Segment Routing
the stuff marketing doesn’t talk about

steve ulrich (sulrich@juniper.net)

objective / disclaimer

objective:

- let's start having operations-oriented discussions around segment routing

disclaimer:

- this is a discussion of *some* of the details that don't come up when people are waxing poetic about segment routing
- nothing discussed here is intractable - it's just work
- as an industry we are still working through many of these issues
  - it's going to take time
  - there will be bruises (and probably some scarring)
- this discussion assumes the desire to do something optimal with traffic
  - if you're simply replacing LDP, most of this doesn't apply to you

agenda

- segment routing in a flash
- obvious things
  - label management (space and stacks)
  - RSVP-TE and SR coexistence / migration
- less obvious things
  - controller care and feeding
  - SRTE protocols
  - traffic protection
- summary

segment types and label spaces

BASIC SEGMENT TYPES

- Adjacency-SID (single router hop)
  - represents an IGP adjacency
  - node-local significance
- Prefix-SID (one or more hops)
  - represents IGP least cost path to a prefix
  - Node-SIDs are a special form of Prefix-SIDs bound to loopback
  - domain-wide significance

ADVANCED SEGMENT TYPES

- Anycast-SID (one or more hops)
  - represents IGP least cost path to a non-uniquely announced prefix
- Binding-SID
  - represents a tunnel

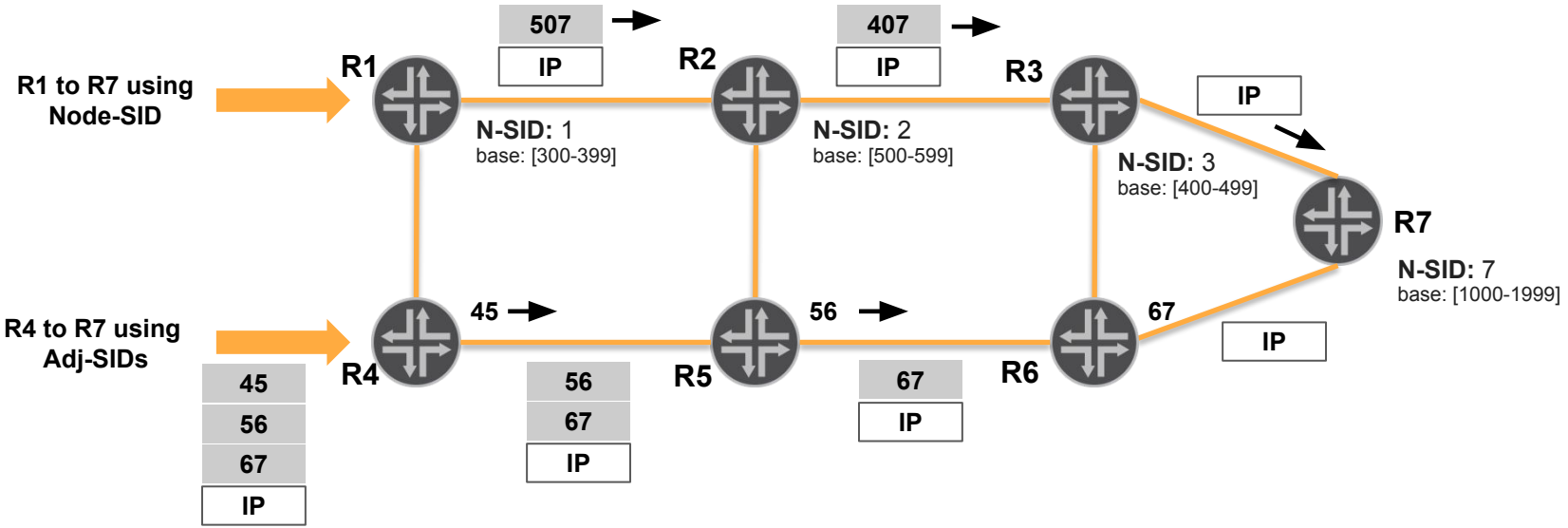
SEGMENT ID (SID) SPACE

- SIDs are not labels
  - but - SIDs are encoded (carried) in labels
- domain-wide SIDs coordinated via IGP
- domain-wide SIDs are allocated in a manner much like RFC1918 addresses
  - each node reserves a block of labels. this label block is the Segment Routing Global Block (SRGB).
  - global label = SRGB base value + index

basic SR forwarding examples

Prefix/Node-SID forwarding (using SRGB)

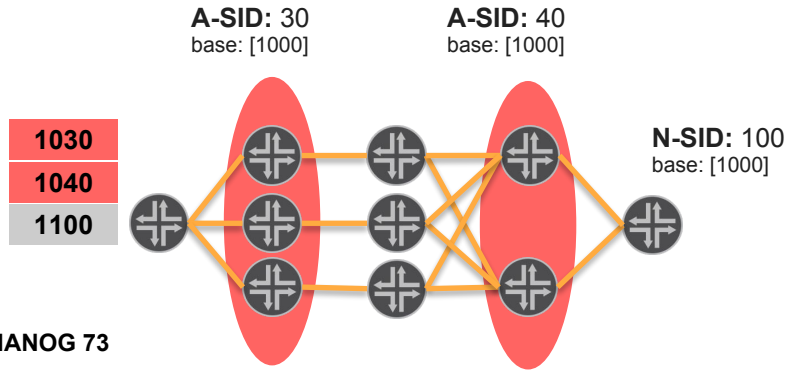
- R1 shortest path to R7 is via R2.
- R2 expects a label value equal to {R2 label-base + index of destination}
  R1 => R2 label = 507 {500 + 7}



Anycast-SIDs / Binding-SIDs

Anycast-SIDs

- have domain-wide significance
- define a set of nodes via a non-uniquely announced prefix
- forwarding choice is made via IGP SPF
- can use ECMP for forwarding
- add redundancy, enable load balancing
- commonly represent a set of geographically close nodes (e.g.: metro)

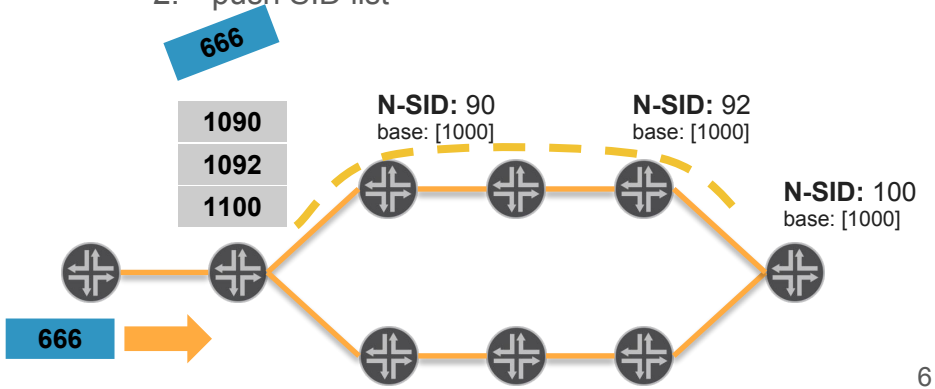


Binding-SIDs

- have node-local significance
- are bound to other SR paths
- enable an SR path to include another SR path by reference
- are useful for scaling the SID stack at ingress

Binding-SID forwarding operation:

1. pop Binding-SID label
2. push SID list



obvious stuff

label space management - global labels

global label space - Prefix-SIDs, Node-SIDs, Anycast-SIDs

- operation of Prefix-SIDs is reasonably well established across implementations
- Anycast-SID operation may have SRGB-specific considerations
  - it is recommended that nodes announcing an Anycast-SID have an identical SRGB, drafts are reasonably explicit on this point
  - further, labels after Anycast-SID must be resolvable by downstream nodes
- Anycast-SID has had interesting interop considerations
  - behavior across major vendors has largely been clarified
  - however, there is still opportunity for misconfiguration and blackholing
    - e.g.: discontinuities in the resolution or announcement of Anycast-SIDs

label space management - local labels

local label space – Adjacency-SIDs, OAM labels, service-specific labels

there may be implementation subtleties in the operation and allocation of local label space

e.g.: some implementations have the concept of static or local service labels, the migration to SR may require managing through the allocation of these service-specific labels in your environment.

label stack size

segment routing provides for very granular traffic control, where the controller does explicit path specification with a combination of global and/or interface specific labels on the head of the packet.

sounds great - carries additional considerations

hardware encapsulation capabilities - some hardware is severely constrained as to the number of labels that can be imposed in a single pass

- includes some popular chipsets
- if you control one end of the connection you may be able to offload some label imposition processing to your host stack
- if you're a transit/network provider pay careful attention to the ingress (edge) hardware capabilities
- if you need very specific traffic engineering capabilities (read: link-specific placement) this is a notable consideration

label stack size

tl;dr - make sure you understand your hardware capabilities and traffic behaviors. deep label stacks have additional hardware considerations, beyond encapsulation.

- transit node/link implications
  - will all transit nodes support / forward deep label stacks?
  - on all line cards in the system?
- load balancing considerations
  - for nodes that support forwarding deep label stacks what are the entropy sources available or activated?
  - does use of deep label stacks obscure L3/L4 entropy sources that you really need to achieve load balancing objectives on LAGs?

“no worries! i’m going to use Anycast-SIDs and Prefix-SIDs to define paths and i’ll have a small label stack.” -- we’ll come back to this.

RSVP/SR coexistence (and migration)

2 parts to this discussion

- objectives
- control-plane behaviors and operation

objectives

- dominant assumption is that **migration** from RSVP to SR is the objective.
- if there is a long-term need to run both RSVP and SR on the same infrastructure – it's likely preferable to put both domains under a common controller as soon as possible
    - particularly if P2MP-TE is in the mix

RSVP/SR coexistence

control-plane behaviors and operation

- placement of SR LSPs in the same domain as RSVP-TE LSPs runs the risk of introducing inaccuracies in the TED that is used by distributed or centralized RSVP path computation engines
- generic problem associated with management of dark bandwidth pools

draft-ietf-teas-sr-rsvp-coexistence-rec-xx - provides detailed discussion of the migration considerations (currently in WGLC)

RSVP/SR coexistence solution options (1)

static bandwidth partitioning

- reservable interface bandwidth is statically partitioned between SR and RSVP-TE
- each operates within respective bandwidth allocation

downside
potentially strands bandwidth; protocols cannot use bandwidth left unused by the other protocol

centralized capacity management

central controller performs path placement for both RSVP-TE and SR LSPs

downside
requires the introduction of a central controller managing the RSVP-TE LSPs as a prerequisite to the deployment of any SR LSPs

RSVP/SR coexistence solution options (2)

flooding SR Utilization in IGP

SR utilization information can be flooded in IGP-TE and the RSVP-TE path computation engine (CSPF) can be changed to consider this information

downside

- requires changes to the RSVP-TE path computation logic
- carries upgrade requirement in deployments where distributed path computation is done across the network

running SR over RSVP-TE

run SR over dedicated RSVP-TE LSPs that carry only SR traffic.

downside

requires SR to rely on RSVP-TE for deployment

RSVP/SR coexistence solution options (3)

reflect SR traffic utilization by adjusting Max-Reservable-BW

- dynamically measure SR traffic utilization on each TE interface and reduce the bandwidth allowed for use by RSVP-TE
- incurs no change to existing RSVP path calculation procedure
- assumes the use of auto-bw w/i RSVP domain
- controller may operate entirely within the context of the SR traffic domain

implementations are shipping

reflection procedure on each TE node as follows:

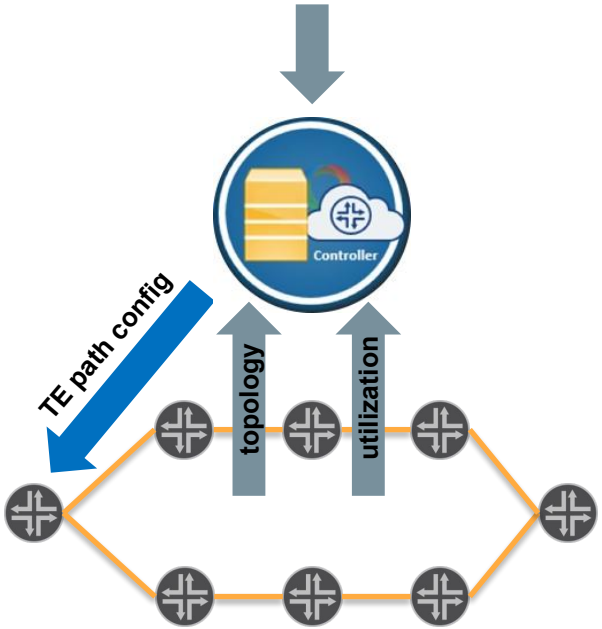
- periodically retrieve SR traffic statistics for each TE interface
- periodically calculate SR traffic average over a set of collected traffic samples
- if the change in SR traffic average is greater than or equal to SR traffic threshold percentage (configured), adjust Max-Reservable-BW
  - results in the RSVP-Unreserved-BW-At-Priority-X being adjusted
- RSVP-TE nodes can re-optimize LSPs accordingly

less obvious stuff

controller (+ collector)

- controller acquires LSDB
  - passive IGP / BGP-LS / telemetry
- controller understands current network state and utilization via collector
- calculates traffic demands vs. capacity and availability requirement
  - understands h/w capabilities
  - aware of current and projected loads
- controller sends segment list (path) to ingress router to place traffic
  - configuration / BGP SRTE / PCEP
  - other RIB programming mechanisms

- business logic
- workload demands
- availability requirements
- network capabilities



centralized path computation

benefits

- centralized control has global view of reserved/available bandwidth
  - not available at any other point in the network
- facilitates analytics driven policy
  - controller receives telemetry
  - based on telemetry, controller configures / alters policy

additional considerations

- requires developing a controller or purchasing a controller
  - staffing and ongoing maintenance of controller development
  - new deployment and/or vendor dependencies
- concentrated point of failure / congestion
  - risks mitigated by redundant controllers

SR traffic engineering

a brief aside

with segment routing traffic engineering is now primarily controller driven

- if there are hardware constraints (on imposition or transit) the controller must calculate longest best paths taking into consideration Anycast/Prefix-SIDs
- algorithms to compress the label stack are a hot area of optimization
- some implementations are being extended to support dynamic, distributed computation with SR ingress nodes providing RSVP-like path calculation taking into consideration path constraints (affinity, SRLGs, etc.)

controller care and feeding

- to effectively place workloads on the network the controller must have visibility into current network utilization and loading
- a controller must respond to fluctuations in traffic quickly to prevent overloading hot links and gracefully migrate traffic loads
- implies significantly more aggressive instrumentation cycles than is commonly seen in today's networks with a complementary feedback loop to move workloads onto less-utilized paths / rebalance traffic
- reworking instrumentation to utilize streaming telemetry is a practical day-0 requirement
- per-label traffic statistics - something we're now talking about

we need to talk about stats

given the controller's need for stats, what does the hardware do?

- **it depends:** the ideal is per-interface, per-direction, per-label, per-class statistics, ditto for policy stats (draft-ali-spring-sr-traffic-accounting-xx)
- reality is far uglier
  - outside of FIB and ACL space, counters are the most precious resource on modern ASICs
  - you're more likely to get a subset of the above (wish)list
- getting stats off of network elements is another consideration
  - per-interface, per-label statistics requires significant and often new collection infrastructure
- if you get some useful subset of stats info, what does a label counter get you?

what's in a counter?

Anycast/Prefix-SIDs

- present as a single counter for lots of traffic underneath
- what are the sources for all that traffic?
  - what's been merged underneath these labels?
  - multiple ingress points in the network?
  - how do you find the right traffic to re-optimize?

SRTE policy counters

- how many policies may resolve to a common segment list?
- how many segment lists collapse to a common set of AnyCast/Prefix-SID destinations at midpoints?
- will require planning on how to manage and instrument sources and sinks within the network

punchline: double down on your investment in IPFIX / sFlow collection infra

SRTE protocols: BGP SRTE

BGP SRTE

- the current draft remains an active area of development
- provides useful capabilities in ECMP-dense environments
- no tunnel/virtual interface configuration, forwarding is instead tied to policy
  - think “rules for steering” - not, explicit-path placement

- new considerations re: data-plane programming and validation
  - Q: how do you know the node accepted the list of segment lists you sent it?
  - Q: how do you know what might have been tangled up in policy logic?
  - A: you don’t. you’ll have to ask the node afterwards. you’ll want telemetry for that.
- Q: do you need to specify a protection / bypass path?
  - this might not be the tool you’re looking for

SRTE protocols: PCEP

PCEP (stateful)

- provides single protocol for the management of RSVP and SR paths
- flexible management and delegation models
- requires additional mechanisms for prefix binding and flow specification
- has an RSVP-ish operational view
  - capable of signaling SR paths; traffic / flow-mapping is work-in-progress
  - protection path placement pending … (resurrect the local protection-draft)
- provides options for some form of contract with the ingress nodes
  - can the hardware do what you asked of it?
  - with PCEP the controller can understand node capabilities and act accordingly

SRTE protocols: RPC-based path placement

emergent RPC-based mechanisms for path placement

some operators are looking to leverage RIB APIs available from vendors and modeling consortia

- pRPD from juniper (https://juni.pr/2rtY2fV)
- gRIBI from OpenConfig (https://bit.ly/2HZwN7i)
- EOS APIs from arista (https://bit.ly/2xuHNVp)
- service layer APIs from cisco (https://bit.ly/2fRvzhz)

RIB APIs

- commonly provide mechanisms to define label stacks / paths
- provide mechanisms to associate RIB entries with these paths
- enable new controller selection models
- use modern software development tools
  - leverage widely available tools & protocols
  - make your developers happy(-ish)
  - enables more sophisticated error-handling

additional considerations:

- requires internal development expertise
- commonly leveraging a vendor-specific interfaces
  - associated API management policies
  - new test, cert and deployment packaging considerations

SRTE traffic protection

- it's 1AM, do you know what your protect path is?
- did you get to specify it? probably not.
- how much traffic is going to go over that path? are you sure?

TI-LFA is commonly the reflexive response for SR traffic protection

lots to like

- no midpoint state
- true post-convergence path provides optimality - no u-loops!
- cool sounding acronym

practical reality

- computationally intensive
  - particularly if SRLGs, etc. in the mix
- may not be deterministic
  - particularly across vendors
- may require label stack compression to stay within protection encap capabilities
- ref. prior conversation about counters and load placement (or finding big flows)

deployment considerations

- protect path placement remains an active area of development
- operators requiring explicit protection placement and an understanding of protect path capacity will want to understand available TI-LFA behaviors deeply or explore other options

summary

- TE didn’t really get easier - it just got different
- lots of work remains to operationalize segment routing for traffic engineering
- data plane simplification and elimination of control plane state network means building new infrastructure to account for lost or shifted functionality
- vendors are actively developing the tooling to make deployments happen
- in the meantime
    - expect considerable variability in implementation capabilities and installed footprint
    - be prepared to roll your own solutions to some of these problems

look forward to more NANOG discussion around these topics as we, as an industry, gain operational experience

thank you.