SRIFT: Segment Routing In Fat Trees
draft-zzhang-rift-sr-01

Abstract

This document specifies signaling procedures for Segment Routing
[RFC8402] with RIFT. Each node’s loopback address, Segment Routing
Global Block (SRGB) and Node Segment Identifier (SID), which must be
unique within the SR domain and are typically assigned by SR
controllers or management, are distributed southbound from the Top Of
Fabric (TOF) nodes via the Key-Value distribution mechanism, so that
each node can compute how to reach a node represented by the topmost
Node SIDs. For an ingress node to send SR traffic to another node
via an explicit path, an SR controller signals the corresponding
label stack to the ingress node so that the ingress node can send
packets accordingly.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in RFC2119.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering
Task Force (IETF). Note that other groups may also distribute
working documents as Internet-Drafts. The list of current Internet-
Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months
and may be updated, replaced, or obsoleted by other documents at any
time. It is inappropriate to use Internet-Drafts as reference
material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 27, 2019.
Before we discuss the SR procedures for RIFT, let us first review how SR works with OSPF/ISIS [I-D.ietf-ospf-segment-routing-extensions] [I-D.ietf-isis-segment-routing-extensions].

Each node is provisioned with a loopback address, an SRGB, and a Node SID. The loopback address and Node SID are co-ordinated centrally - it is unique for each node across the SR network - and then communicated out of band to each node and stored as configuration information. For example, the out of band communication could be via primitive pen and paper, or modern signaling from controllers.

SRGB configuration can be local to each node and different node can have the same or different label blocks for flexible label allocation. Typically, modern SR networks have identical SRGB on each node so that a Node SID corresponds to the same label on each node. However that is not mandatory. Either way, the SRGB is part of the node’s local configuration. In today’s network it is very likely pushed down from some controllers.
Each node will then signal its SRGB, and its Node SID. The Node SID is just an index into the SRGB and other nodes will derive the corresponding label for each advertised Node SID. Consider the following example:

```
A               D
*               *
   *     *
   *     *
   *     *
   *     *
   *     *
   *     *
B

```

<table>
<thead>
<tr>
<th>Node Name</th>
<th>Loopback</th>
<th>Node SID</th>
<th>SRGB Label Base</th>
<th>SRGB Label Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10.1.1.1</td>
<td>1</td>
<td>100</td>
<td>50</td>
</tr>
<tr>
<td>B</td>
<td>10.1.1.2</td>
<td>2</td>
<td>100</td>
<td>50</td>
</tr>
<tr>
<td>C</td>
<td>10.1.1.3</td>
<td>3</td>
<td>200</td>
<td>50</td>
</tr>
<tr>
<td>D</td>
<td>10.1.1.4</td>
<td>4</td>
<td>100</td>
<td>50</td>
</tr>
</tbody>
</table>

Node A computes its IP and label routes as following:

```
<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1.1.1</td>
<td>local</td>
</tr>
<tr>
<td>10.1.1.2</td>
<td>if_ab</td>
</tr>
<tr>
<td>10.1.1.3</td>
<td>if_ac</td>
</tr>
<tr>
<td>10.1.1.4</td>
<td>if_ab, if_ac</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Label</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 (La_a)</td>
<td>pop and look up next header</td>
</tr>
<tr>
<td>101 (Lb_a)</td>
<td>swap to 101, send out of if_ab</td>
</tr>
<tr>
<td>102 (Lc_a)</td>
<td>swap to 202, send out of if_ac</td>
</tr>
<tr>
<td>103 (Ld_a)</td>
<td>swap to 103, send out of if_ab, swap to 203, send out of if_ac</td>
</tr>
</tbody>
</table>
```

For example, Node A computes the route to node D (represented by its loopback address) and the next hops are node B via interface if_ab and node C via if_ac. A uses D’s Node SID (advertised along with D’s loopback address), and index it into its own SRGB to obtain label.
Ld_a (103). It also uses D’s Node SID to index into B’s and C’s SRGBs to obtain label Ld_b (103) and Ld_c (203) respectively. Now it programs its label forwarding state with (Ld_a --> (outgoing if_ab swap to Ld_b, outgoing if_ac swap to Ld_c)). Notice that Ld_a, Ld_b and Ld_c are most likely the same though not necessarily so.

Node B computes the route to D and finds that the next hop is node D itself via interface if_bd. It use D’s Node SID (advertised along with D’s loopback address) and index it into its own SRGB and obtain label Ld_b (103). It also uses the SID and index it into D’s SRGB to obtain label Ld_d (103). Now it programs its label forwarding state with (Ld_b->outgoing if_bd swap to label Ld_d).

Similarly, D programs a label forwarding state (Ld_d->pop and lookup next header).

It is clear that the following needs to happen:

- Each node’s SRGB needs to be signalled to all other adjacent nodes
- Each node’s Node SID needs to be signalled to all other nodes
- Each node’s loopback address needs to be signalled to all other nodes

With ISIS/OSPF, each node’s SRGB is actually flooded everywhere for simplicity. With RIFT, North TIEs are flooded all the way north but South TIEs are only flooded one hop south (and then reflected one hop north). While the Node TIEs could be used to flood SRGBs, each node would need to learn its own SRGB first. With RIFT ZTP, the TOF nodes learn the SRGB and Node SID provisioning for every node (from SR controllers) and flood them southbound via K-V distribution - there is no need to flood SRGB via Node TIEs any more.

For the purpose of SR-TE, a label stack is used - each entry in the stack represents a node on the TE path towards the destination. Consider the following 4-level topology:

```
TOF1          TOF2
Spine1_11    Spine1_21    Spine1_21    Spine1_22
Spine2_11    Spine2_21    Spine2_21    Spine2_22
Leaf11       Leaf12       Leaf21       Leaf22
```
Say the TE controller instructs Leaf11 to send a packet to Spine2_11 with label stack (Label_TOF2, Label_Spine2_21, Label_Leaf21). Spine2_11 needs to recognize that Label_TOF2 maps to node TOF2 and it should not simply follow the default route (because the default route could lead to an unintended path via TOF1). In other words, each node needs to have a specific route to every node in the north. That means for RIFT that the southbound distance vector routing needs to additionally advertise routes for loopback address of the nodes in the north. Each node originates a route for its own loopback address and advertises it southbound, with a special marking that allows a south node to re-advertise it further south.

As an alternative, routes for loopback addresses may be replaced by "routes" for SystemIDs. This is for further study in future revisions.

Considerations for Prefix SIDs will be provided in future revisions.

2. Specifications

This document defines the following Key-Value for SR purpose, in the form of (key type, key value, value). The key type is SR, The key value is the loopback address, and the value is a (SRGB label base, SRGB label range, Node SID) tuple. This also assumes that each node learns its own loopback address somehow, probably through another KV (given the ZTP consideration).

More details will be specified in future revisions.

3. Security Considerations

To be provided.

4. Acknowledgements

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

5. References

5.1. Normative References

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

[I-D.ietf-isis-segment-routing-extensions]

[I-D.ietf-ospf-segment-routing-extensions]


Authors’ Addresses

Zhaohui Zhang
Juniper Networks
EMail: zzhang@juniper.net

Jeff Tantsura
Apstra, Inc
EMail: jefftant.ietf@gmail.com

Don Fedyk
Individual
EMail: don.fedyk@gmail.com