Auto EVPN nodes in the current fabric and other Auto-EVPN RRs addresses.

5.3. Leaf Nodes

Leaf nodes derive their role from realizing they are at the bottom of the fabric, i.e. not having any southbound adjacencies. Alternately, a node can assume a leaf node if it has only southbound adjacencies to nodes with explicit LEAF_LEVEL to allow for scenarios where RIFT leaves do NOT participate in Auto-EVPN.

Leaf nodes MUST derive the following variables:

IPv6 RR Loopback Adresses
Addresses of the RRs present in the fabric. Those addresses are used to build BGP sessions to the RR.

EVIs
Leaf node derives all the necessary variables to instantiate EVIs with layer-2 and optionally layer-3 functionality.

If a leaf node is required to perform layer-2 VXLAN gateway functions, it MUST be capable of deriving the following types of variables:

Route Distinguisher
The route distinguisher corresponding to a MAC-VRF that uniquely identifies each node.

Route Target
The route target that corresponds to a MAC-VRF.

MAC VRF name
This is an optional variable to provide a common MAC VRF name across all leaves.

Set of VLANs
Those are VLANs provisioned either within the fabric or allowing to stretch across fabrics.

For each VLAN derived in an EVI the following variables MUST be derived:

VLAN
The VLAN ID.

name
This is an optional variable to provide a common VLAN name across all leaves.

VNI
The VNI that corresponds to the VLAN ID. This will contribute to the EVPN Type-2 route.

IRB
Optional variables of the IRB for the VLAN if the leaf performs layer-3 gateway function.

If a leaf node is required to perform layer-3 VXLAN gateway functions, it MUST additionally be capable of deriving the following types of variables:

IP Gateway MAC Address
The MAC address associated with IP gateway.

IP Gateway Subnetted Address
The IPv4 and/or IPv6 gateway address including its subnet length.

Type-5 EVPN IP Prefix with ToFs performing gateway functionality can also be derived and will be described in a future version of this document.

6. Auto-EVPN Variable Derivation

As previously mentioned, not all nodes are required to derive all variables in a given network (e.g. a transit spine node may not need to derive any or participate in Auto-EVPN). Additionally, all derived variables are derived from RIFT's FSM or ZTP mechanism so no additional flooding beside RIFT flooding is necessary for the functionality.

It is also important to mention that all variable derivation is in some way based on combinations of System ID, MAC-VRF ID, Fabric ID, EVI and VLAN and MUST comply precisely with calculation methods specified in the Appendix section to allow interoperability between different implementations.

6.1. Auto-EVPN Version

This section describes extensions to both the RIFT LIE packet and Node-TIE schemas in the form of a 16-bit value that identifies the Auto-EVPN Version. Auto-EVPN capable nodes MUST support this extension, but MAY choose not to advertise it in LIEs and Node-TIEs when Auto-EVPN is not being utilized. The appendix describes LIE (Appendix A.1.1) and Node-TIE (Appendix A.2.1) schema changes in detail.

6.2. MAC-VRF ID

This section describes a variable MAC-VRF ID that uniquely identifies an instance of EVPN instance (EVI) and is used in variable derivation procedures. Each EVPN EVI MUST be associated with a unique MAC-VRF ID, this document does not specify a method for making that association or ensuring that they are coordinated properly across fabric(s).

6.3. Loopback Address

First and foremost, RIFT does not advertise anything more specific than the fabric default route in the southbound direction by default. However, Auto-EVPN nodes MUST advertise specific loopback addresses southbound to all other Auto-EVPN nodes so to establish MP-BGP reachability correctly in all scenarios.

Auto-EVPN nodes MUST derive a ULA-scoped IPv6 loopback address to be used as both the IBGP source address, as well as the VTEP source when VXLAN gateways are required. Calculation is done using the 6-bytes of reserved ULA space, the 2-byte Fabric ID, and the node's 8-byte System ID. Derivation of the System ID varies slightly depending upon the node's location/role in the fabric and will be described in subsequent sections.

IPv4 addresses MAY be supported, but it should be noted that they have a higher likelihood of collision.

The required algorithm can be found in the appendix (Appendix A.3.3).

6.3.1. Leaf Nodes as Gateways

Calculation is done using the 6-bytes of reserved ULA space, the 2-byte Fabric ID, and the node's 8-byte System ID.

6.3.2. ToF Nodes as Route Reflectors

ToF nodes acting as route reflectors MUST derive their loopback address according to the specific section describing the algorithm. Calculation is done using the 6-bytes of reserved ULA space, the 2-byte Fabric ID, and the 8-byte System ID of each elected route reflector.

6.3.2.1. Route Reflector Election Procedures

Four Top-of-Fabric nodes MUST be elected as an IBGP route reflector. Each ToF performs the election independently based on system IDs of other ToFs in the fabric obtained via southbound reflection. The route reflector election procedures are defined as follows:

1. ToF node with the highest System ID.

2. ToF node with the lowest System ID.

3. ToF node with the 2nd highest System ID.

4. ToF node with the 2nd lowest System ID.

This ordering is necessary to prevent a single node with either the highest or lowest System ID from triggering changes to route reflector loopback addresses as it would result in all BGP sessions dropping.

For example, if two nodes, ToF01 and ToF02 with System IDs 002c6af5a281c000 and 002c6bf5788fc000 respectively, ToF02 would be elected due to it having the highest System ID of the ToFs (002c6bf5788fc000). If a ToF determines that it is elected as route reflector, it uses the knowledge of its position in the list to derive route reflector v6 loopback address.

Considerations for multiplane route reflector elections will be included in future revisions.

6.4. Autonomous System Number

Nodes in each fabric MUST derive a private autonomous system number based on its Fabric ID so that it is unique across the fabric.

The required algorithm for 2-byte ASNs can be found in the appendix (Appendix A.3.4).

6.5. Cluster ID

Route reflector nodes in each fabric MUST derive a cluster ID that is based on its Fabric ID so that it is unique across the fabric. Implementations MAY choose to simply use the AS number as the cluster ID.

The required algorithm can be found in the appendix (Appendix A.3.5).

6.6. Router ID

Nodes MUST drive a Router ID that is based on both its System ID and Fabric ID so that it is unique to both.

The required algorithm can be found in the appendix (Appendix A.3.6).

6.7. Route Target

Nodes hosting EVPN EVIs MUST derive a route target extended community based on the MAC-VRF ID for each EVI so that it is unique across the network. Route targets MUST be of type 0 as per RFC4360.

For example, if given a MAC-VRF ID of 1, the derived route target would be "target:1"

The required algorithm can be found in the appendix (Appendix A.3.7).

6.8. Route Distinguisher

Nodes hosting EVPN EVIs MUST derive a type-0 route distinguisher based on its System ID and Fabric ID so that it is unique per MAC-VRF and per node.

The required algorithm can be found in the appendix (Appendix A.3.8).

6.9. EVPN MAC-VRF Services

It’s obvious that applications utilizing Auto-EVPN overlay services may require a variety of layer-2 and/or layer-3 traffic considerations. Variables supporting these services are also derived based on some combination of MAC-VRF ID, Fabric ID, and other constant values. Integrated Routing and Bridging (IRB) gateway address derivation also leverages a set of constant "random seed" values to provide additional entropy.

The required derivation procedures can be found in the appendix (Appendix A.3).

6.9.1. Untagged Traffic in Multiple Fabrics

This section defines a methods to derive unique VLAN, VNI, MAC, and gateway address values for deployments where untagged traffic is stretched across multiple fabrics.

6.9.1.1. VLAN

Untagged traffic stretched across multiple fabrics MUST derive VLAN tags based on MAC-VRF ID in conjunction with a constant value of 1 (i.e. MAC-VRF ID + 1).

6.9.1.2. VNI

Untagged traffic stretched across multiple fabrics MUST derive VNIs based on MAC-VRF ID and Fabric ID in conjunction with a constant value. These VNIs MUST correspond to EVPN Type-2 routes.

6.9.1.3. MAC Address

The MAC address MUST be a unicast address and also MUST be identical for any IRB gateways that belong to an individual bridge-domain across fabrics. The last 5-bytes MUST be a hash of the MAC-VRF ID and a constant value of 1 that is calculated using the previously mentioned random seed values.

6.9.1.4. IPv6 IRB Gateway Address

The derived IPv6 gateway address MUST be from a ULA-scoped range that will account for the first 6-bytes. The next 5-bytes MUST be the last bytes of the derived MAC address. Finally, the remaining 7-bytes MUST be ::0001.

6.9.1.5. IPv4 IRB Gateway Address

The derived IPv4 gateway address MUST be from a RFC1918 range, which accounts for the first octet. The next octet MUST a hash of the MAC-VRF ID and a constant value of 1 that is calculated using the previously mentioned random seed values. Finally, the remaining 2 octets MUST be 0 and 1 respectively.

6.9.2. Tagged Traffic in Multiple Fabrics

This section defines a methods to derive unique VLAN, VNI, MAC, and gateway address values for deployments where tagged traffic is stretched across multiple fabrics.

6.9.2.1. VLAN

Tagged traffic stretched across multiple fabrics MUST derive VLAN tags based on MAC-VRF ID in conjunction with a constant value of 16 (i.e. MAC-VRF ID + 16).

6.9.2.2. VNI

Tagged traffic stretched across multiple fabrics MUST derive VNIs based on MAC-VRF ID and Fabric ID in conjunction with a constant value. These VNIs MUST correspond to EVPN Type-2 routes.

6.9.2.3. MAC Address

The MAC address MUST be a unicast address and also MUST be identical for any IRB gateways that belong to an individual bridge-domain across fabrics. The last 5-bytes MUST be a hash of the MAC-VRF ID and a constant value of 1 that is calculated using the previously mentioned random seed values.

6.9.2.4. IPv6 IRB Gateway Address

The derived IPv6 gateway address MUST be from a ULA-scoped range that will account for the first 6-bytes. The next 5-bytes MUST be the last bytes of the derived MAC address. Finally, the remaining 7-bytes MUST be ::0001.

6.9.2.5. IPv4 IRB Gateway Address

The derived IPv4 gateway address MUST be from a RFC1918 range, which accounts for the first octet. The next octet MUST a hash of the MAC-VRF ID and a constant value of 16 that is calculated using the previously mentioned random seed values. Finally, the remaining 2 octets MUST be 0 and 1 respectively.

6.9.3. Tagged Traffic in a Single Fabric

This section defines a methods to derive unique VLAN, VNI, MAC, and gateway address values for deployments where untagged traffic is contained within a single fabric.

6.9.3.1. VLAN

Tagged traffic contained to a single fabric MUST derive VLAN tags based on MAC-VRF ID and Fabric ID in conjunction with a constant value of 17 (i.e. MAC-VRF ID + Fabric ID + 17).

6.9.3.2. VNI

Tagged traffic contained to a single fabric MUST derive VNIs based on MAC-VRF ID and Fabric ID in conjunction with a constant value. These VNIs MUST correspond to EVPN Type-2 routes.

6.9.3.3. MAC Address

The MAC address MUST be a unicast address and also MUST be identical for any IRB gateways that belong to an individual bridge-domain across fabrics. The last 5-bytes MUST be a hash of the MAC-VRF ID and a constant value of 1 that is calculated using the previously mentioned random seed values.

6.9.3.4. IPv6 IRB Gateway Address

The derived IPv6 gateway address MUST be from a ULA-scoped range, which accounts for the first 6-bytes. The next 5-bytes MUST be the last bytes of the derived MAC address. Finally, the remaining 7-bytes MUST be ::0001.

6.9.3.5. IPv4 IRB Gateway Address

The derived IPv4 gateway address MUST be from a RFC1918 range, which accounts for the first octet. The next octet MUST a hash of the MAC-VRF ID and a constant value of 17 that is calculated using the previously mentioned random seed values. Finally, the remaining 2 octets MUST be 0 and 1 respectively.

6.9.4. Traffic Routed to External Destinations

6.9.4.1. Route Distinguisher

Nodes hosting IP Prefix routes MUST derive a type-0 route distinguisher based on its System ID and Fabric ID so that it is unique per IP-VRF and per node.

The required algorithm can be found in the appendix (Appendix A.3.8).

6.9.4.2. Route Target

Nodes hosting IP prefix routes MUST derive a route target extended community based on the MAC-VRF ID for each IP-VRF so that it is unique across the network. Route targets MUST be of type 0.

The required algorithm can be found in the appendix (Appendix A.3.7).

7. Acknowledgements

TBD

8. Security Considerations

This document introduces no new security concerns to RIFT or other specifications referenced in this document.

9. References

9.1. 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>.

[RFC7432] Sajassi, A., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", February 2015, <https://www.rfc-editor.org/info/rfc7432>.

[RIFT] Przygienda, T., Sharma, A., Thubert, P., Rijsman, B., and D. Afanasiev, "RIFT: Routing in Fat Trees", Work in Progress, draft-ietf-rift-rift-12, May 2020.

Appendix A. Appendix

A.1. RIFT LIE Schema

A.1.1. Auto-EVPN Version

```
struct LIEPacket {
 ...
   /** It provides the optional ID of the configured fabric  */
   25: optional common.FabricIDType       fabric_id;
...
```

A.1.2. Fabric ID

```
...
struct LIEPacket {
 ...
   /** It provides optional version of EVPN ZTP as 256 * MAJOR + MINOR */
   26: optional i16                       auto_evpn_version;
...
```

A.2. RIFT Node-TIE Schema

A.2.1. Auto-EVPN Version

```
struct NodeTIEElement {
...
   /** It provides optional version of EVPN ZTP as 256 * MAJOR + MINOR */
   13: optional i16                          auto_evpn_version;
```

A.2.2. Fabric ID

```
 struct NodeTIEElement {
 ...
   /** It provides the optional ID of the Fabric configured */
   12: optional common.FabricIDType         fabric_id;
```

A.3. Variable Derivation

A.3.1. Random Seed Values

To be provided in future version of this document.

A.3.2. Fabric ID

To be provided in future version of this document.

A.3.3. Loopback Address

To be provided in future version of this document.

A.3.4. Autonomous System Number

To be provided in future version of this document.

A.3.5. Cluster ID

To be provided in future version of this document.

A.3.6. Router ID

To be provided in future version of this document.

A.3.7. Route Target

To be provided in future version of this document.

A.3.8. Route Distinguisher

To be provided in future version of this document.

A.3.9. VLAN

To be provided in future version of this document.

A.3.10. VNI

To be provided in future version of this document.

A.3.11. Gateway (MAC)

To be provided in future version of this document.

A.3.12. Gateway (IPv6)

To be provided in future version of this document.

A.3.13. Gateway (IPv4)

To be provided in future version of this document.

Authors' Addresses

Jordan Head (editor)
Juniper Networks
1137 Innovation Way
Sunnyvale, CA
United States of America

Email: jhead@juniper.net

Tony Przygienda
Juniper Networks
1137 Innovation Way
Sunnyvale, CA
United States of America

Email: prz@juniper.net

Wen Lin
Juniper Networks
10 Technology Park Drive
Westford, MA
United States of America

Email: wlin@juniper.net