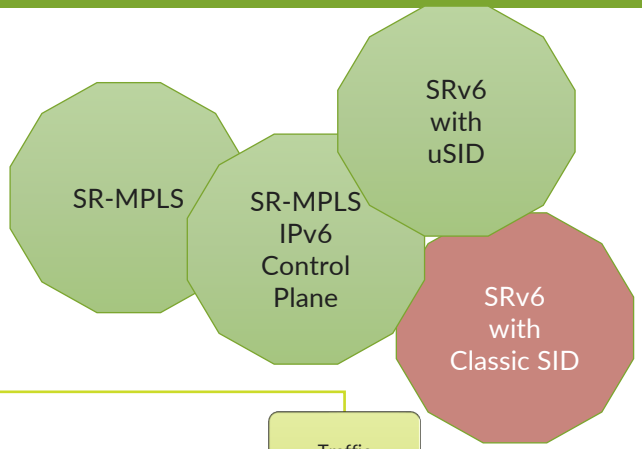
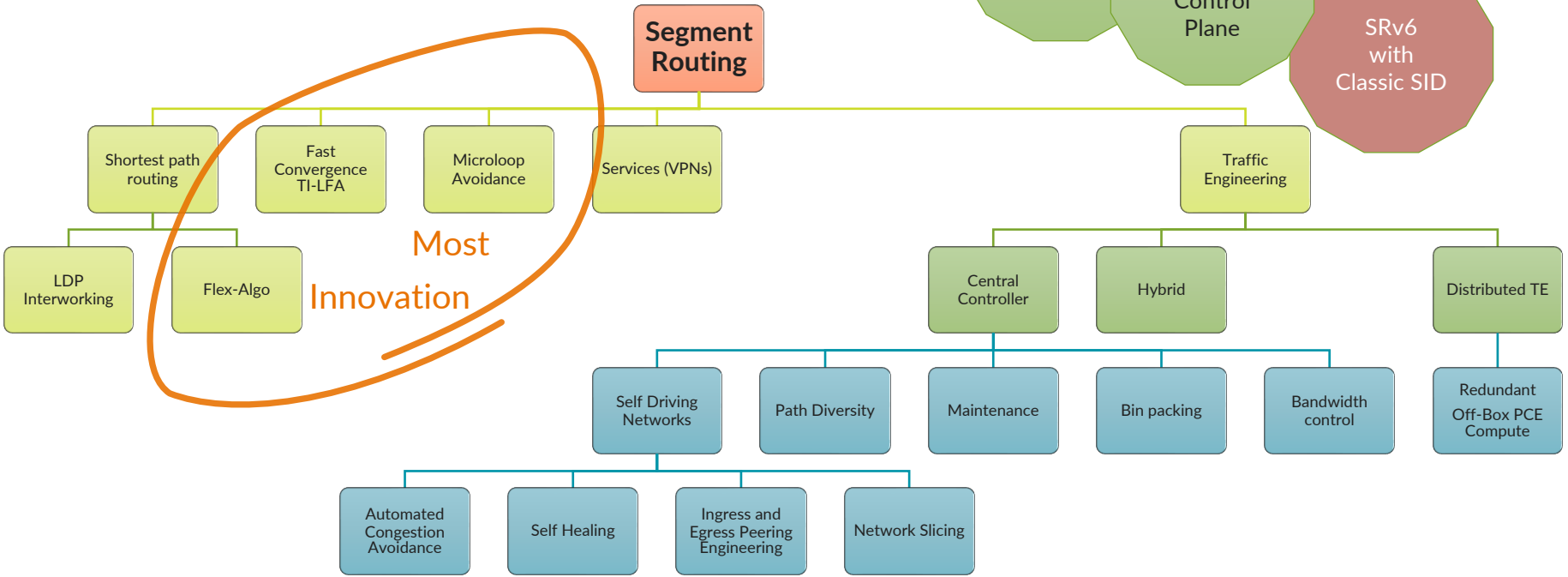
Segment Routing: Applications, Innovation?

Anton Elita

March 2024

Segment Routing Applications





Topology-Independent Loop Free Alternate (TI-LFA)

Best backup coverage – even there, where LFA or remote LFA fail

Compressed and efficient label stack – mix of Node SIDs and Adj SID

Protects against link and/or node failure, considers SRLG and fate-sharing groups

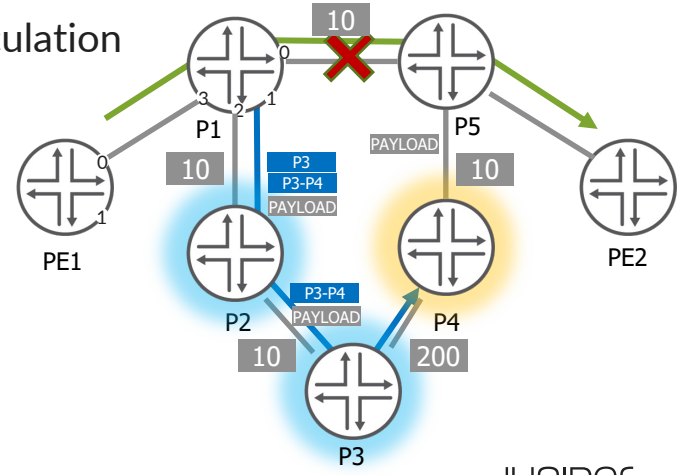
Backup path is actually the post-convergence path, no need to bounce traffic again

- SPF just prunes the protected element during pre-calculation

Limiting factors for TI-LFA:

- area/level boundary

- non-SR hops



TI-LFA *Node* Protection: SR *vs* SR-TE [stack]

In this topology, PE1 has two MPLS paths towards PE2: blue SR-TE with label stack [1006 1002], and green SR path with hop-by-hop swap of [1006]. P1 and P4 are configured for TI-LFA node protection.

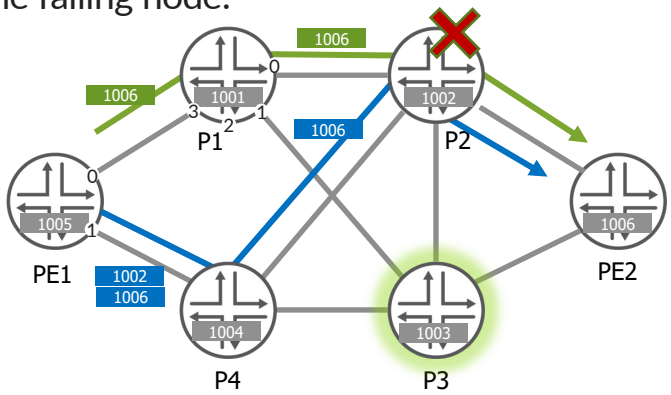
If P2 fails, P1 will be able to fast-reroute over P3, but P4 will not. Why is that so?

A protecting node like P1 or P4 install pre-computed backup paths for the protected destinations.

For P1, the backup interface is towards P3, and label operation is SWAP (1006 <-> 1006).

For P4, the incoming label is 1002 (operation POP) – the Node SID of the failing node!

P4 cannot do "node protection" for the label 1002.
Label 1006 comes in as a second label in the stack.



SR-MPLS for Inter-AS with [Anycast] Prefix-SID

```
* 11.1.0.1/32 (1 entry, 1 announced)
     Accepted
     Route Label: 10002
     Nexthop: 11.40.0.2
     AS path: 64501 I
     Entropy label capable, next hop field matches route next hop
               Prefix SID 2, (ref cnt 1)
                SRGB Start Index: 10000 Size: 36000
```

Traditional BGP-LU advertises dynamic labels (e.g. for PE2)

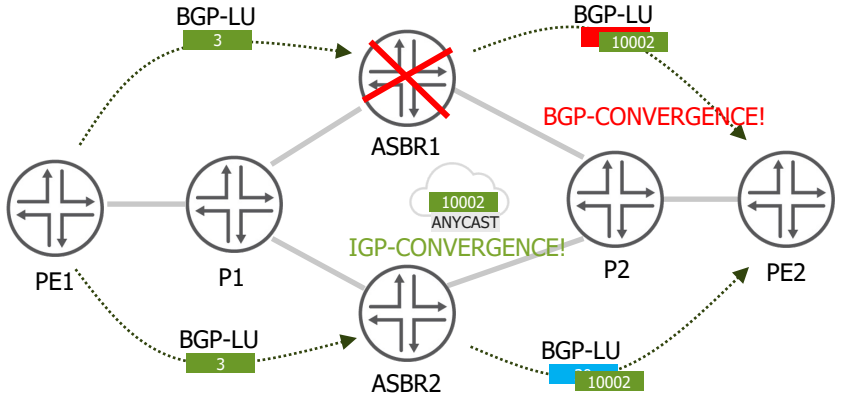
When ASBR1 is unreachable, need to wait for BGP to converge

- ASBR1 and ASBR2 allocated different labels for PE2

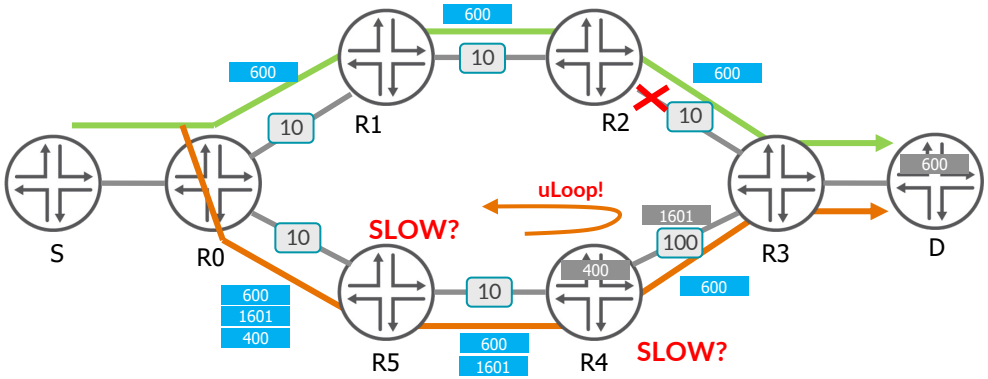
RFC8669 defines Prefix-SID advertisement via BGP-LU (optionally, SRGB)

ASBR1 and ASBR2 use the same Anycast SID as "next-hop-self" towards PE2

- when ASBR1 fails, IGP redirects traffic towards ASBR2
- ASBR2 already has label stitching programmed
- SR node labels are now "global" significance



Microloop Avoidance: Local, Remote



Problem:

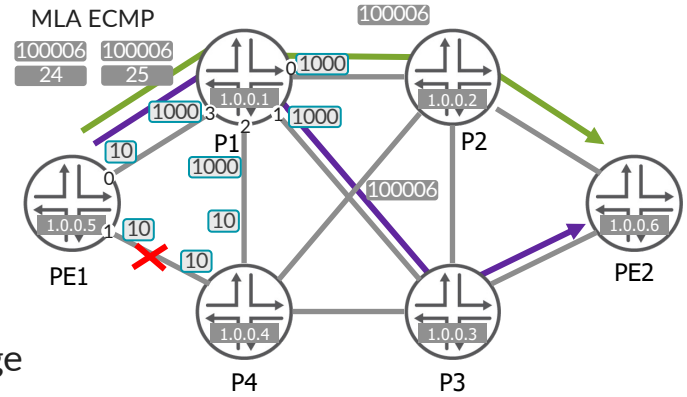
- Before the failure of link R2->R3:
  Shortest path from Source(S) to Destination(D) is S→R0→R1→R2→R3→D.
- After the failure of link R2->R3, micro-loops may occur if:
  - If R0 updates its forwarding state before R5, packets will loop between R0 and R5.
  - If both R0 and R5 have updated their forwarding states and R4 has not, packets will loop between R4→R5

Solution:

In the event of failure of R2-R3 link, R0 programs the microloop avoidance path towards R3 using Node SID of R4 and Adj-SID of R4-R3, for a configurable amount of time.

Microloop Avoidance

- Microloop avoidance is offered for local or remote failures
- Creates multiple ECMP protection paths
  - if maximum-labels/sids not exceeded
- uLoop avoidance path is created after IGP has signaled a change
  - link down
  - link up
  - metric change



Example of local uLoop avoidance

ECMP for protection paths

- it's not a replacement for fast-reroute mechanisms like TI-LFA, as it's not pre-programmed in hardware
  - TI-LFA and microloop avoidance can co-exist
- supports Flex-Algo and multi-instance topologies
  - MLA paths would be established within respective Flex-Algo or instance

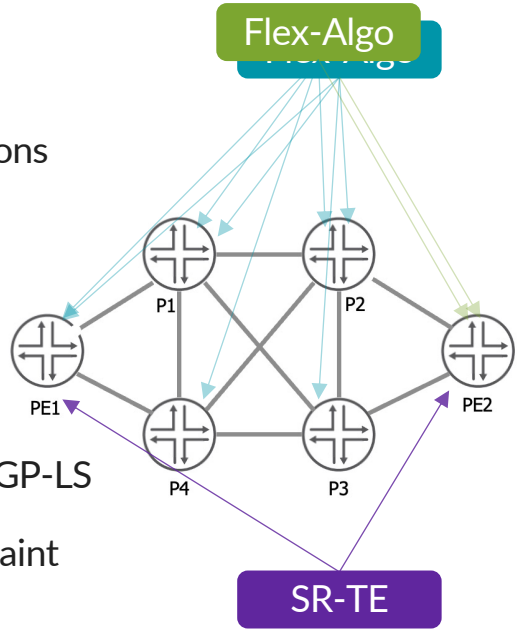
SR-TE or Flex-Algo?

Flex-Algo

- One additional SPF topology per Algo
- Winner's Flex-Algo Definition broadcasted
  - even P devices need it
- Additional Node SID per node per Algo
  - even P devices need it (e.g. for TI-LFA)
- Still SPF-based, i.e. "lightweight" TE
- TI-LFA within Flex-Algo 👍
- A topology per Algo - scaling? 👎
- ECMP

SR-TE

- Local (or PCE) based routing decisions
- Only ingress PE need policies
- Full-fledged traffic engineering
  - many more constraints
  - combination of constraints per TE
- TE database can be extended by BGP-LS
- Can even use Flex-Algo as a constraint
- ECMP with node SIDs
  - or multiple segment-lists with Adj-SIDs
- On-Demand Next-Hop as an alternative to Full-Mesh

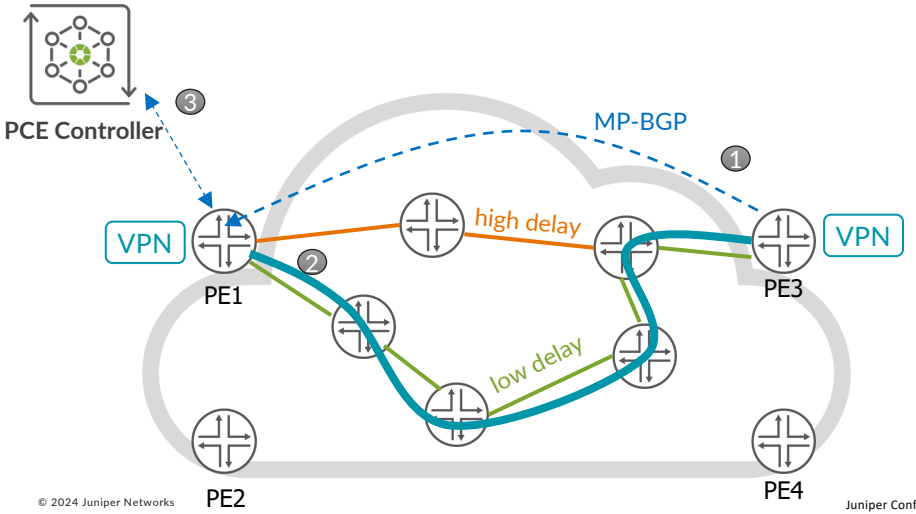


TE Full-Mesh? On-demand TE!

Creating a pre-defined “mesh” of tunnels can be complex, cumbersome and/or generally undesirable.

Dynamic/on-demand tunnels automates tunnel creation

1. Triggered by the arrival of a “service route(s)”
2. Creates only the required tunnels, auto-destroys unused after a time-out
3. On-box or off-box computation



Define dynamic-tunnel template

```
dynamic-tunnels
    sr-dyn-tun {
        spring-te {
            source-routing-path-template {
                odn-igp color 999;
                odn-te color 666;
                odn-delay color 333;
            }
            destination-networks {
                1.1.1.0/24;
```

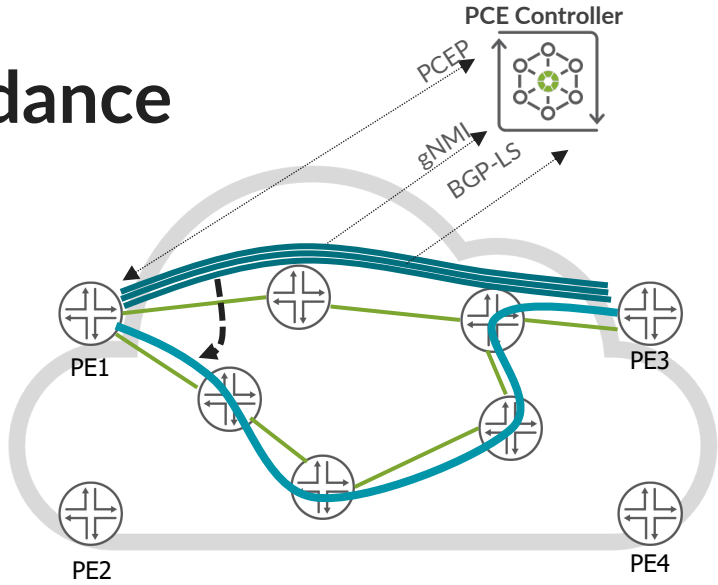
Define optimization objective

```
source-routing-protocols {
    compute-profile delay {
        metric-type {
            delay;
            variation-threshold 1234;
```

Define define SR-policy template

```
source-routing-path-template odn-delay {
        primary {
            sl1 {
                compute delay;
```

Automated Congestion Avoidance



Basic idea:

Moving TE LSPs is more precise and less impacting, compared to interface metric changes.

Conditions:

- enough entropy, enough traffic over multiple TE LSPs.
- a controller with a global view, measuring per-LSP and per-Interface traffic.
- no new congestion to be created, LSP constraints to be respected: latency, number of hops, diversity, priority etc.

Large Networks (Thousands of PE devices)

- separate IGP in every network part
- BGP-LU for end-to-end services, best-effort routing
- great scale
- service over best-effort paths

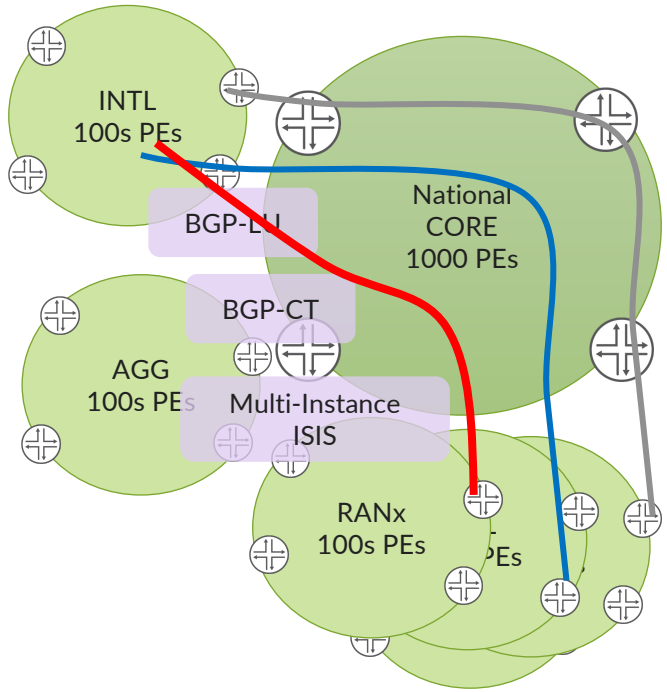
MPLS

- separate IGP in every network
- BGP-CT, differentiated services (delay, geo policy ...)
- great scale
- any underlay (RSVP, SR-TE, Flex-Algo)

MPLS or SRv6

- separate ISIS instance in every network
- Flex-Algo + Prefix-Metric for end-to-end colored service
- summarization at the ABR, great scale

SRv6



ÖL? BIER?

Multicast in the Core

Is Multicast Still Needed?

Yes it is, for many service providers, broadcaster companies and so on.

Legacy multicast implementations require state in the core: PIM, PIMv6, mLDP, RSVP-TE P2MP or SR-P2MP.

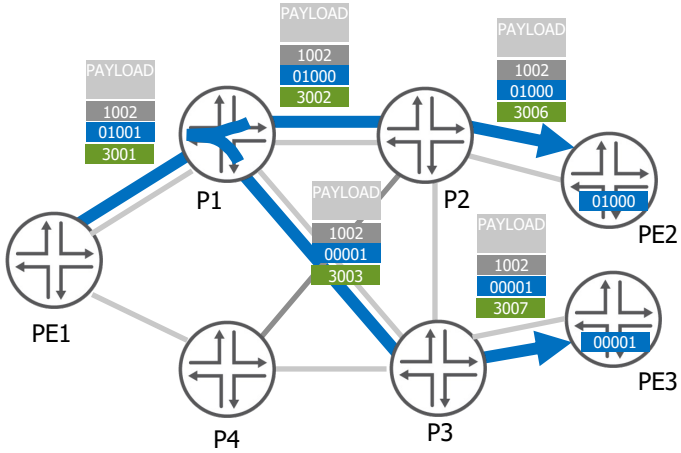
Ingress replication does not require per-tree state in the core, but is an inefficient type of replication.

Bit **I**ndex **E**xplicit **R**eplication: efficient, no per-tree state (signaled by IGP), any multicast scenario.

BIER at IETF was defined five years ago... Why did we wait so long?

- Vendor support for pipeline architectures – new ASICs
- Segment Routing adoption (unicast first)

Interoperability tests performed in 2023, more to come in 2024



1002 VPN Label
01001 BIER header with bitstring
3001 MPLS label

Example forwarding for BIER-MPLS

Multicast Options in SR Networks

BIER

- If you care about efficient replication without per-tree state inside the network, and,
- If most routers support BIER

Traditional Multicast (PIM/P2MP/IR)

- If it works well for you
  - You don’t need controller, and,
  - You don’t mind running PIM/mLDP/RSVP in your SR network for multicast
    - Perfectly ok to run PIM/mLDP/RSVP for multicast while running SR unicast

Controller Signaled Multicast

- If you need controller-calculated trees, and/or,
- You want to remove PIM/mLDP/RSVP
- Note that you will still have per-tree/tunnel state inside the network

Questions?

Thank you

JUNIPER NETWORKS | Driven by Experience™

NET NOD