You make **possible**

Traffic Engineering with Segment Routing

Leonir Hoxha – Senior Consulting Engineer

BRKSPG-2021

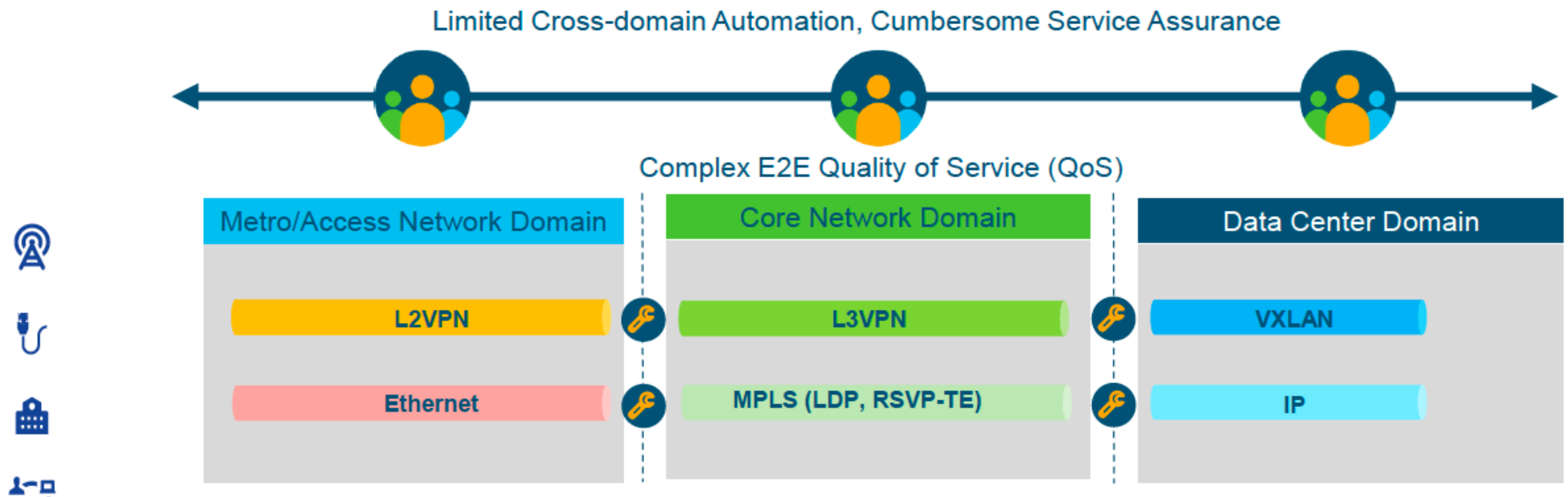
@ccie49534

Agenda

- Segment Routing Overview
- Control & Data Plane
- Segment Routing Traffic Engineering
- Automated Steering
- On-Demand Next-Hop
- SR-PCE
- Flexible Algorithm (Flex-Algo)

Segment… what?

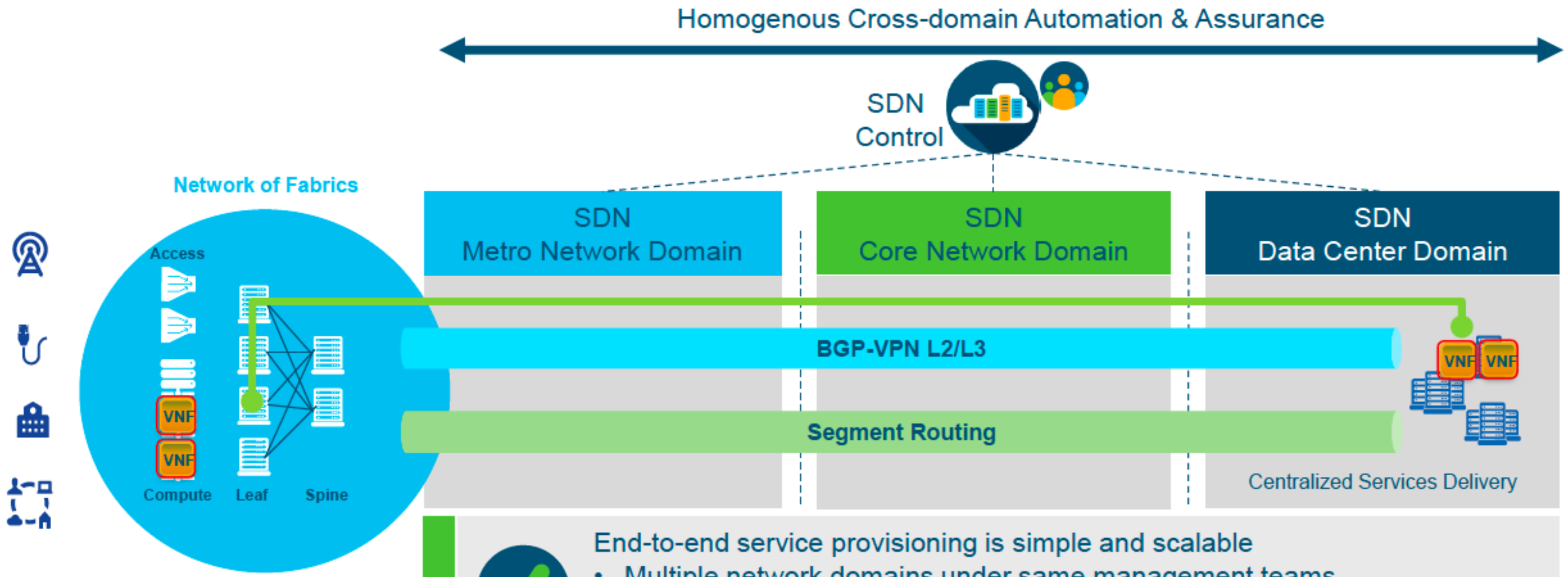
Problem Statement: Today's Service Creation



End-to-end service provisioning is lengthy and complex

- Routing protocols are blind to applications (Shortest-path for any traffic)
- Multiple network domains under different management teams
- Manual operations
- Heterogeneous underlay and overlay networks

Segment Routing Unified Fabric Vision



End-to-end service provisioning is simple and scalable

- Multiple network domains under same management teams
- Automated operations
- Homogenous underlay and overlay networks

Segment Routing: Value Proposition

Create New Revenue Streams
- Differentiate Services with SR Policies
- Statelessly Program Value-Add Services (no added protocols)

Deploy with Ease
- Seamless Brownfield Integration
- Single Control for Inter Domain Implementations

Monitor Health
- Data Path Validation Including ECMP
- Real Time Per-Link Performance Monitoring with Telemetry

Increase Availability
- Automated 50ms Protection
- Assured Loopfree Convergence upon Recovery

Multi-vendor consensus - Designed and built with network operators

Segment Routing Overview

You make networking **possible**

Segment Routing

- **Source Routing**
  - the source chooses a path and encodes it in the packet header as an ordered list of segments
  - the rest of the network executes the encoded instructions
- **Segment**: an identifier for any type of instruction
  - forwarding or service

Segment Routing – Forwarding Plane

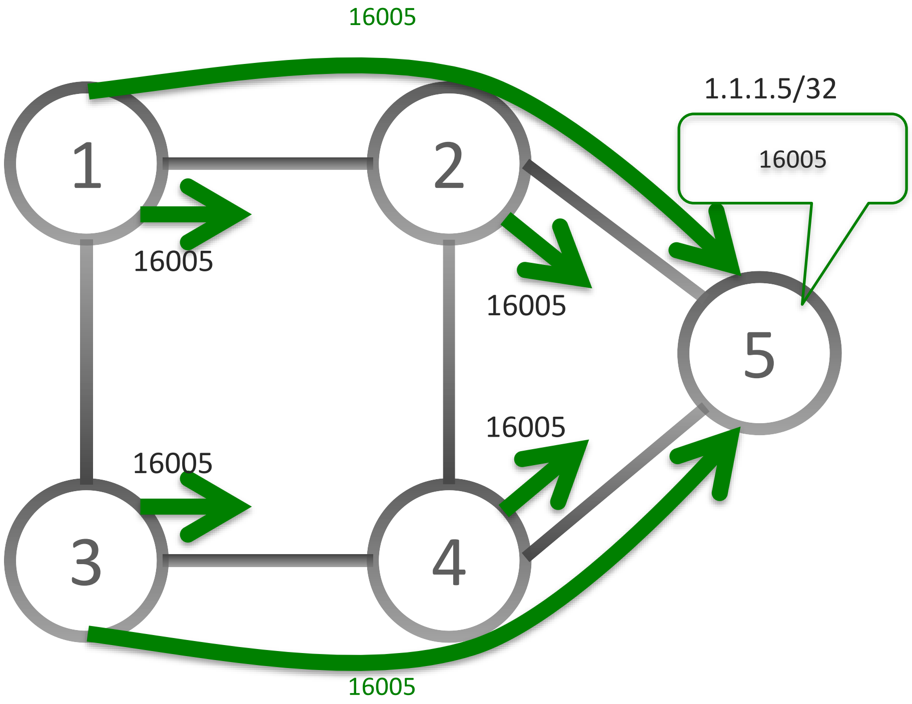
- **MPLS**: an ordered list of segments is represented as a stack of labels
- **IPv6**: an ordered list of segments is encoded in a routing extension header
- This session: **MPLS data plane**

Segment Routing – IGP Segments

- Two basic building blocks distributed by IGP
  - Prefix Segments
  - Adjacency Segments

IGP Prefix Segment

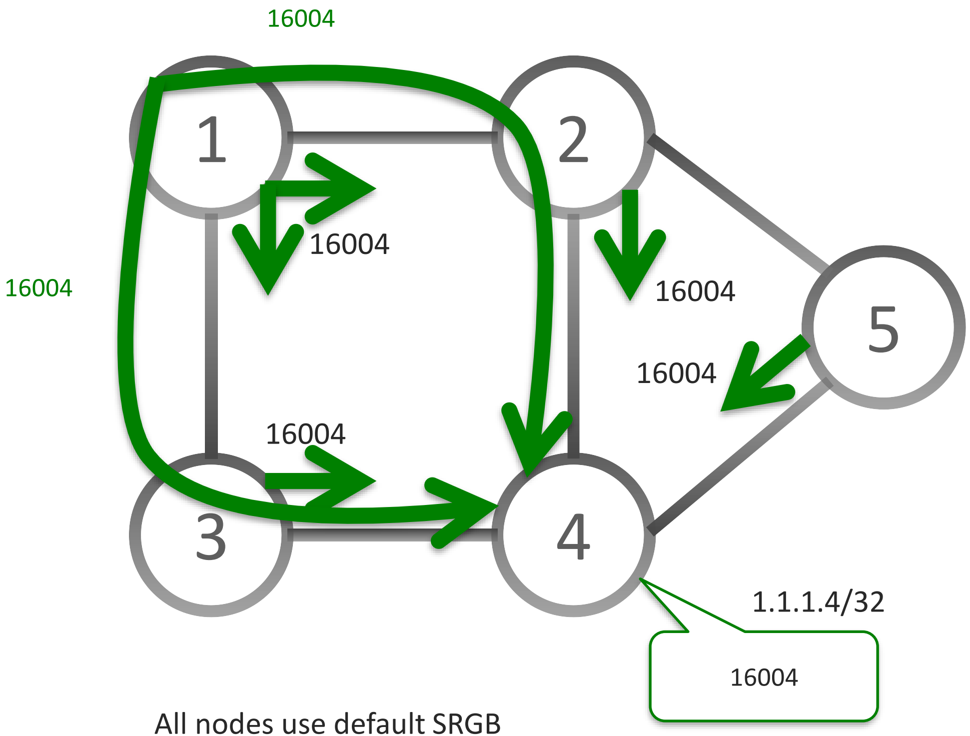
- Shortest-path to the IGP prefix
  - Equal Cost Multipath (ECMP) Aware
- Global Segment
- Label = 16000 + Index
  - Advertised as index
- Distributed by ISIS/OSPF
- Manually assigned



All nodes use default SRGB
16,000 – 23,999

IGP Prefix Segment

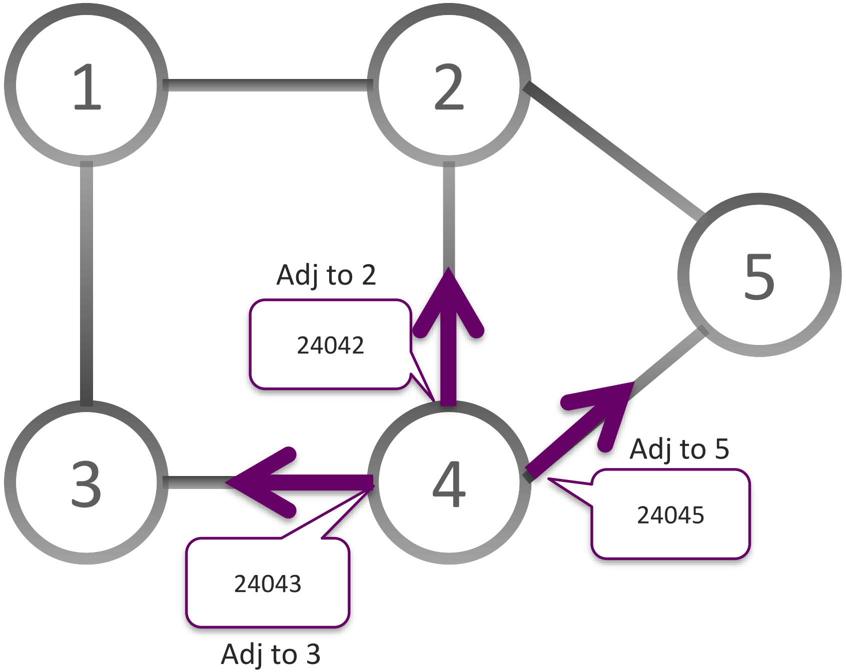
- Shortest-path to the IGP prefix
  - Equal Cost Multipath (ECMP) Aware
- Global Segment
- Label = 16000 + Index
  - Advertised as index
- Distributed by ISIS/OSPF
- Manually assigned



All nodes use default SRGB
16,000 – 23,999

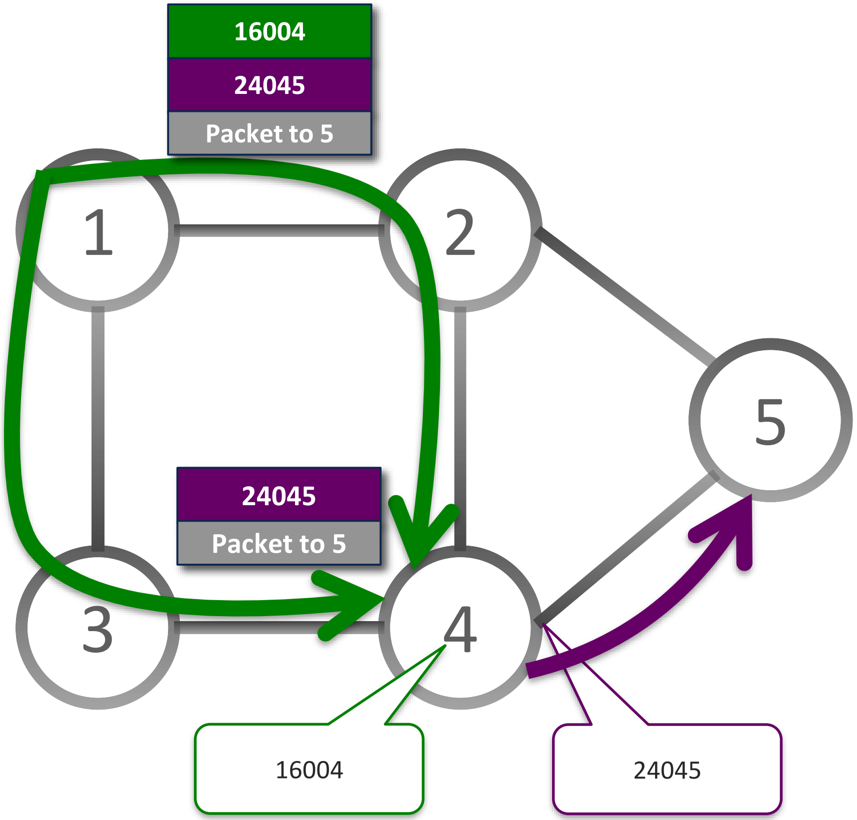
IGP Adjacency Segment

- Forward on the IGP adjacency
- Local Segment
  - Dynamically Allocated
- Distributed by ISIS/OSPF
- Advertised as a label value



Combining IGP Segments

- Steer traffic on any path through the network
- Path is specified by a stack of labels
- No path is signaled
- No per-flow state is created
- Single protocol: IS-IS or OSPF

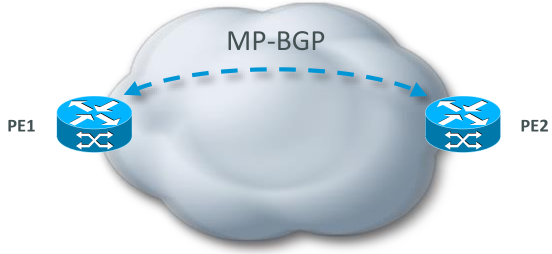


Control & Data Plane

You make security **possible**

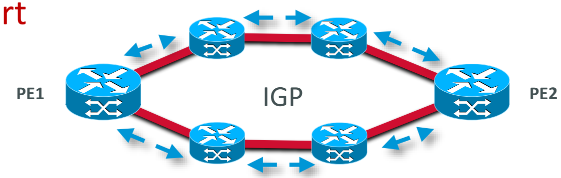
MPLS Control and Forwarding Operation with Segment Routing

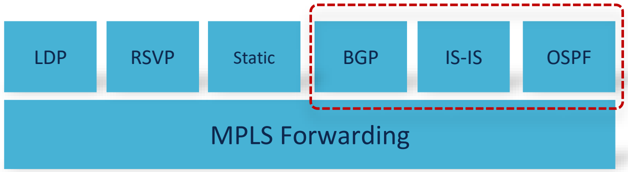
Services



No changes to control or forwarding plane

Packet Transport





IGP or BGP label distribution for IPv4 and IPv6. Forwarding plane remains the same (MPLS)

SID Encoding

SR enabled node

SRGB = [16,000 – 23,999] – Advertised as base = 16,000, range = 8,000
Prefix SID = 16,001 – Advertised as Prefix SID Index = 1
Adjacency SID = 24000 – Advertised as Adjacency SID = 24000

- Prefix SID
  - Label form SR Global Block (SRGB)
  - SRGB advertised within IGP via TLV
  - In the configuration, Prefix-SID can be configured as an absolute value or an index
  - In the protocol advertisement, Prefix-SID is always encoded as a globally unique index

    Index represents an offset from SRGB base, zero-based numbering, i.e. 0 is 1st index

    E.g. index **1** → SID is 16,000 + **1** = 16,001
- Adjacency SID
  - Locally significant
  - Automatically allocated by the IGP for each adjacency
  - Always encoded as an absolute (i.e. not indexed) value

SR IS-IS Control Plane Summary

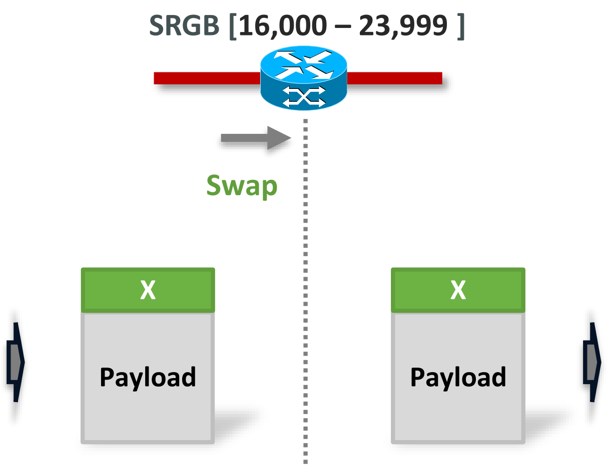
- IPv4 and IPv6 control plane
- Level 1, level 2 and multi-level routing
- Prefix Segment ID (Prefix-SID) for host prefixes on loopback interfaces
- Adjacency Segment IDs (Adj-SIDs) for adjacencies
- Prefix-to-SID mapping advertisements (mapping server)
- MPLS penultimate hop popping (PHP) and explicit-null signaling

SR OSPF Control Plane Summary

- OSPFv2 control plane
- Multi-area
- IPv4 Prefix Segment ID (Prefix-SID) for host prefixes on loopback interfaces
- Adjacency Segment ID (Adj-SIDs) for adjacencies
- Prefix-to-SID mapping advertisements (mapping server)
- MPLS penultimate hop popping (PHP) and explicit-null signaling

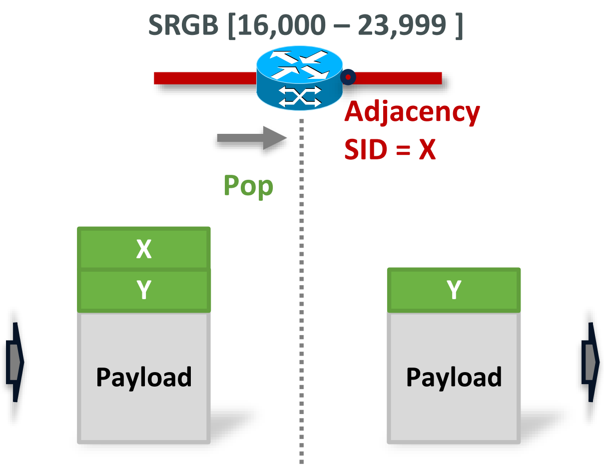
MPLS Data Plane Operation (Labeled)

Prefix SID



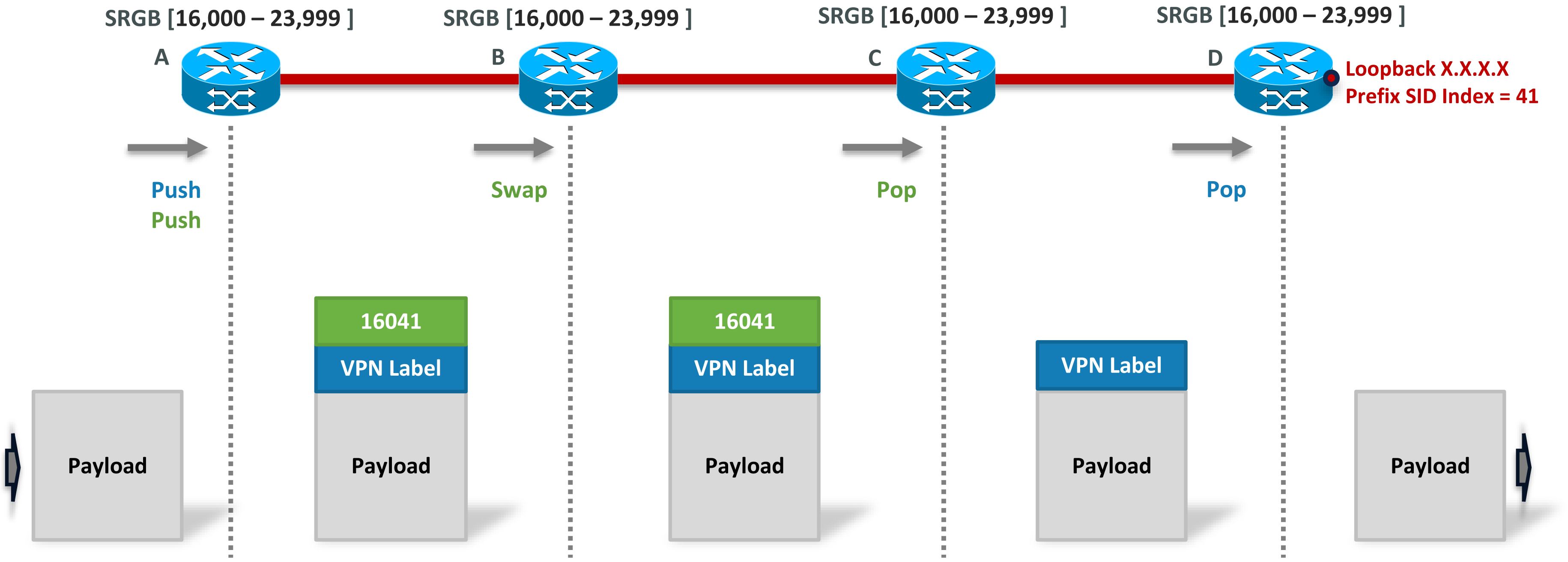
- Packet forwarded along IGP shortest path (ECMP)
- Swap operation performed on input label
- Same top label if same/similar SRGB
- PHP if signaled by egress LSR

Adjacency SID

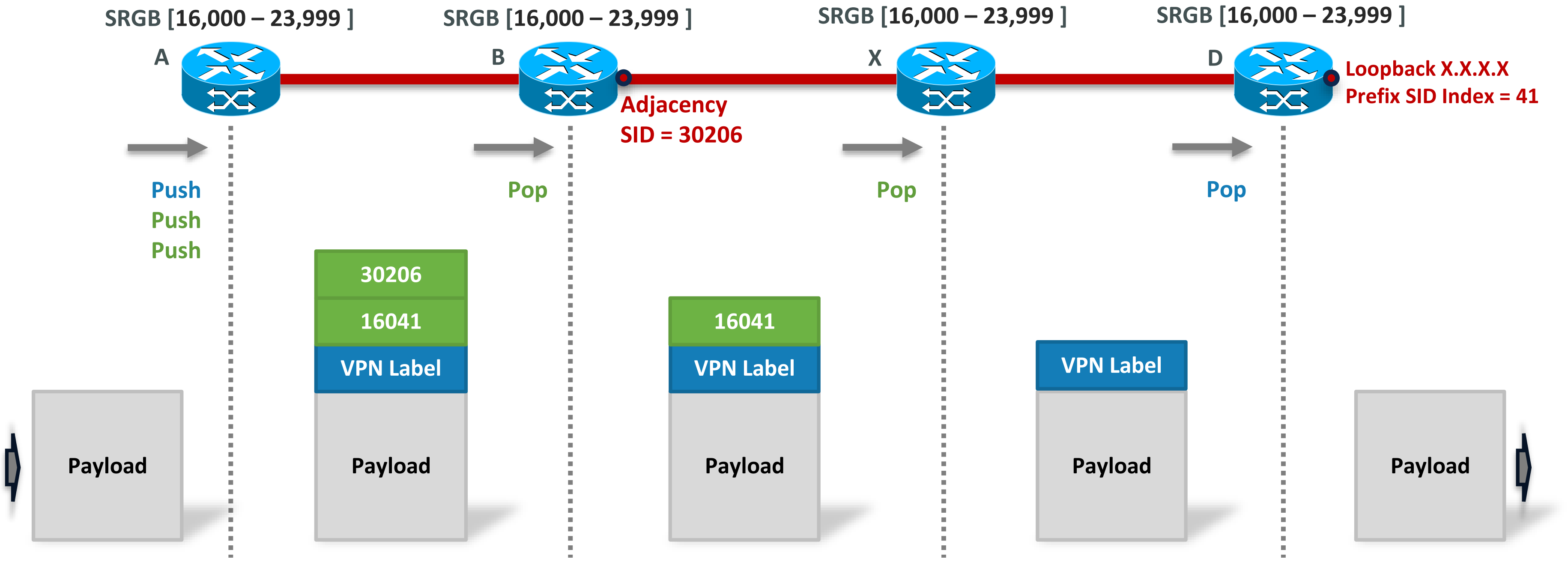


- Packet forwarded along IGP adjacency
- Pop operation performed on input label
- Top labels will likely differ
- Penultimate hop always pops last adjacency SID

MPLS Data Plane Operation (Prefix SID)



MPLS Data Plane Operation (Adjacency SID)

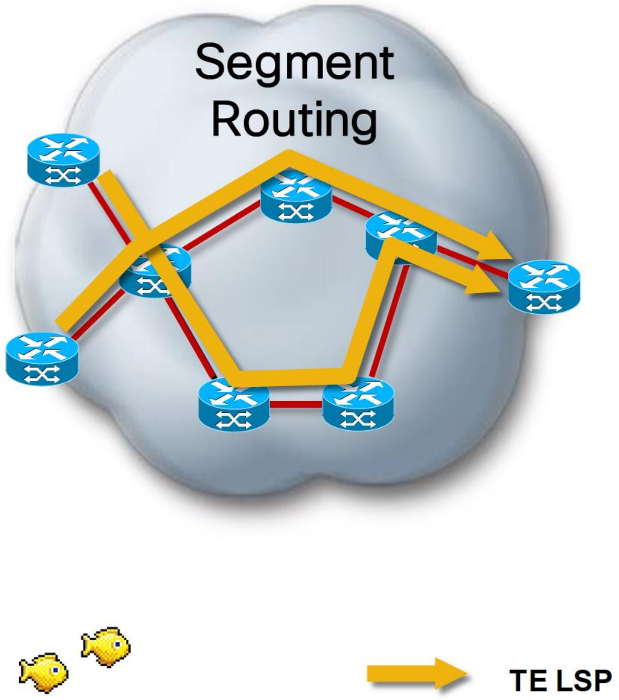


Segment Routing Traffic Engineering

You make the power of data possible

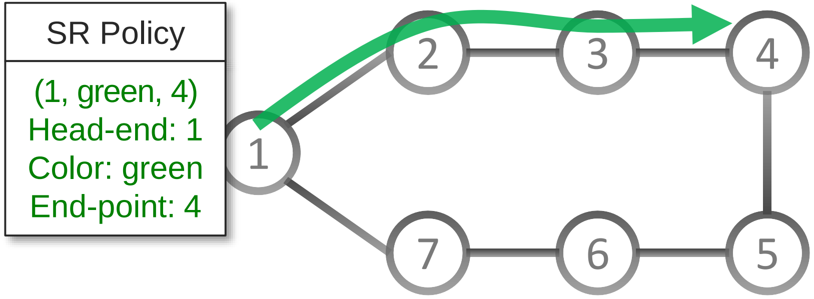
Traffic Engineering with SR

- Source based routing
  - State only at the Ingress PE
  - Supports constraint-based routing
- Uses existing ISIS / OSPF extensions
  - To advertise link attributes
- No RSVP-TE to establish LSPs
- ECMP Aware



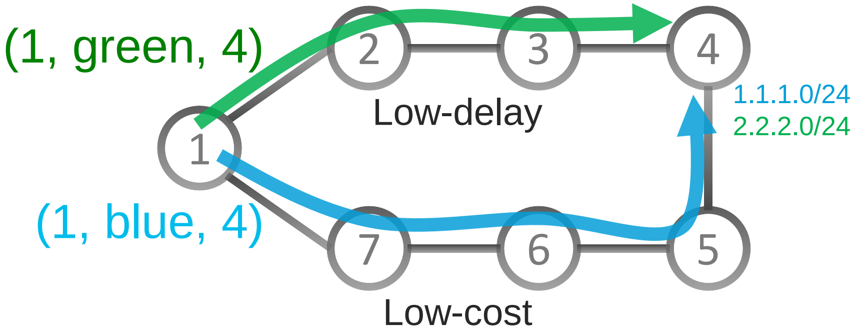
SR Policy

- SR-Policy is a Traffic Engineering intent by means of a solution SID-list.
- An SR-Policy is uniquely identified by a tuple:
  - Head-end: where the SR Policy has been instantiated (configured)
  - End-point: the destination of the SR Policy
  - Color: a numerical value to differentiate multiple SRTE Policies between the same pair of nodes



SR Policy Color

- Each SR Policy has a color
  - Color can be used to indicate a certain treatment (SLA, policy) provided by an SR Policy
- Only one SR Policy with a given color C can exist between a given node pair (head-end (H), end-point (E))
  - In other words: each SR Policy triplet (H, C, E) is unique
- Example:
  - Low-cost="blue", Low-delay="green"
  - steer traffic to 1.1.1.0/24 via Node4 into Low-cost SR Policy (1, blue, 4)
  - steer traffic to 2.2.2.0/24 via Node4 into Low-delay SR Policy (1, green, 4)



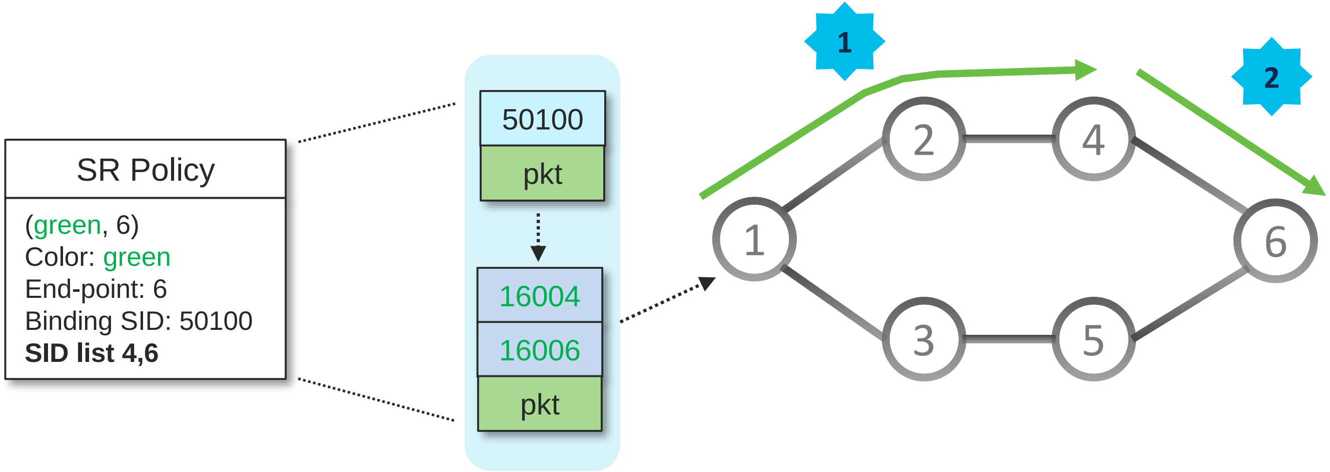
SR Policy on IOS-XR

- SR Policy Configuration on IOS-XR

```
segment-routing
 traffic-eng
  policy POLICY1
   color 20 end-point ipv4 1.1.1.4
   candidate-paths
    preference 100
     dynamic
      metric
       type igp
```

Binding SID

- The **Binding SID** is a fundamental building block of SR-TE solution
- Explicitly configured or dynamically allocated
- A Binding SID identifies a SRTE Policy
  - Packet received with Binding-SID as Top Label is steered into the SRTE Policy associated with the Binding-SID
  - Binding-SID label is popped, SRTE Policy's SID list is pushed



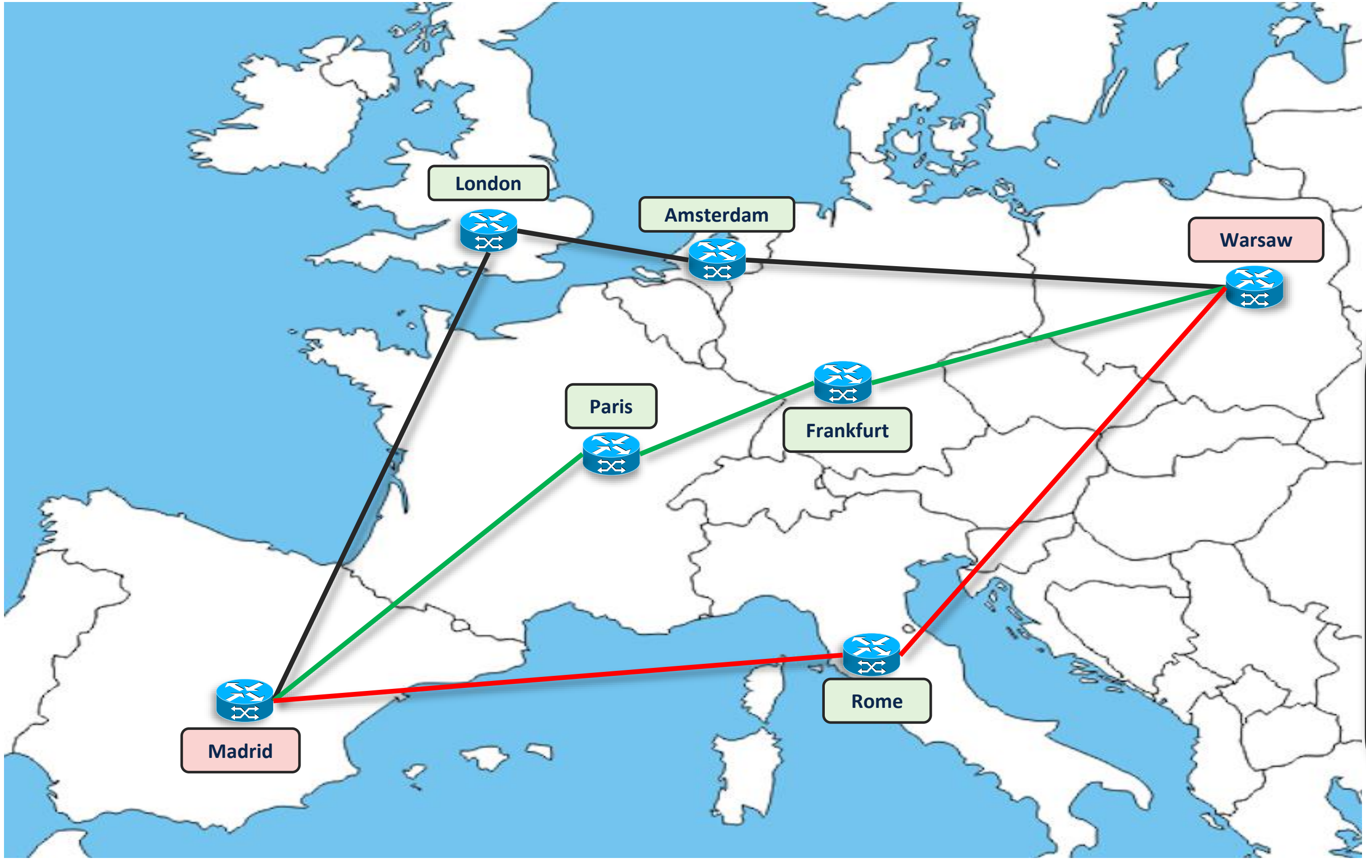
Binding SID Allocation

- Explicit allocation

```
segment-routing
 traffic-eng
  policy POLICY1
   color 20 end-point ipv4 1.1.1.6
   binding-sid mpls 50100
   candidate-paths
    preference 100
     dynamic
      metric
       type igp
```

- Dynamic allocation is the default

Practical Use-Cases (Optimization Based SR-TE Path)



- TE Affinity-bit (coloring)
- SRLG
- Delay
- Disjoint Path

High Bandwidth Path:
Madrid – London – Amsterdam – Warsaw

Secure / Encrypted Path (MacSec enabled):
Madrid – Paris – Frankfurt – Warsaw

Low Delay Path:
Madrid – Rome – Warsaw

Disjoint Path:
1. MAD – PAR – FFM – WAW
2. MAD – RME – WAW
3. MAD – LON – AMS - WAW

Practical Use-Cases (SR Policy Configuration)

MacSec enabled Path

```
segment-routing
 traffic-eng
  affinity-map
   !! 32-bit maps
   name macsec bit-position 0
  !
  interface HundredGig0/0/0/0
   affinity name macsec
  !
  policy POLICY1
      color 20 end-point ipv4 1.1.1.4
      candidate-paths
        preference 100
          dynamic
            metric type igp
          constraints
            affinity
              include-all name macsec
```

Low Delay Path

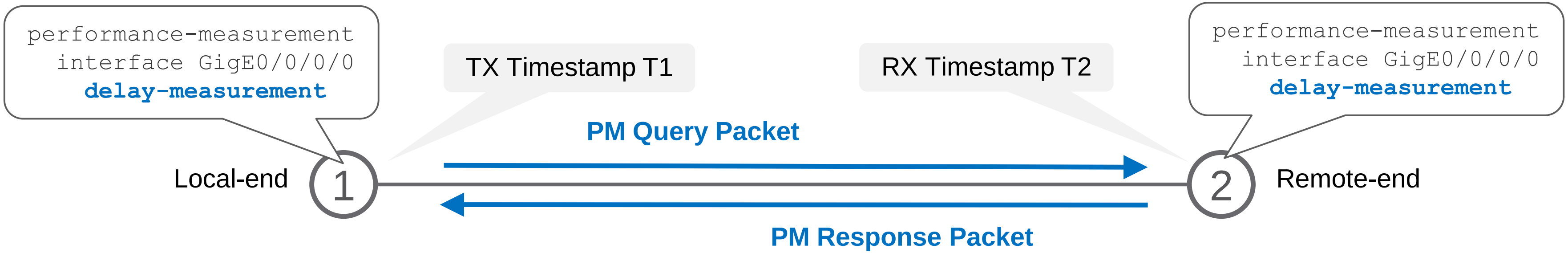
```
segment-routing
 traffic-eng
 !
  policy POLICY2
      color 30 end-point ipv4 1.1.1.4
      candidate-paths
        preference 110
          dynamic
            metric
              type delay
```

Disjoint Path

```
segment-routing
 traffic-eng
 !
  policy POLICY3
   color 10 end-point 1.1.1.4
   candidate-paths
    preference 120
     dynamic
      pcep
      metric type te
     constraints
       disjoint-path group-id 1 type node
```

Delay Measurement

- Link Delay (using Probes) Measurement feature

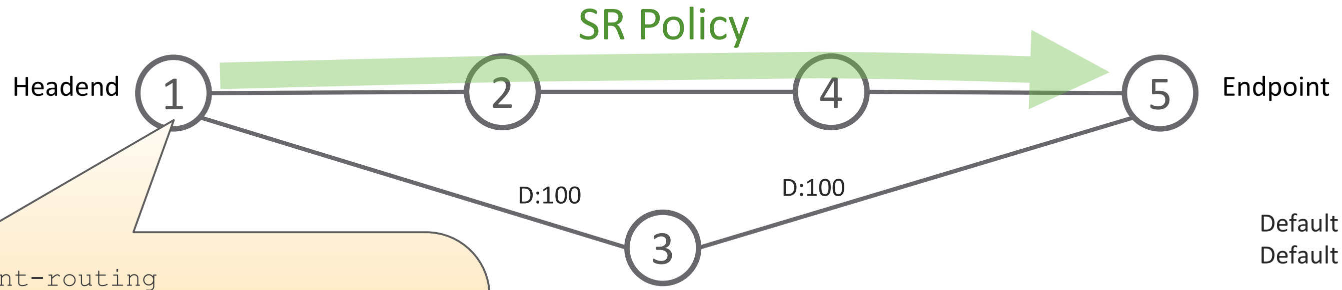


- One Way Delay = (T2 – T1)

Default: every 3 sec

- Timestamps added in hardware
- PM Query format: RFC 6374 (MPLS/GAL) or RFC 5357 (IP/UDP/TWAMP)

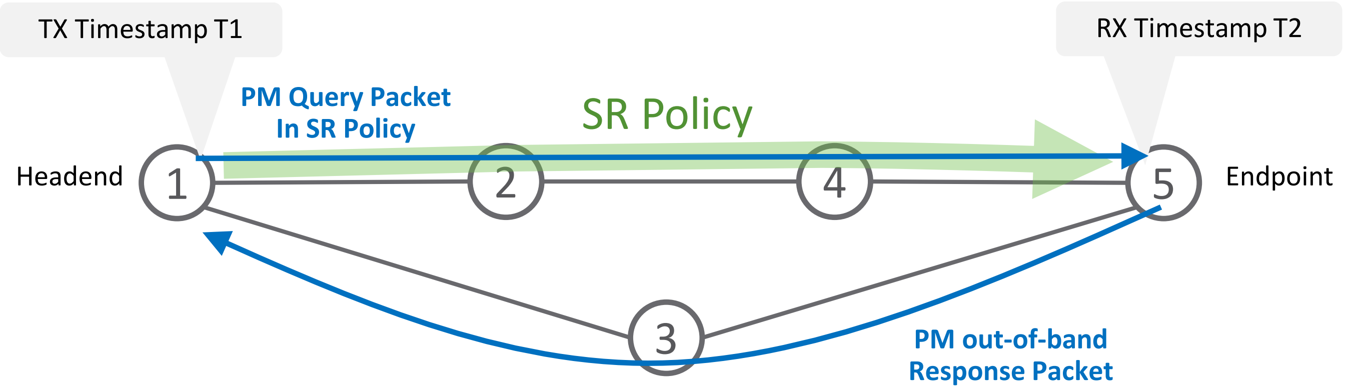
Per SR-Policy Delay Measurement




```
segment-routing
 traffic-eng
  policy FOO
   color 20 end-point ipv4 1.1.1.5
   delay-measurement
   candidate-paths
    preference 100
     dynamic
      metric
       type delay
```

Default ISIS metric: 10
Default ISIS delay (D): 10

Probe Measurement



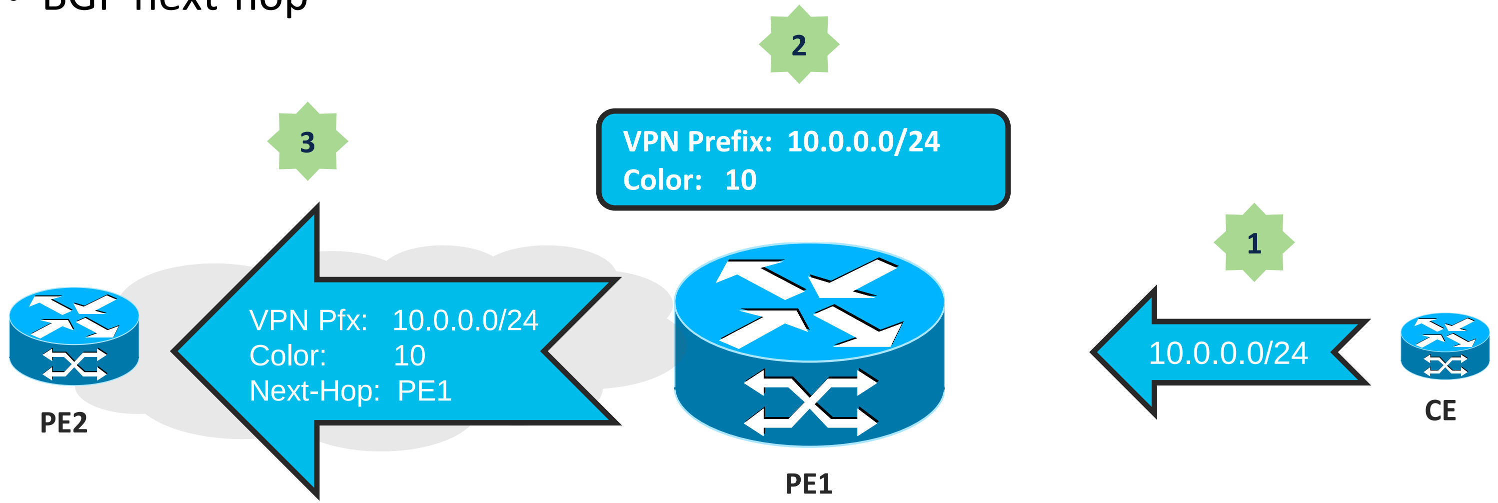
- One Way Delay = (T2 – T1)
  - Requires clock synchronization
- Default: Send Query every 3 sec

Automated Steering

You make customer experience **possible**

Automated Steering

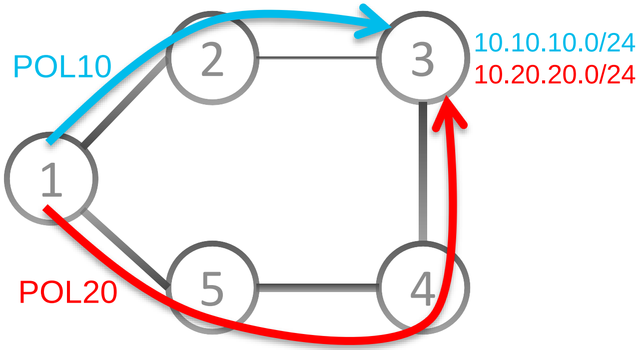
- Consists of:
  - Prefix / Customer routes (L2/L3 VPN)
  - Color (BGP extended community attribute)
  - BGP next-hop



Automated Steering

- BGP can automatically steer traffic into an SR Policy based on BGP next-hop and color of a route
  - Color of a route is specified by its color extended community attribute
- When a BGP next-hop and color of a route match the end-point and color of an SR Policy, then BGP installs the route resolving on the BSID of the SR Policy

10.10.10.0/24 (color 10, NH 1.1.1.3)
via SR Policy POL10 (10, 1.1.1.3)
10.20.20.0/24 (color 20, NH 1.1.1.3)
via SR Policy POL20 (10, 1.1.1.3)

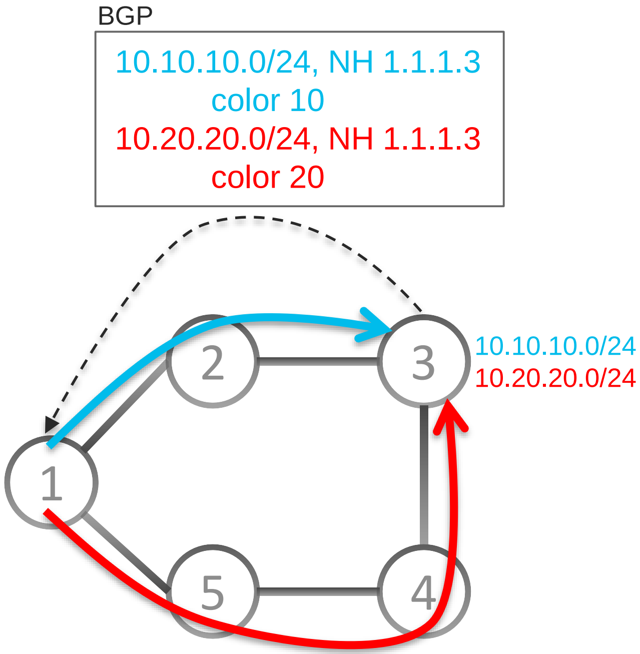


Automated Steering – Setting Color of a Route

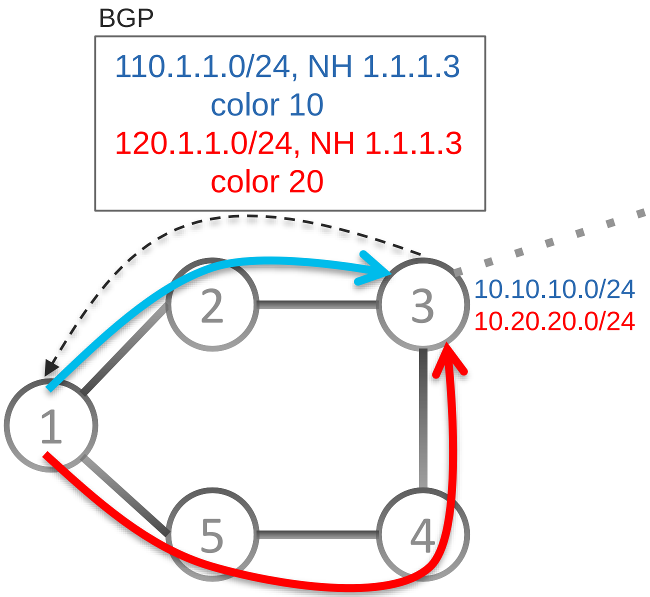
- The color extended community is specified in **RFC 5512** and updated in *draft-previdi-idr-segment-routing-te-policy*
- The color of a BGP route is typically set at the egress PE by adding a color extended community to the route
  - The color extended community is propagated to the ingress PE
  - Traffic steering on the ingress PE is then done automatically based on the color

Automated Steering – Color Assignment on PE

- Node1 has two SR Policies with end-point Node3:
  - POL10 with color 10 (blue) via Node2
  - POL20 with color 20 (red) via Node4
- Node3 advertises two prefixes with next-hop 1.1.1.3 in BGP:
  - 10.10.10.0/24 with color 10 (blue)
  - 10.20.20.0/24 with color 20 (red)



Automated Steering – Color Assignment Egress PE



Node3

```
extcommunity-set opaque BLUE
  10
end-set
!
extcommunity-set opaque RED
  20
end-set
!
route-policy SET_COLOR
  if destination in (10.10.10.0/24) then
    set extcommunity color BLUE
  endif
  if destination in (10.20.20.0/24) then
    set extcommunity color RED
  endif
end-policy
!
router bgp 1
 neighbor 1.1.1.1
  remote-as 1
  update-source Loopback0
  address-family ipv4 unicast
   route-policy SET_COLOR out
```

On-Demand Next-Hop (ODN)

You make multi-cloud **possible**

On-Demand Next-Hop (ODN)

- A head-end router automatically instantiates an SR Policy to a BGP next-hop when required (on-demand)
- Color community is used as SLA indicator
- Reminder: an SR Policy is defined (color, end-point)

BGP Color Community

BGP Next-hop

- Automated Steering (AS) automatically steers the BGP traffic into this SR Policy, also based on next-hop and color.

On-Demand SR Policy

- Configure an SR Policy template for each color for which on-demand SR Policy instantiation is desired
- An example with two color templates configured:
  - color 10 for high bandwidth (optimize IGP metric)
  - color 20 for low-delay (optimize link-delay metric)

```
segment-routing
 traffic-eng
  on-demand color 10
   dynamic
    metric type igp
  !
  on-demand color 20
   dynamic
    metric type delay
```

SR Policy template High-BW (color 10)

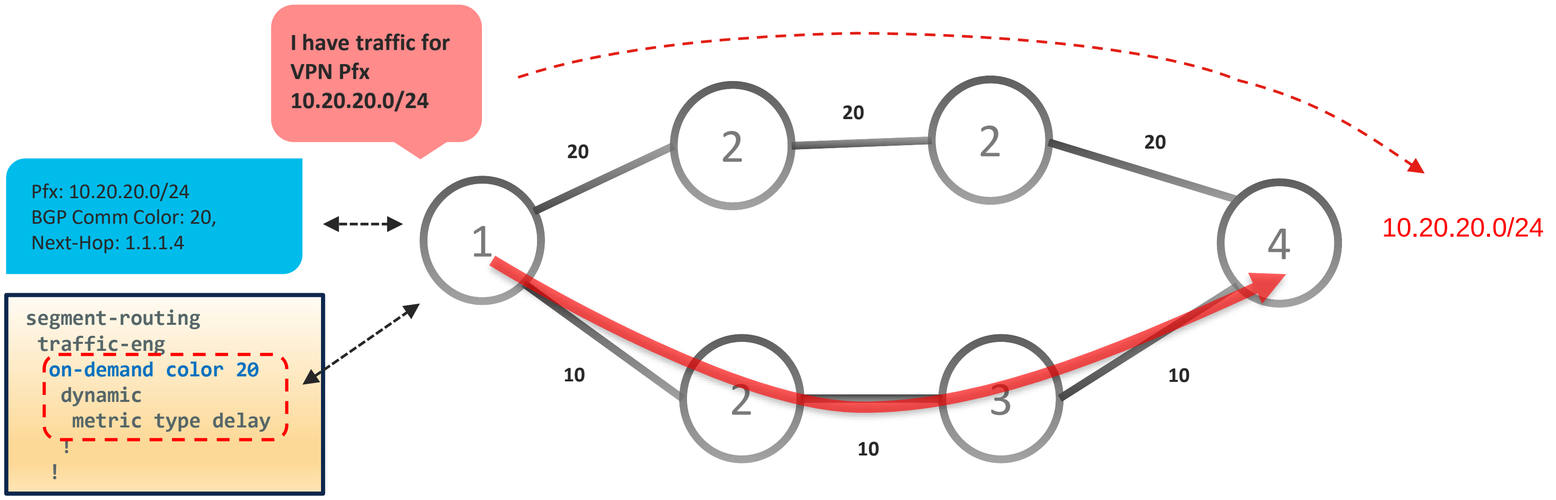
SR Policy template Low-Delay (color 20)

ODN Benefits

- SLA-aware BGP service
- No a-priori full-mesh of SR Policy configuration
  - 3 to 4 common optimization templates are used throughout the network
    - color → optimization objective
- No complex steering configuration
  - Automated Steering of BGP routes on the right SLA path
  - Data plane performant
  - BGP PIC FRR data plane protection is preserved
  - BGP NHT fast control plane convergence is preserved

ODN Summary

The ODN functionality is only triggered when receiving a service route with an authorized color. A color is authorized when an on-demand template is configured for that color.



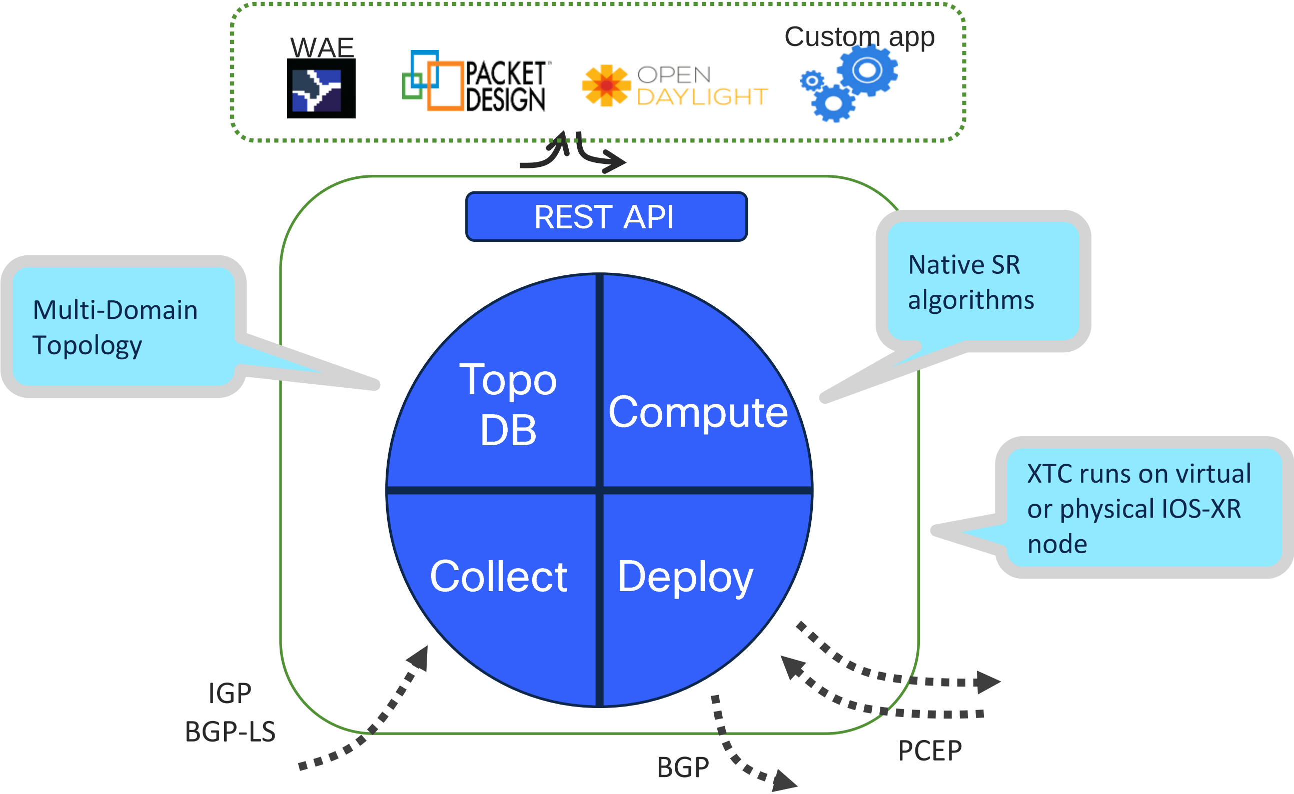
SR-PCE

You make customer experience **possible**

What about inter-domain path calculation?

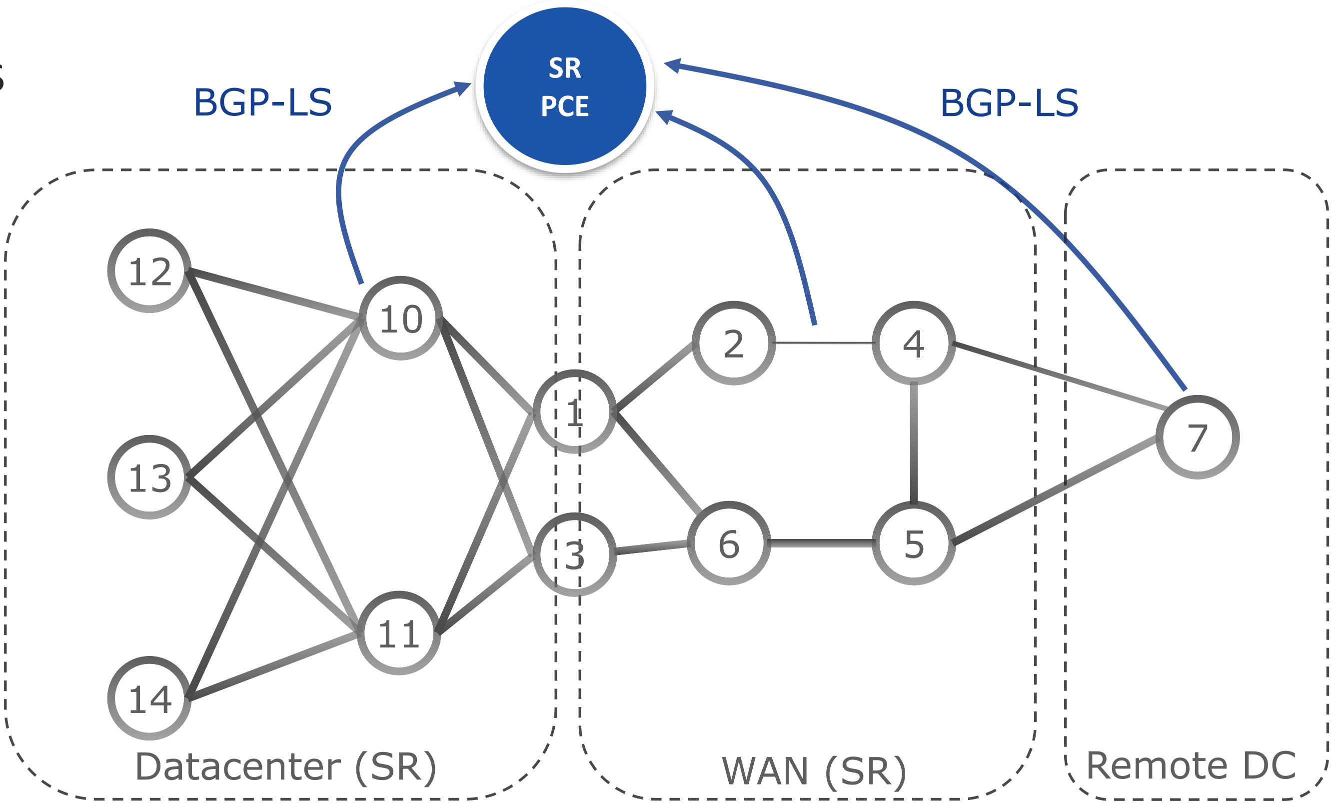
- SR-PCE is an IOS XR multi-domain stateful SR Path Computation Element (PCE)
  - IOS XR: XTC functionality is available on any physical or virtual IOS XR node, activated with a single configuration command
  - Multi-domain: Real-time reactive feed via BGP-LS/ISIS/OSPF from multiple domains; computes inter-area/domain/AS paths
  - Stateful: takes control of SRTE Policies, updates them when required

SR-PCE Building Blocks



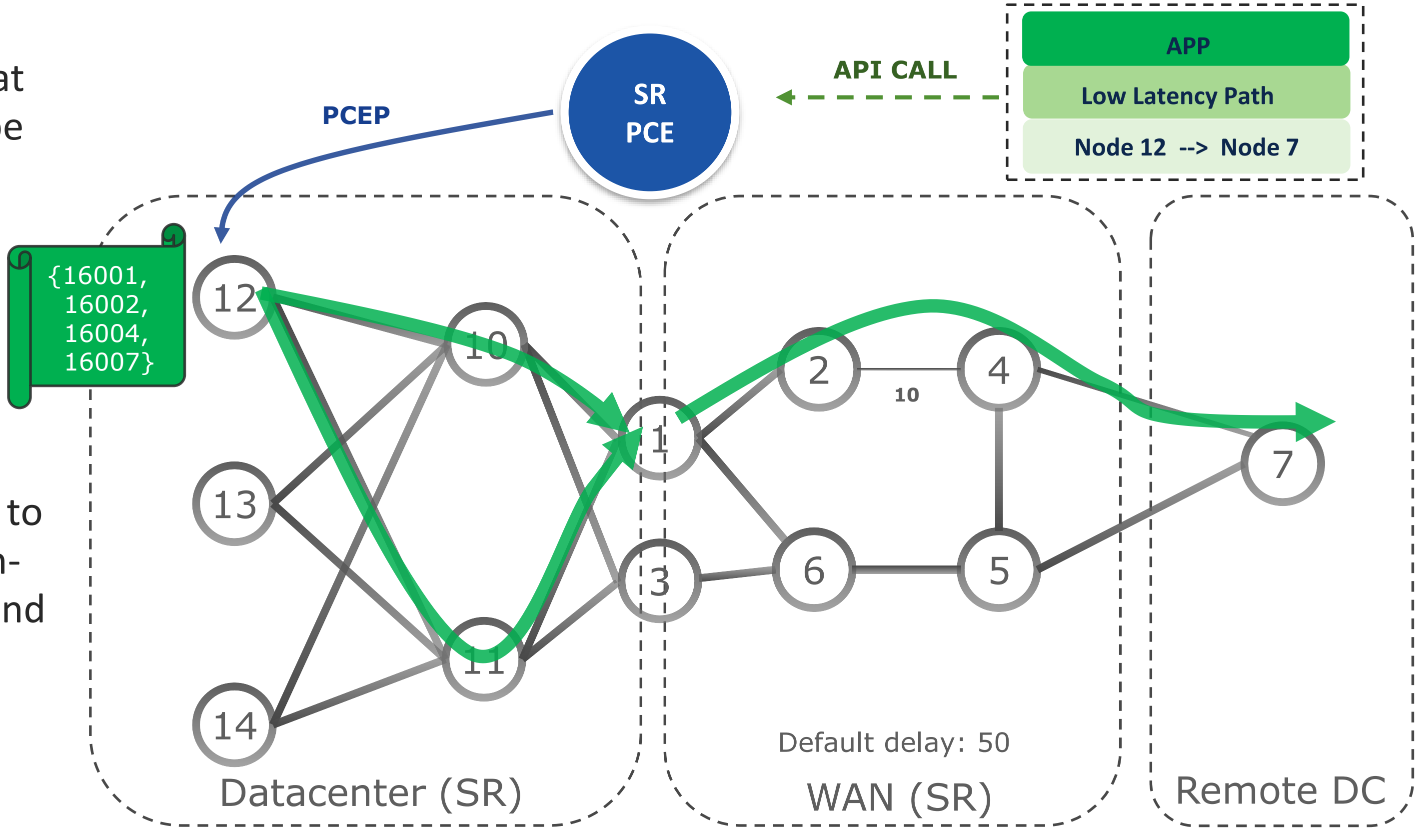
SR-PCE Controller

- SR PCE collects via BGP-LS
  - IGP segments
  - BGP segments
  - Topology



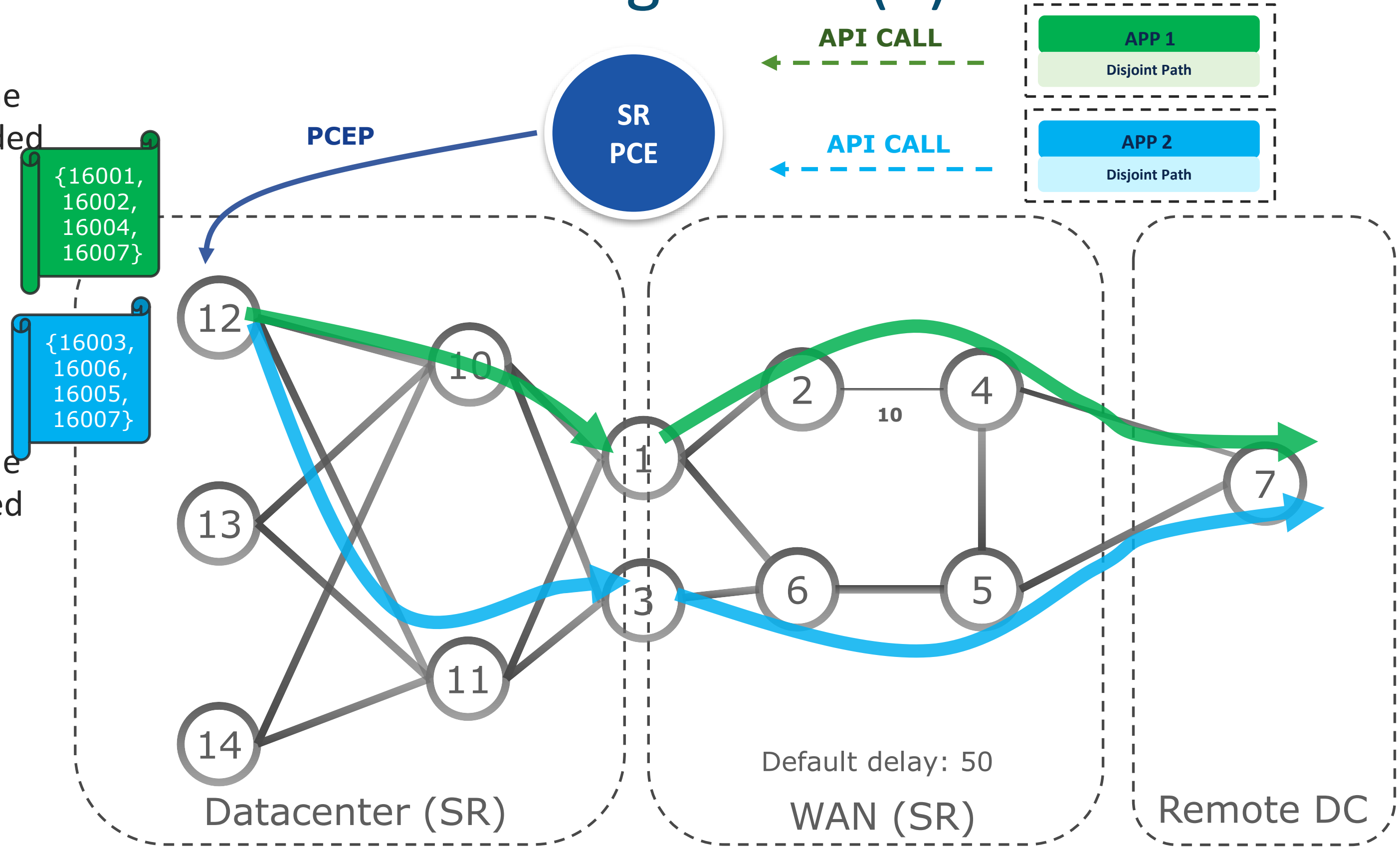
An End-to-end Path as a List of Segments (1)

- SR PCE computes that the green path can be encoded as
  - 16001
  - 16002
  - 16004
  - 16007
- SR PCE programs a single per-flow state to create an application-engineered end-to-end policy



An End-to-end Path as a List of Segments (2)

- SR PCE computes that the **green** path can be encoded as
  - 16001
  - 16002
  - 16004
  - 16007
- SR PCE computes that the **blue** path can be encoded as
  - 16003
  - 16006
  - 16005
  - 16007



Flexible Algorithm (Flex-Algo)

You make networking **possible**

Flex-Algo

- Complements the SRTE solution by adding new Prefix-Segments with specific optimization objective and constraints
  - Minimize igp-metric or delay or te-metric
  - Avoid SRLG or affinity
- Leverages the SRTE benefits of simplicity and automation:
  - Automated sub-50msec FRR (TILFA)
  - On-Demand Policy (ODN)
  - Automated Steering (AS)

Flex-Algo Definition

- The definition of an **"Algorithm"** is defined by the operator, on a per-deployment basis
- Flex-Algo 'K' is defined as
  - The specification of minimal metric: IGP, delay, …
  - The exclusion of certain link properties: link-affinity, SRLG, …
- Each node MUST advertise Flex-Algo(s) that it is participating in

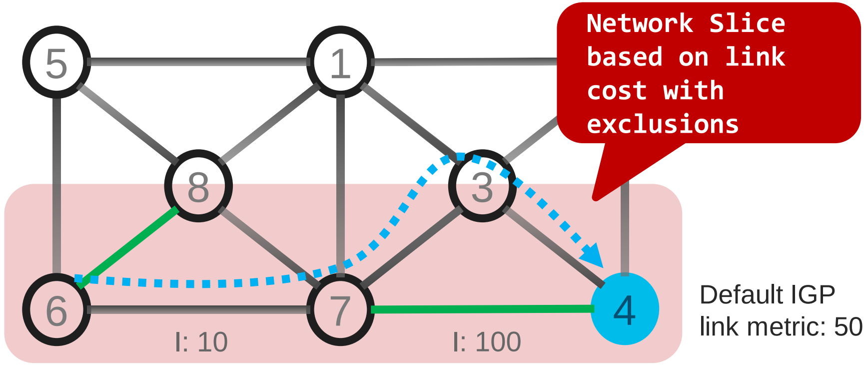
Currently Defined Algorithms

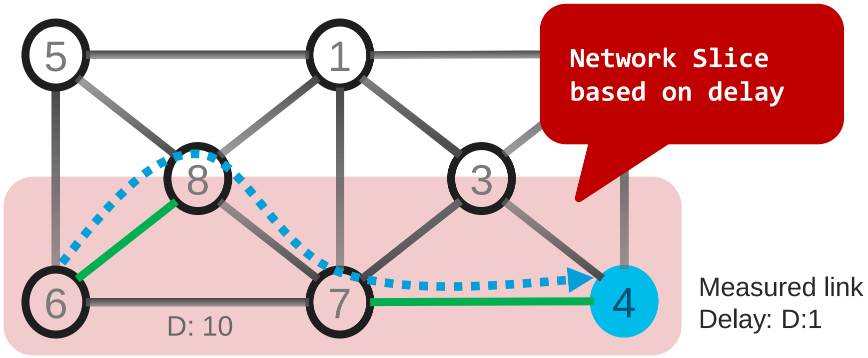
- 0: Shortest Path First (SPF) algorithm based on link metric.
  - This is the well-known shortest path algorithm as computed by the IS-IS Decision process. Consistent with the deployed practice for link-state protocols, algorithm 0 permits any node to overwrite the SPF path with a different path based on local policy

- 1: Strict Shortest Path First (SPF) algorithm based on link metric.
  - The algorithm is identical to algorithm 0 but algorithm 1 requires that all nodes along the path will honor the SPF routing decision. Local policy MUST NOT alter the forwarding decision computed by algorithm 1 at the node claiming to support algorithm

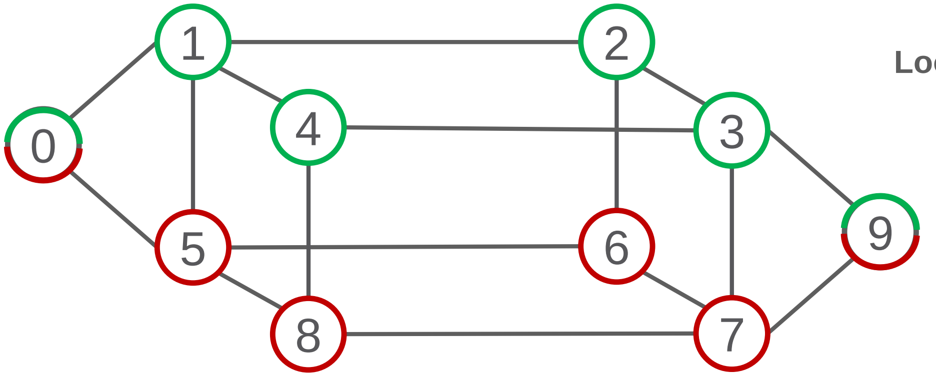
Flex-Algo Definition

- Example:
  - Operator-1 defines Flex-Algo 128 as ***“minimize IGP metric and avoid link-affinity ‘green’”***
  - Operator-2 defines Flex-Algo 128 as ***“minimize delay metric”***





Use Case: Multi-Plane Networks



Loopback 1.1.1.9/32

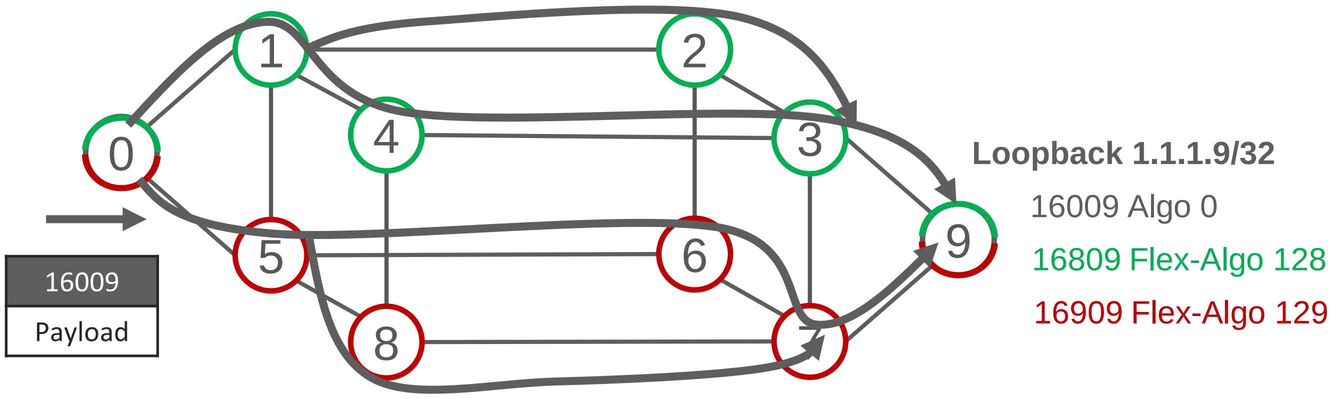
16009 Algo 0

16809 Flex-Algo 128

16909 Flex-Algo 129

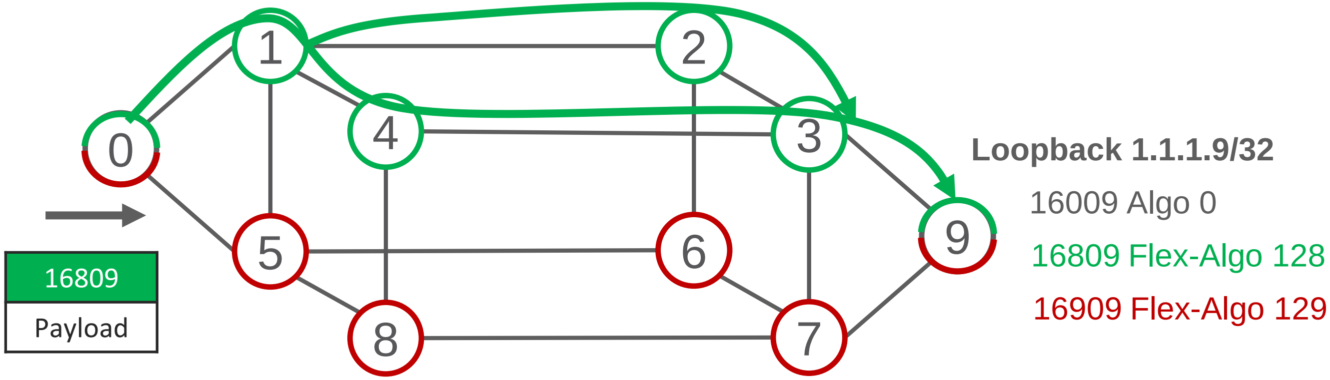
- All the nodes support Algo 0: minimize IGP metric
- Green nodes also support 128: minimize IGP metric
- Red nodes also support 129: minimize Delay

Use Case: Multi-Plane Networks (cont)



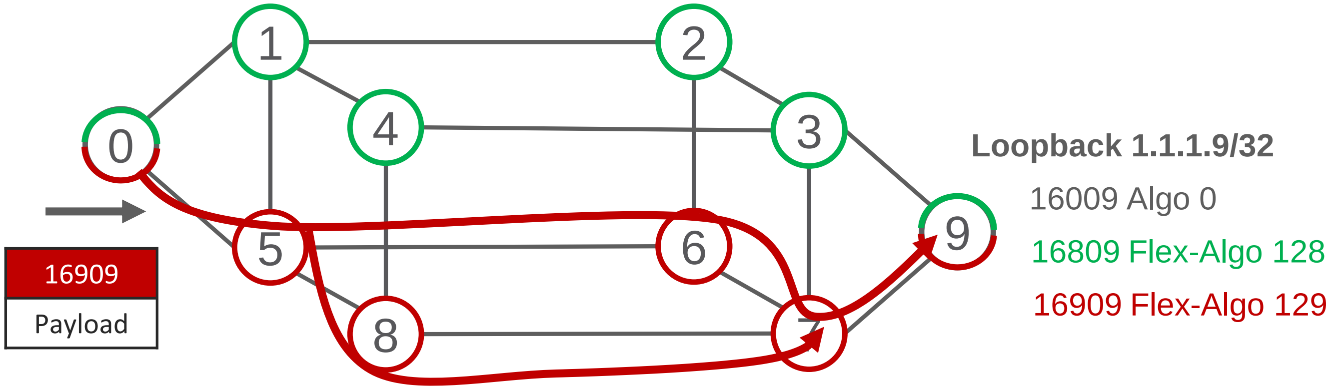
- Path to Node 9 across Algo 0
- ECMP based forwarding across all Planes

Use Case: Multi-Plane Networks (cont)



- Path to Node 9 across Flex-Algo 128
- ECMP based forwarding WITHIN green Plane

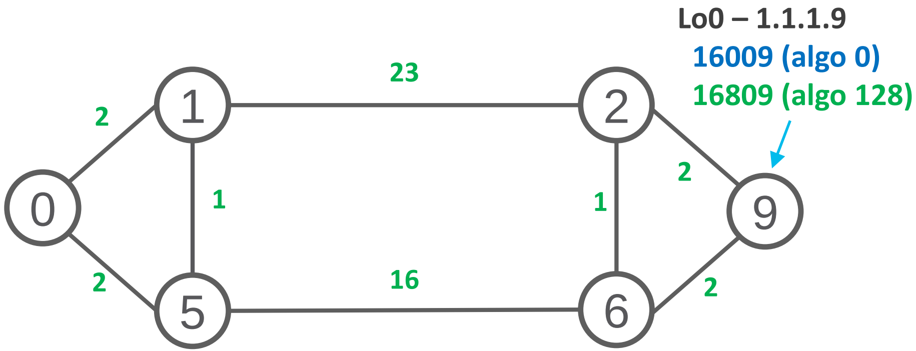
Use Case: Multi-Plane Networks (cont)

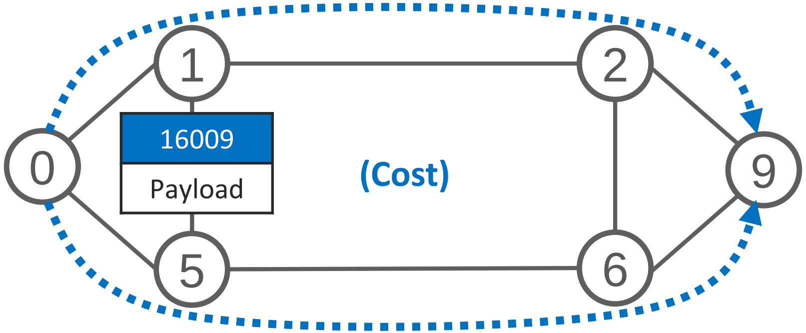


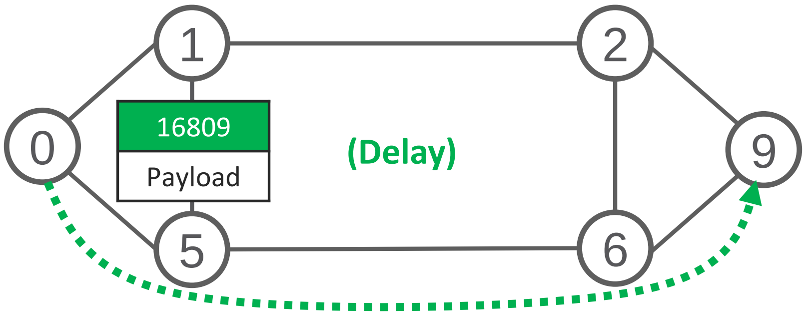
- Path to Node 9 across Flex-Algo 129
- ECMP based forwarding WITHIN red Plane

Use Case: Delay vs Cost of Transport

- All nodes support Algo 0 & 128
- ISIS link metric 10
- Algo 128: minimize delay metric
- Per-link measurement of delay and advertisement as delay metric via ISIS
- Delay metric at that time shown in green

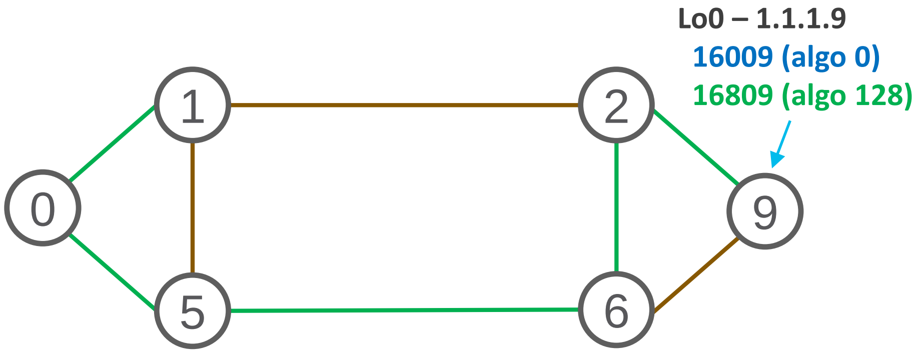


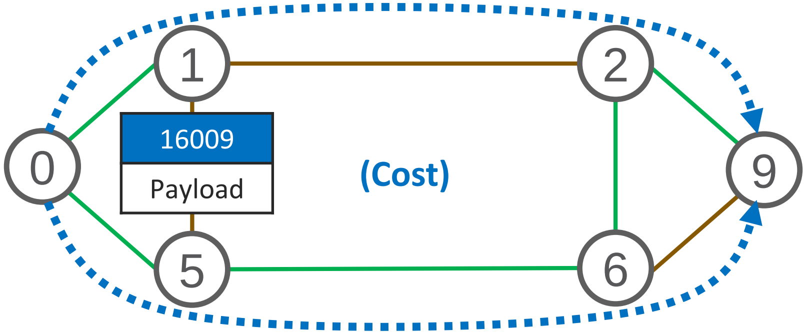


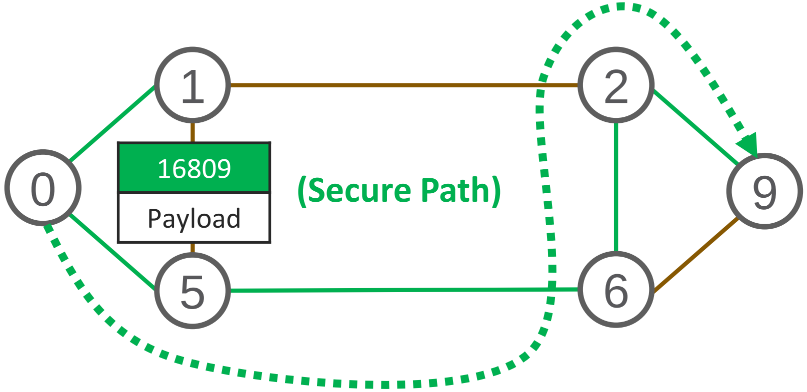


Use Case: SRTE for Secure Paths

- ISIS link metric 10
- Link colors shown **Unencrypted** / **Encrypted**
- All nodes support Algo 0 & 128
- Algo 128: minimize IGP while traversing links with encryption enabled (**exclude brown**)
- Per-link colors flooded in IGP







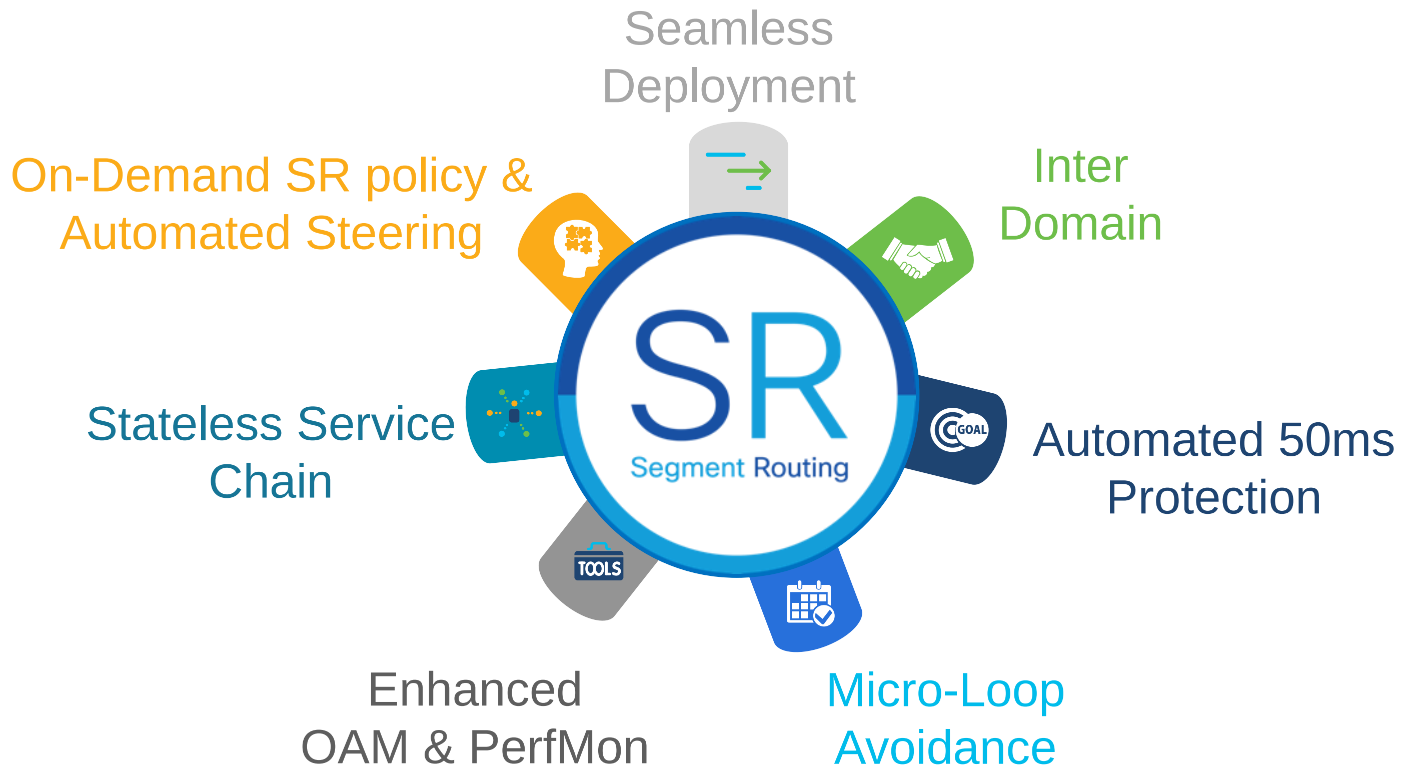
Flex-Algo Benefits

- **Minimize label stack, and maximize ECMP at midpoints:**
  - Single SID needed to enforce traffic towards a specific Path/Plane
  - ECMP if possible (cost related) on that specific Path/Plane
- **Slicing of the network on a per:**
  - Latency
  - Bandwidth
  - Secure links – MACSec
- **TI-LFA aware**
  - Protected path stays in Flex-Algo virtual topology

Conclusion

You make networking **possible**

SR Unified Fabric Attributes



Industry at Large Backs up SR

Strong customer adoption
WEB, SP, DC, Metro, Enterprise

De-facto SDN Architecture

Standardization
IETF

Multi-vendor Consensus

Open Source
Linux, VPP

Thank you

You make **possible**