YANG Data Models MPLS Base and Static LSPs

draft-ietf-mpls-base-yang-04
draft-ietf-mpls-static-yang-04
code @ https://github.com/ietf-mpls-yang/te

Tarek Saad, Kamran Raza, and Rakesh Gandhi, Cisco Systems
Vishnu Pavan Beeram, Juniper Networks
Xufeng Liu, Jabil
Igor Bryskin, Huawei
Himanshu Shah, Ciena
R. Jones, Brocade
B. Wen, Comcast

IETF-99, July 2017, Prague
Agenda

• Updates to I-Ds (since IETF98)
• Open issues
• Next steps
Summary of Changes

• Migrated MPLS label stack from “leaf-list” to “list” with “key index”

• Met with authors of “draft-acee-rtgwg-yang-rib-extend”:
  – Discussed moving per route attribute extensions from MPLS YANG model to RIB-EXT model
  – Discussed existing RIB’s model per-AF next-hop address limitation
Update # 1

MPLS label stack modelling (list ordering)

Label Index:
- distinguisher to allow repetition of same label in the stack
- can be used to derive the “offset” per entry in label stack – e.g. index == offset?

List ordering modes:
- Ordered-by-user – User defined order:
  - order of appearance in configuration should be same as that in label stack (top to bottom)
  - can use insert/delete before/after, top, etc.
- Ordered-by-system (default):
  - order of appearance in the configuration list is not important
  - backend sorts/generates label offset in stack from label entries
  - backend may use list index as label offset "as-is"
    - if so, need to validate presence of n-1 before accepting n
    - if not, recomputes label offset from sorted label list entries
    - if not, backend needs to resort/recompute offset everytime a new entry is added/removed from the list
Update # 2
MPLS label stack modelling (list ordering)

Label Index:
- distinguisher to allow repetition of same label in the stack
- can be used to derive the “offset” per entry in label stack – e.g. index == offset?

List ordering modes:
- **Ordered-by-user – User defined order:**
  - order of appearance in configuration should be same as that in label stack (top to bottom)
  - can use insert/delete before/after, top, etc.
- **Ordered-by-system (default):**
  - order of appearance in the configuration list is not important
  - backend sorts/generates label offset in stack from label entries
  - backend may use list index as label offset “as-is”
    - if so, need to validate presence of n-1 before accepting n
    - if not, recomputes label offset from sorted label list entries
    - if not, backend needs to resort/recompute offset everytime a new entry is added/removed from the list
Open Issue # 1
MPLS label encoding

Currently MPLS-Static model uses

- list of rt-types:mpls-label values only

ietf-routing-type defines label stack as:
- label, ttl, traffic-class
- allows setting TC per entry in stack at ingress
  - TC -> PHB on LSRs
  - rules defined in RFC3270 (short/Uniform) Pipe model
- allows setting TTL per entry in stack on ingress
  - rules defined in RFC3443 for short/uniform Pipe models (section 3.4)
  - should this show in state (if so, is it iTTL, outTTL?)

- Should same grouping be used in MPLS/RIB state too?
  - Usually routes appearing in RIB (installed by signaling protocols e.g. LDP, BGP, RSVP) will not set TC, TTL, etc.
Open Issue # 2
Use extended RIB route attributes

• Migrate to using the newly introduced RIB route extended model < draft-acee-rtgwg-yang-rib-extend>
  for repair-path attributes
  • originally defined in MPLS model but applicable to routes for other Afs
  • Follow-up on missing per-path attributes (e.g. loadshare)
Open Issue #3:
Migration to NMDA style

- Migrate from OC-style to the recommended NMDA style
Next Steps

• Address the open issues

• Request further review and comments on the models
Thank You