RSVP-TE Usability at scale

Ravi Singh (ravis@juniper.net)

NANOG-64 [San Francisco]

Agenda

- RSVP-TE: Quick recap
- RSVP-TE: Key operational problems
- RSVP-TE: Ways to address these operational problems

RSVP-TE: Quick recap

- A signaling protocol to setup MPLS LSPs
- Widely deployed
- Enables
  - Traffic engineering
  - Bandwidth accounting
  - Fast failure protection (fast reroute): local-protection + global-repair

- Increase in network-size has led to increased LSP scale
- Scale magnifies operational issues

RSVP-TE: Key operational problems

- Configuration challenges:
  - How to ease the need to configure N^2 LSPs between N routers?

- Monitoring LSPs at scale - LSP telemetry:
  - What events are happening on the LSPs? Which ones need operator attention?
  - What properties of the LSPs are changing?

- Auto-bandwidth deployment issues:
  - How to improve “appropriateness” of resizing of the LSPs?

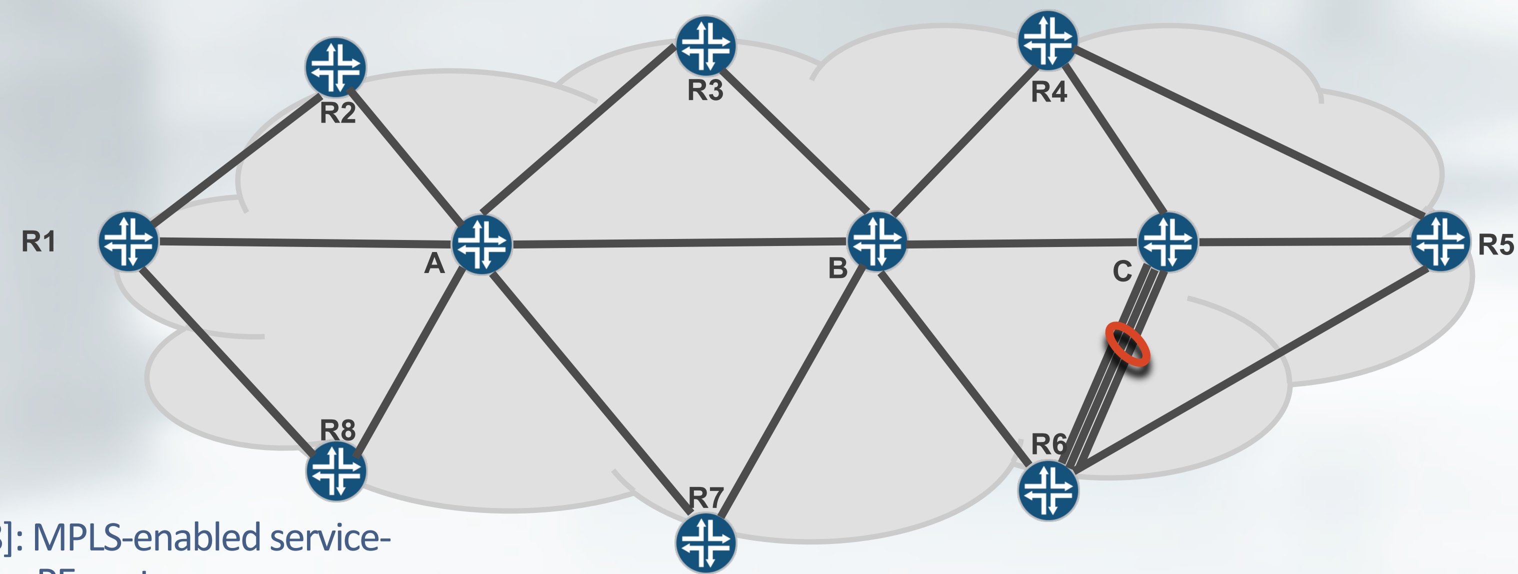
RSVP-TE: Key operational problems (cont....)

- Maximizing n/w utilization: Multi-path and load-balancing:
  - How to more effectively increase network utilization by spreading the load across the network?
  - How to load-balance traffic better?

- Data losses during LSP re-optimization:
  - How to prevent data losses on an LSP caused by a router advertising control-plane readiness before its data-plane is ready?

- Reducing network churn:
  - How to prevent avoidable re-signaling of LSPs?

Exemplar Network



R [1-8]: MPLS-enabled service-hosting PE routers

A,B,C: transit routers

Configuration challenges

- How to ease the need to configure N^2 LSPs between N routers?
  - Key: Not having to configure each of the N^2 LSPs:

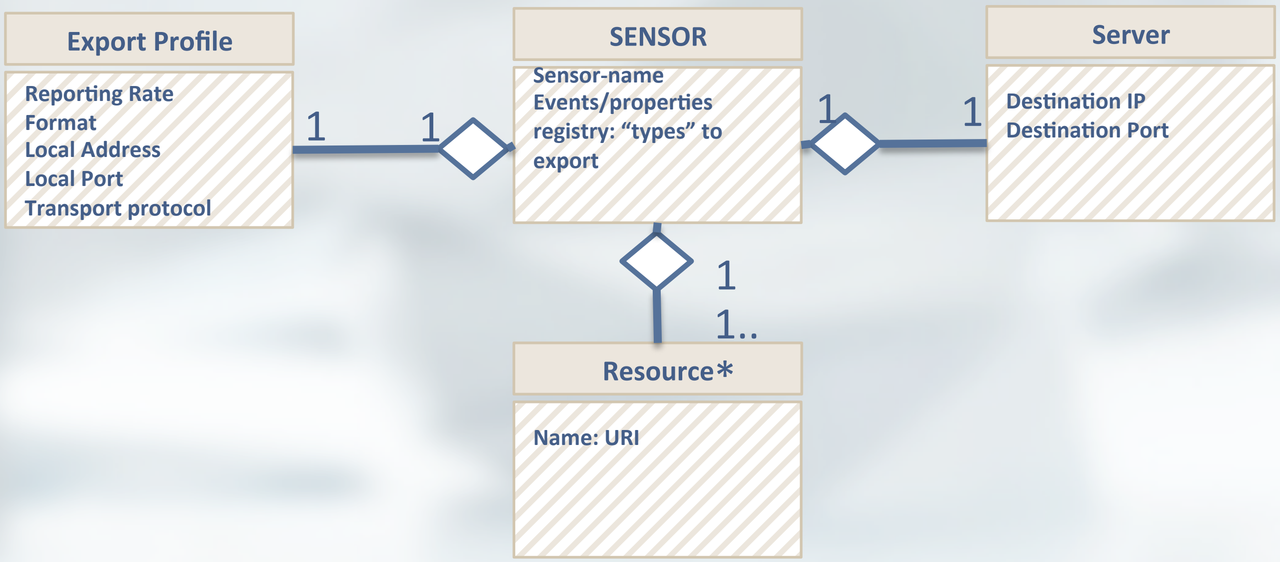
- RSVP auto-mesh:
  - Configuration template to specify the general characteristics of multiple (perhaps, all) LSPs originating at this router
  - No need to configure individual LSPs
  - Mechanism relies on "on-demand" LSP creation by availability of BGP routes the protocol-nexthop for which does not already have an LSP created to it

Monitoring LSPs at scale : LSP telemetry

- What events are happening on the LSPs? Which ones need operator attention? What properties of the LSPs are changing?
  - Key: Not having to rely on a polling mechanism (like SNMP) to poll LSPs' health

- A push-based approach to export LSP events/properties to an off-router client:
  - Transmitter (on router) transmits in a message created by <u>code generated</u> off a message-template
  - Collector relies on same message-template to <u>generate code</u> using which parse the received information
  - Collector: created by router vendor or by operator

LSP Telemetry: Object model



LSP **sensor:** basic unit of LSP- telemetry.
Tracks events/properties to export.
Export as per the **Export Profile**.

Auto-bandwidth deployment issues

- How to improve “appropriateness” of resizing of the LSPs?
  - Key: When making resizing decision, enable resize computation by:
    - Efficient/recent collection of LSP data plane statistics
    - Allowing programmable analytics to determine “appropriate” size

- Less expensive data-plane LSP statistics gathering: LSP-statistics telemetry
  - PUSH raw line-card stats without explicit polling or aggregation

- Statistics collector runs analytics algorithms to compute new LSP size
- Statistics collector may reside on:
  - Router’s control plane, or
  - External server

Maximizing n/w utilization

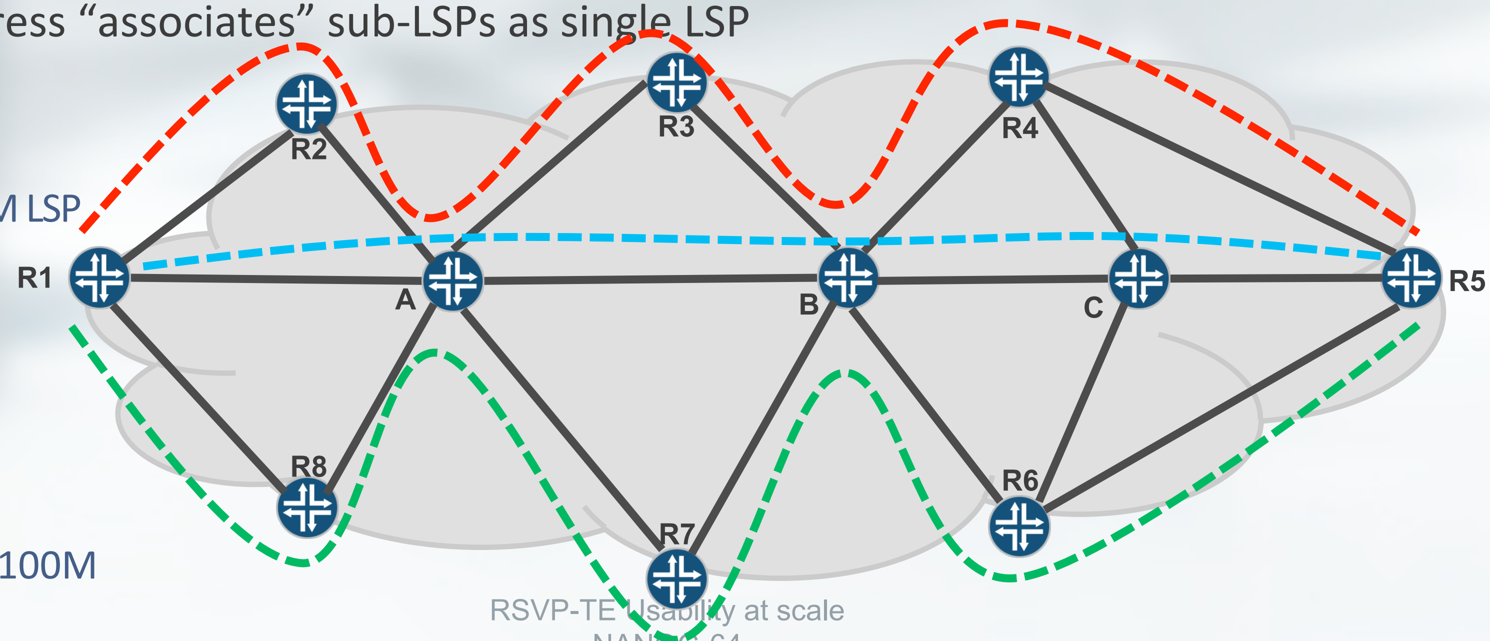
- How to
  - More effectively increase network utilization by spreading the load across the network?
  - Load-balance traffic better?

- Key: effectively spread out traffic for the same LSP over different links

Maximizing n/w utilization: Multi-path

- TE++:
  - Splitting (at ingress/transit) traffic of a single LSP into sub-LSPs that merge
  - Ingress "associates" sub-LSPs as single LSP

R1->R5: A 300M LSP



Every link is 100M

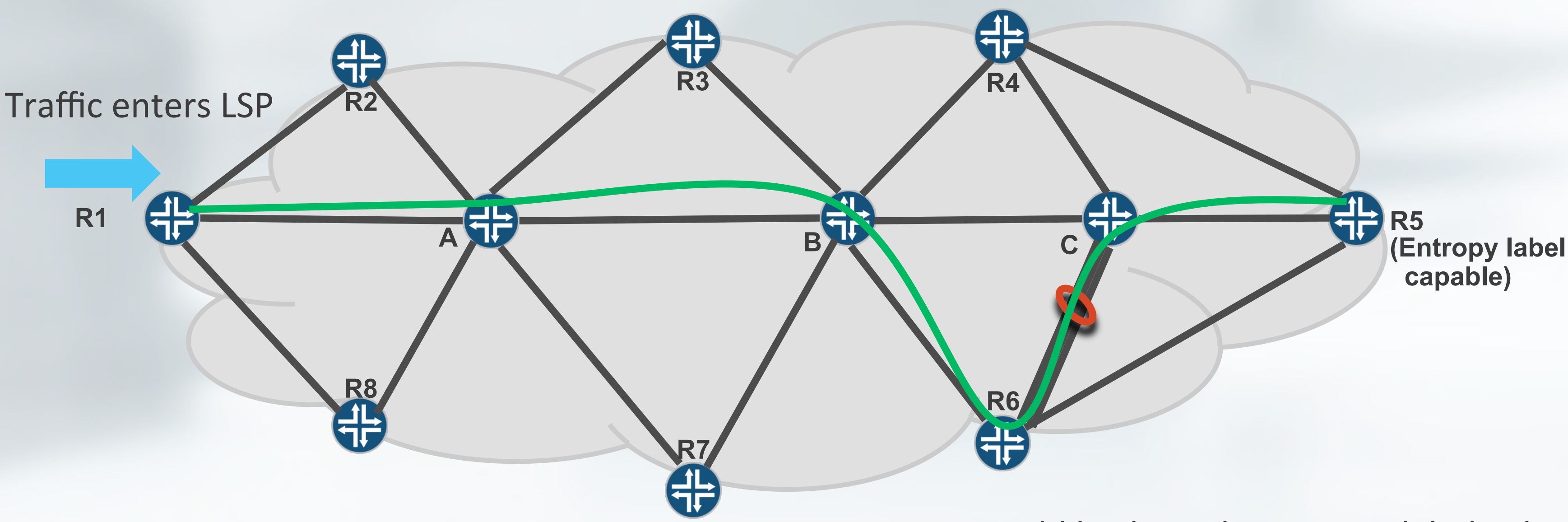
Maximizing n/w utilization: Load-balancing:

- Entropy label:
  - Enables transit router to compute load-balancing hashes without deeper packet inspection
  - Usable even when not every ingress/egress LER pair supports entropy label
  - Ingress <u>computes</u> & inserts entropy label by hashing on flow-identifying header fields

LSP Label	ELI (Label 7)	Entropy Label	IP Payload

EL inside an MPLS packet carrying an IP payload

Load-balancing using entropy label



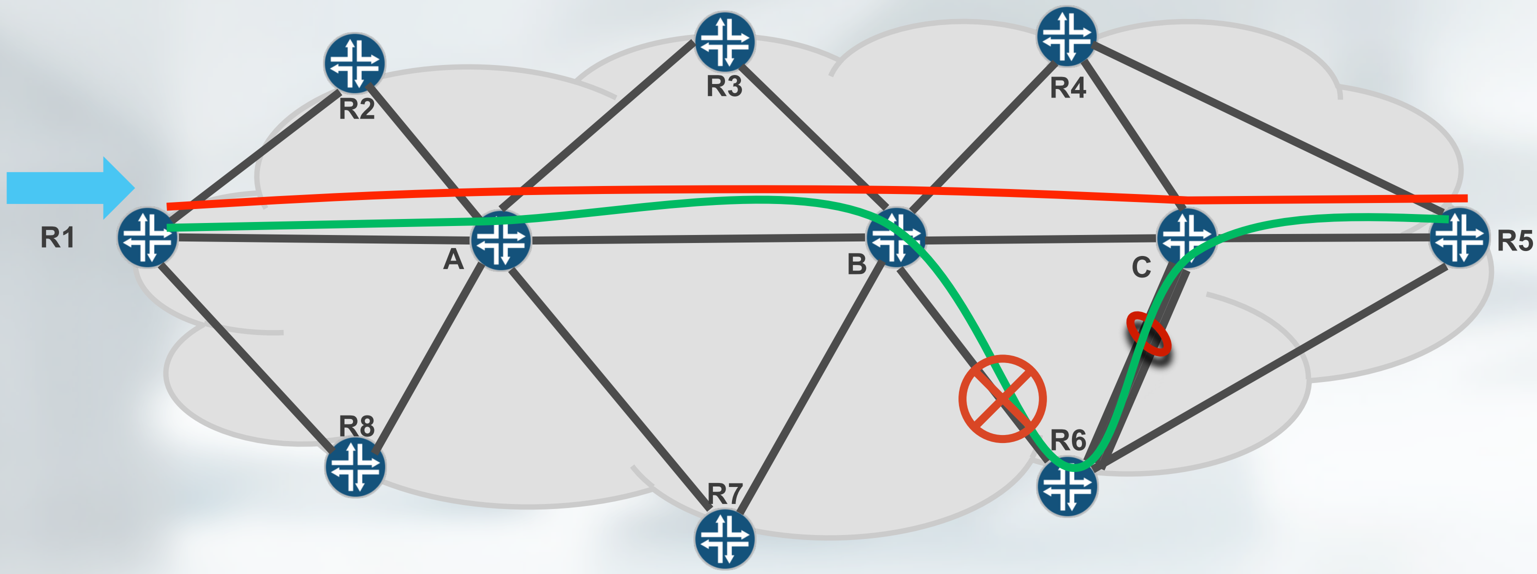
R6 would hash on the entropy label, when sending traffic over LAG between R6 & C

Data losses during LSP re-optimization

- How to prevent data losses on an LSP caused by a router advertising control-plane readiness before its data-plane is ready? Keys:
  - Ingress not start to use the re-optimized LSP path until the LSP's data-plane is healthy at every router on the path, or
  - Every transit router ensures that it not signal control plane readiness until its data plane is ready

- LSP Self-ping:
  - Ingress sends probe messages over the re-optimized LSP to ascertain readiness of data-plane of all routers on the LSP's path, before using it
    - IP data probes addressed to self
    - Data probes do not rely on egress control plane

Self-ping to check data-plane live-ness for an LSP



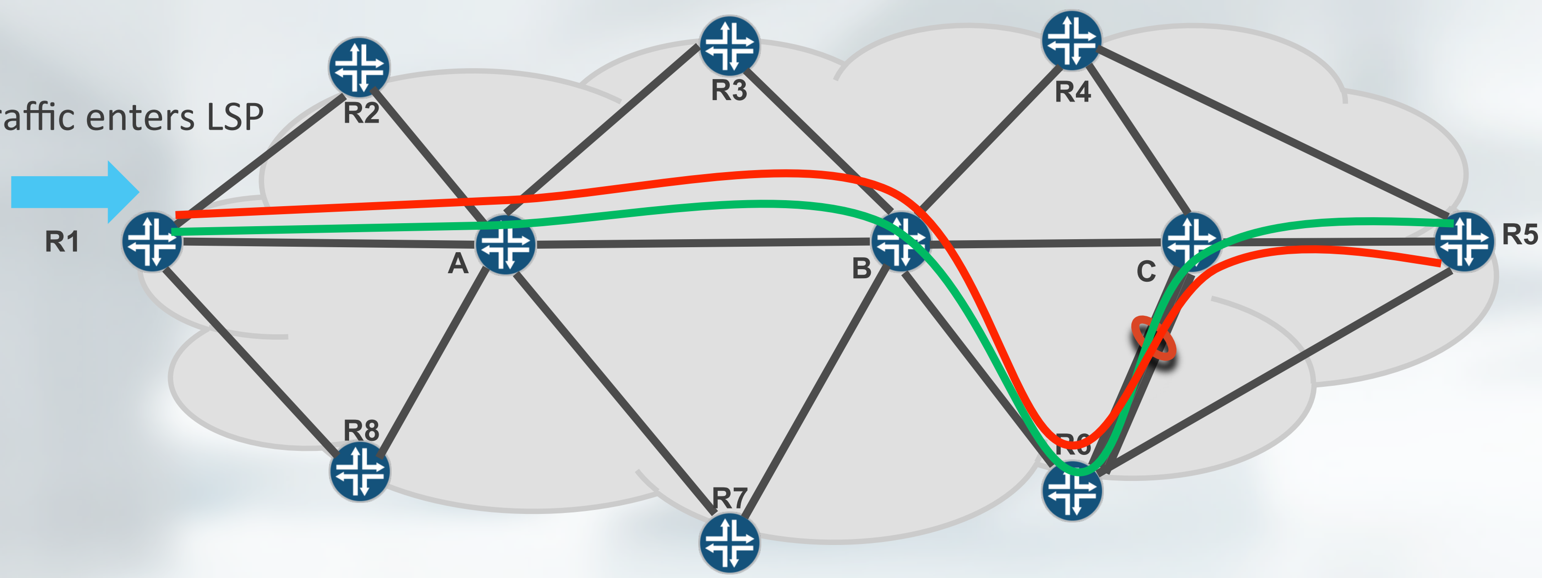
1. LSP's first instance: Green
2. B-R6 link fails (assume link-bypass LSP exists)
3. Global repair instance created: Red
4. R1 sends probes over Red instances
5. When probe received back at R1, Red instance goes live

Reducing network churn

- How to prevent avoidable re-signaling of LSPs? Keys:
  - **Reusing label** during make-before-break (when new path same as old path):
    - Starting at the egress/penultimate-hop router, for the re-optimized LSP
    - At transit router when every router south of this has reused a label on the re-optimized LSP
  - Judiciousness in determining
    - “need”: whether to re-signal an LSP:
    - “timing”: when to recompute path of LSP: smart CSPF delay
  - Partitioning importance of LSPs based on prioritization of not just the LSPs but also their bandwidth requirements

Label reuse during MBB

Traffic enters LSP



1. First LSP instance: Green
2. B-R6 link fails.
3. Global repair instance Red uses same path
4. Reusing same label (as in green instance): no need to reprogram FIBs
5. Churn avoided

Summary

- Solutions (conceptual/already-implemented) are available for the listed operational issues
- It is possible, at scale, to deploy RSVP-TE LSPs that:
  - Are easy to configure
  - Easy to monitor
  - Resize themselves
  - More effectively utilize the network
  - Load-balance traffic effectively

References

- Self-ping: http://tools.ietf.org/html/draft-bonica-mpls-self-ping-06
- RSVP-multipath/TE++: https://tools.ietf.org/html/draft-kompella-mpls-rsvp-ecmp-06
- Entropy label: https://tools.ietf.org/html/rfc6790
- Setup retry: https://tools.ietf.org/html/draft-ravisingh-teas-rsvp-setup-retry-00
- RSVP-TE scaling best current practices: https://tools.ietf.org/html/draft-beeram-mpls-rsvp-te-scaling-01

Thanks

V2.8