MPLS Segment Routing in IP Networks

draft-bryant-mpls-unified-ip-sr
Stewart Bryant <stewart.bryant@gmail.com>
Adrian Farrel <afarrel@juniper.net>
John Drake <jdrake@juniper.net>
Jeff Tantsura <jefftant.ietf@gmail.com>

IETF-100, Singapore, November 2017
Objective
Objectives

1. Tunnel MPLS-SR over an IP network
   - To connect two MPLS-SR networks (e.g., data centres)
2. Enable SR in legacy networks by tactically introducing SR-capable nodes at strategic points in the network.
3. It is not a specific objective, but the approach is IPv4/v6 neutral.
Use Cases

SR Domain

SR-in-UDP Tunnel

SR Domain

Border Router

IP Network

Border Router

Legacy IP Router

SR-Capable Router Not In SID Stack

Native IP Forwarding

SR-Capable Router In SID Stack
Technical Summary
Overview

• In summary, this is MPLS-over-UDP as RFC 7510

  • Encapsulate a “normal” MPLS SID stack in UDP in IP
  • Address to next SR-capable node in the SR path
  • UDP destination port indicates “MPLS below”
MPLS-SR-in-UDP Processing

• IGP and control plane just like MPLS-SR
• Source processing is just like MPLS-SR
  – But encapsulate in UDP and IP to first router identified by first SID
• Legacy transit nodes
  – It is just an IP packet, so simply forward it
• SR-capable transit nodes
  – Process MPLS-SID stack as normal
  – Encapsulate in UDP and IP and send to router identified by next SID
• Final hop just strips outer header and forwards payload packet
A Little More Detail

- **IP Header**
  - Src = Sending SR capable node
  - Dst = Next SR capable node
  - Next protocol = UDP

- **UDP Header**
  - Src Port = Entropy
  - Dst Port = MPLS-in-UDP

- **MPLS SIDs Label Stack**
  - Stack of SIDs exactly like it is an MPLS SR packet

- **IP Header**
  - Unchanged IP header and data i.e., encapsulated packet
Advertising SIDs

• Advertisements are just like for MPLS SR
  – IGP or BGP advertises
    • Address of node or link
    • Associated SID
  – All SID types are supported

• Need to add advertisements in routing protocol to specify
  – Encapsulation Type
  – PHP behaviour
Source Processing

• Build and impose MPLS-SR stack
• Encapsulate in UDP
  – Dst Port = MPLS-in-UDP
  – Src Port = Entropy
• Encapsulate in IP with
  – Source as this node
  – Destination address of first hop in SR stack
    • Requires look-up to match SID to address
    • At source, this lookup can use RIB, etc.
• FIB lookup and send
• (This all looks a lot like RFC 7510)
Transit Non-SR Processing

• Important that this mechanism can traverse nodes that are not SR-capable
  – Also, no special processing by SR nodes to which the packet is not addressed
• It’s just an IP packet, so forward it
• ECMP entropy is achieved through the UDP source port value set by source
  – Established technique (RFC 7510)
• TTL decrements as usual
Transit SR Processing

• If the packet is addressed to me
  – Otherwise just forward the packet as normal IP
• Find UDP inside
• Find UDP Dst port is “MPLS-in-UDP”
• Look at top of MPLS SR stack
  – Extract SID and look up “next hop” IP address
  – Pop label stack entry
• Re-encapsulate packet as MPLS-in-UDP-in-IP (just as source did)
  – IP Src = this node
  – IP Dst = next address as found from label lookup
  – UDP Src Port = Entropy (ideally from received packet)
  – UDP Dst Port = MPLS-in-UDP
  – (SID stack is “shorter”)
• FIB lookup and send
Key Changes from -00

• Forwarding clarifications
  – Transit node elements of procedure.
  – Entropy handling.
  – PHP processing described in detail.
  – Clarification of egress processing.
  – Processing of an erroneously received packet described.

• Control Plane
  – A summary of the existing control planes and extension required to support PHP is provided.
Moving Forward
Additional Encapsulation Technologies

• This draft describes the use of UDP as the encapsulation.

• There may be a need for other encapsulations (VXLAN, GRE, IPSEC etc.)

• If there is such a need, the right approach is to write up a data-plane specific solution for each of these cases in separate RFCs as and when there is an established need for that encapsulation type.
Control Plane Separation

• The control plane solutions should be written up independently of the data-plane.
• The description of each control plane should specify the encapsulation technology as a parameter and thus be usable in configuring future encapsulation technologies as and when they become important and are documented.
Proposal

• This is a simple solution to a simple problem.
• It would be useful to have the problem discussed in the SPRING WG and have experts there flag up concerns and missing functions.
• In our view the solution to this problem belongs in the MPLS WG.