Let's go

#CiscoLiveAPJC

The bridge to possible

A Guide to a Successful Segment Routing Deployment

Jose Liste, Technical Marketing Engineer

BRKSPG-2510

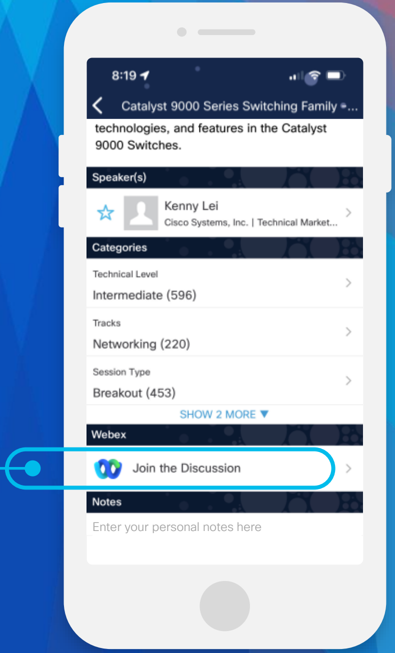
Cisco Webex App

Questions?

Use Cisco Webex App to chat
with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space



Webex spaces will be moderated by the speaker until December 22, 2023.

https://ciscolive.ciscoevents.com/ciscolivebot/#BRKSPG-2510

Abstract

- Segment Routing is a de-facto industry standard architecture adopted by operators of all sizes.
- It delivers an end-to-end policy-aware network at scale and simplicity over a stateless IP fabric based on MPLS or IPv6 data-plane.
- In this session, you will learn the key technical building blocks, deployment considerations, migration strategies and best practices for a successful rollout of SR in your network.
- Understanding of SR fundamentals as well as MPLS and IPv6 is recommended.

Agenda

- Introduction
- SRGB planning
- BGP-SR
- SRTE / SR-PCE
- Flexible Algorithm
- SRv6 uSID
  - Locator Addressing
  - Migration
- Conclusion

Before We Get Started

- Basic SR knowledge is required
  - MPLS data plane
  - IPv6 data plane
- Focus is IOS-XR
  - Latest and greatest
  - Majority of examples leverage ISIS as IGP
- Stay up-to-date

http://www.segment-routing.net/

Network Evolution with Segment Routing

One Architecture / Two Data-Plane instantiations

Segment Routing

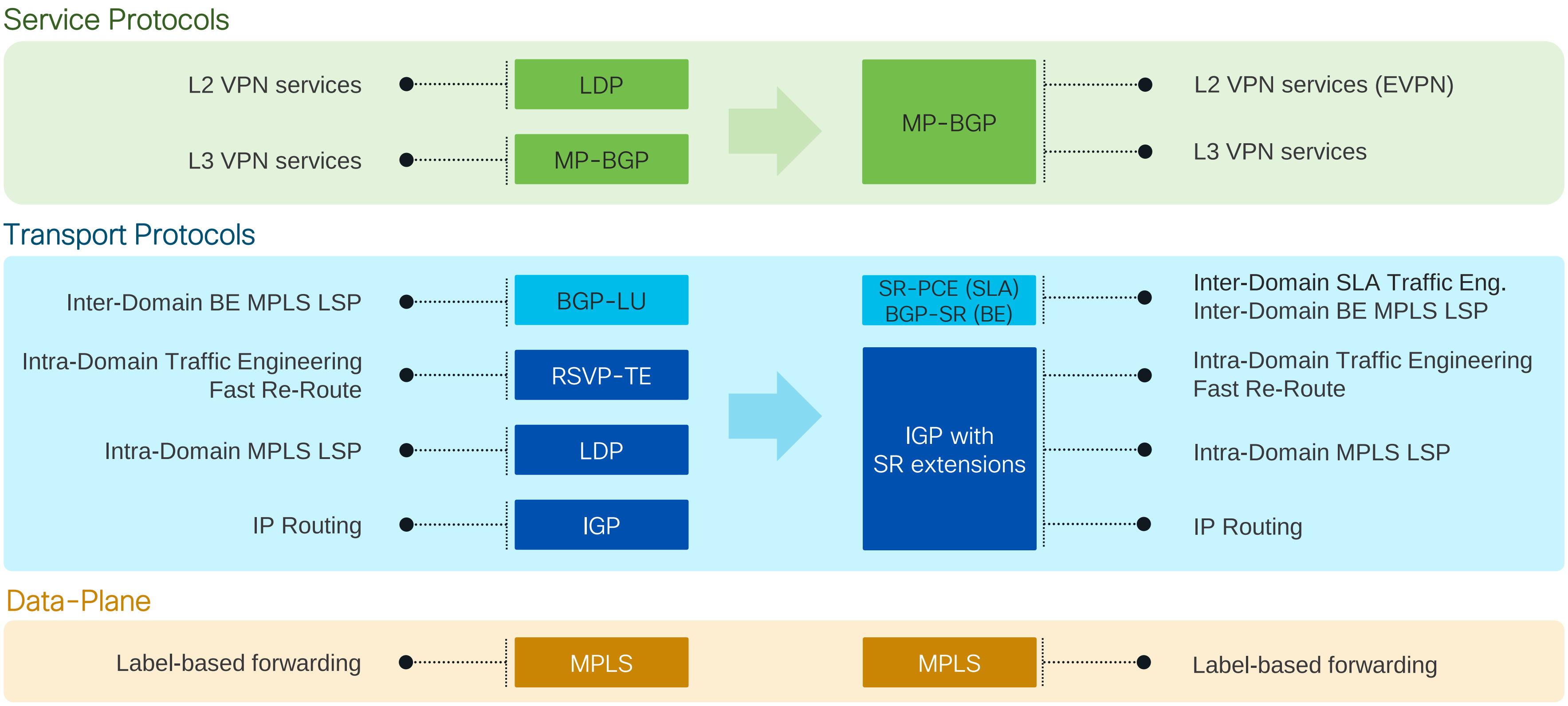
SR-MPLS

- Instantiation of SR on the MPLS data plane
- A segment is encoded with an MPLS label

SRv6

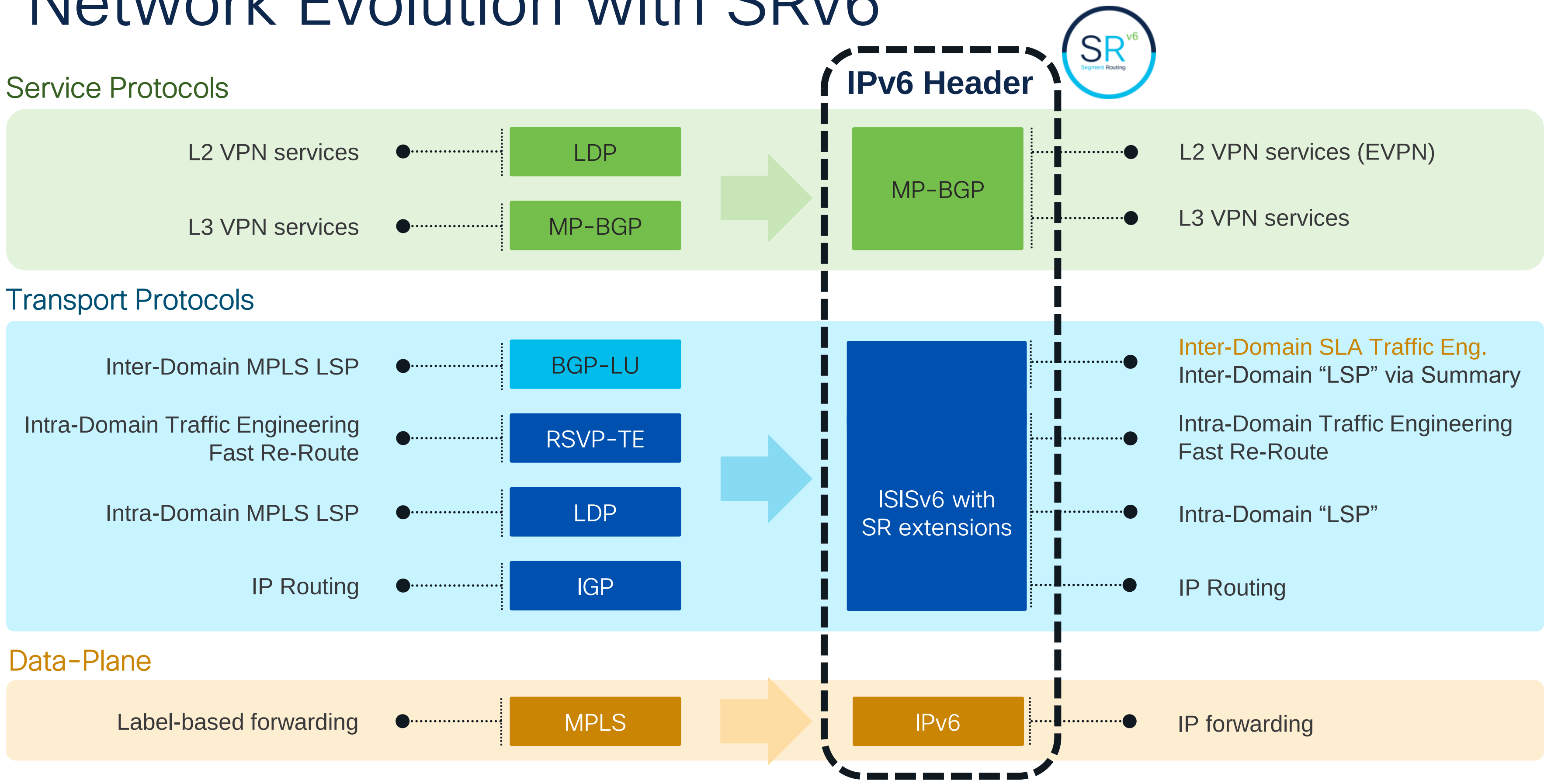
- Instantiation of SR on the IPv6 data plane
- One or more segments are encoded with an IPv6 address

Network Evolution with SR-MPLS



LDP: Label Distribution Protocol, **MP-BGP**: Multi-protocol BGP, **BGP-LU**: BGP Labeled-Unicast, **PCE**: Path Computation Element, **RSVP-TE**: Reservation Protocol Traffic Engineering

Network Evolution with SRv6



LDP: Label Distribution Protocol, **MP-BGP**: Multi-protocol BGP, **BGP-LU**: BGP Labeled-Unicast, **RSVP-TE**: Reservation Protocol Traffic Engineering

SR Global Block (SRGB)

Segment Routing Global Block (SRGB)

- Segment Routing Global Block
  - Range of labels reserved for Segment Routing Global Segments
  - Default Cisco's SRGB is 16,000 – 23,999
- A prefix-SID is advertised as a domain-wide unique index
- The Prefix-SID index points to a unique label within the SRGB
  - Index is zero based, i.e. first index = 0
  - Label = Prefix-SID index + SRGB base
  - E.g. Prefix 1.1.1.65/32 with prefix-SID index 65 gets label 16065

SRGB label range preservation

- LSD preserves the default SRGB label range [16,000-23,999]
  - In any Segment Routing capable software release
  - Even if Segment Routing is not enabled
  - Except if the configured mpls label range includes this default range
- LSD allocates dynamic labels starting from 24,000

SRGB label range preservation

- Preservation of the default SRGB label range makes future Segment Routing activation possible without reboot
  - No labels are allocated from that preserved range. When enabling Segment Routing with default SRGB some time in the future, that label range is available and ready for use

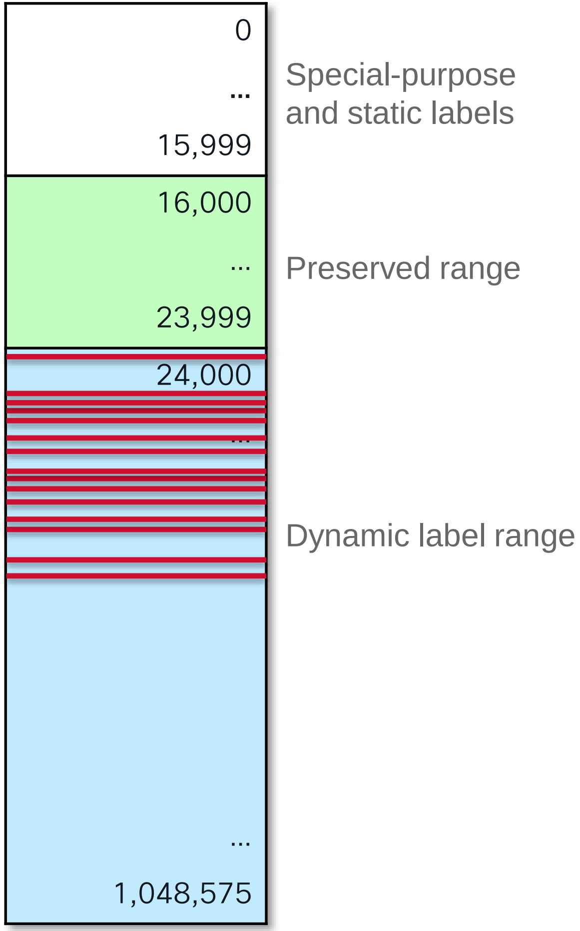
Segment Routing Global Block (SRGB) Notes

- Modifying a SRGB configuration is disruptive for traffic
  - And may require a reboot if the new SRGB is not (entirely) available
  - Allocating a non-default SRGB in the upper part of the MPLS label space increases the chance that the labels are free

LSD SRGB allocation - Example

- An example sequence of Segment Routing activation:

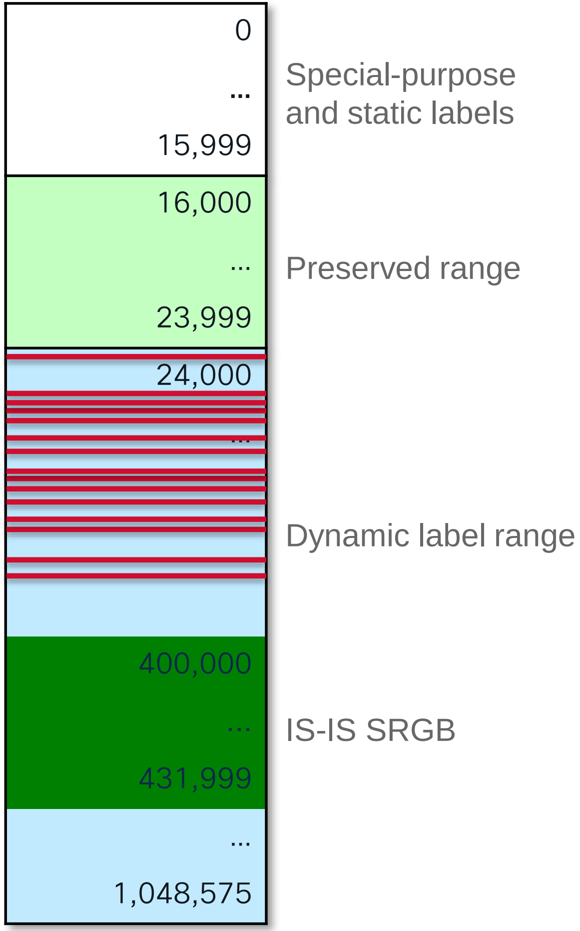
1. No Segment Routing enabled, no SRGB allocated
   - LSD preserves default SRGB label range
   - Dynamic labels are allocated by various MPLS applications (— in diagram)



LSD SRGB allocation - Example

- An example sequence of Segment Routing activation:

1. No Segment Routing enabled, no SRGB allocated
   - LSD preserves default SRGB label range
   - Dynamic labels are allocated by various MPLS applications (— in diagram)
2. Sometime later, SR IS-IS is enabled with non-default SRGB in the upper label range (hence likely unused)
   - SRGB label range is free, start using SR without reboot!



Segment Routing Global Block (SRGB)

Non-default SRGB Example

```
segment-routing
 global-block 18000 19999
!
router isis 1
 address-family ipv4 unicast
  segment-routing mpls
```

Configure a non-default SRGB
18,000 – 19,999

```
RP/0/0/CPU0:xrvr-1#show mpls label table detail
Table Label   Owner                            State
----- ------- -------------------------------- ------
<...snip...>
0     18000   ISIS(A):1                        InUse  No
  (Lbl-blk SRGB, vers:0, (start_label=18000, size=2000)
0     24000   ISIS(A):1                        InUse  Yes
  (SR Adj Segment IPv4, vers:0, index=1 type=0, intf=( /0/0/0, nh=10.0.0.2)
```

IS-IS SRGB

Start_label = 18,000

Size = 2,000

Non-default SRGB label block allocation for ISIS
[18,000 – 19,999]

SRGB Design Recommendations

SRGB Configuration

- The SRGB can be configured
  - Globally (Recommended)
    - By default, all IGP instances and BGP use this global SRGB
  - Per-IGP (Not Recommended)

```
segment-routing
 global-block 18000 19999
```

Recommended

```
router isis 1
 segment-routing global-block 18000 19999
```

Not Recommended

SRGB design

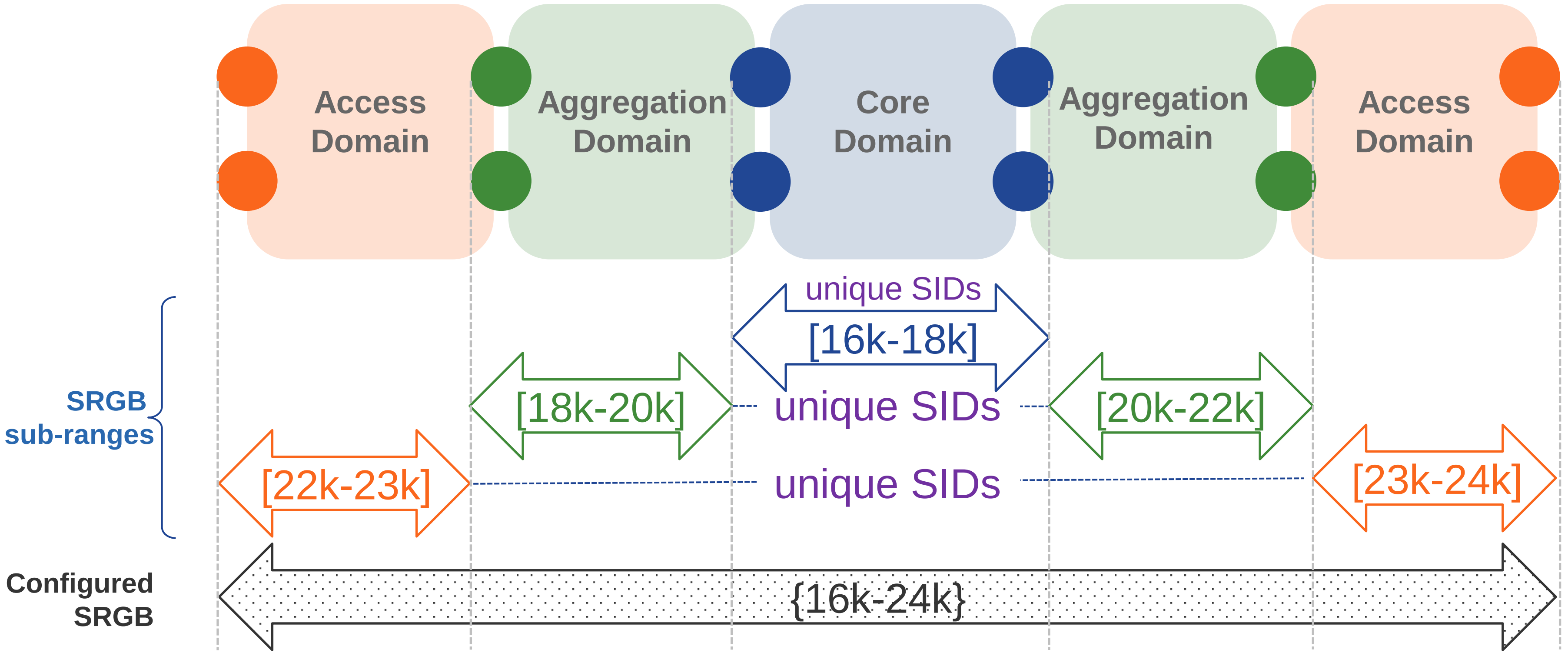
- SRGB planning should aim for the following goals:
  - Goal 1: Homogenous SRGB
  - Goal 2: Unique SID-to-prefix mappings
    - SRGB size > # required SIDs
    - Each SID can be allocated to a single prefix. No SID re-use among prefixes
- Large majority of deployments should be able to meet these goals
  - in some cases, they cannot be achieved due to scaling and platform limitations

Configured SRGB and SRGB sub-ranges

- For ease of administration and operations, the configured SRGB is carved in administrative sub-ranges
  - Allocate a sub-range to each domain
  - The configured SRGB is still the entire SRGB, not the SRGB sub-range
- Alternatively, an operator could treat the configured SRGB as a global pool of SIDs
  - A global pool might lead to a more optimal use of the SRGB
  - More complicated administration

SRGB and SRGB sub-ranges

Notation convention:
SRGB {XXX-YYY}
sub-range [QQQ-RRR]



Note: [16k-17k] really means [16000-16999]

How many SIDs are needed?

- Number of SIDs ≠ number of nodes

- >1 SIDs needed per node
  - Algo(0) Prefix-SID
  - With Flex-Algo, multiple SIDs are mapped to a prefix
  - How many Flex-Algo SIDs?
    - Delay metric: 1 Flex-Algo for low-delay service
    - TE metric: 1 Flex-Algo for e.g. premium service
    - IGP metric: 2 Flex-Algos for dual-plane
    - FA with affinity constraints? E.g., use encrypted links, avoid low BW links

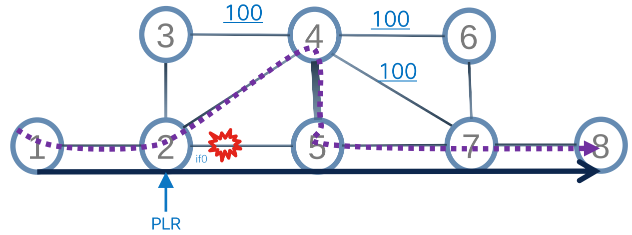
How many Flex-Algos?

- A high-bar estimate that 8 Flex-Algos are required
  - IGP metric: 2 Flex-Algos for dual-plane
    - Maybe more for additional slices?
    - More using affinities? E.g. use encrypted links, avoid low BW links
  - TE metric: 1 Flex-Algo for e.g. premium service
  - Delay metric: 1 Flex-Algo for low-delay service
  - Multiply by 2 for future expansion
    → (2 + 1 + 1) * 2 = 8 Flex-Algos
- This fits the rule-of-thumb to limit # Flex-Algos on a node to single-digit number

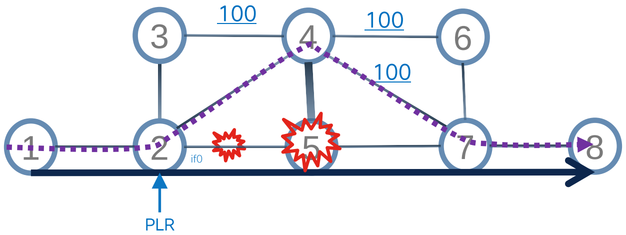
TI-LFA

Protect with automatic TI LFA FRR

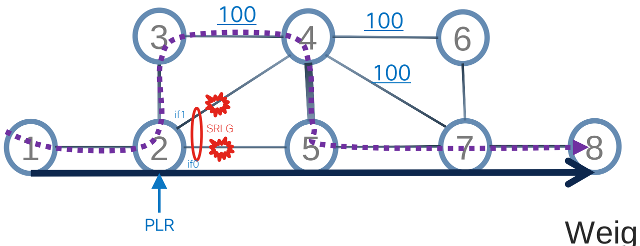
Link protection



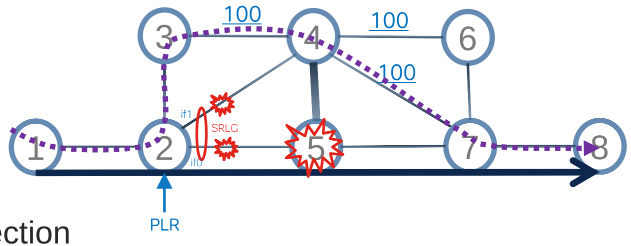
Node protection



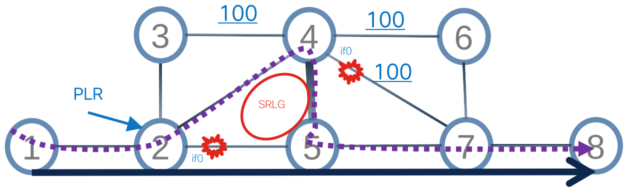
Local SRLG protection



Local SRLG + Node protection



Weighted Remote SRLG protection



TI-LFA – Backup based on Prefix-priority

- Usecase
- Tight control of ASIC resources consumed for fast-reroute

TI-LFA – Backup based on Prefix-priority

```
RP/0/RP0/CPU0:R1(config)#router isis 100
RP/0/RP0/CPU0:R1(config-isis)#address-family ipv6 unicast
RP/0/RP0/CPU0:R1(config-isis-af)#fast-reroute per-prefix ?
  <...>
  priority-limit      Limit backup computation upto the prefix priority
  <...>

RP/0/RP0/CPU0:R1(config-isis-af)#fast-reroute per-prefix priority-limit ?
  critical  Compute for critical priority prefixes only
  high      Compute for critical & high priority prefixes
  medium    Compute for critical, high & medium priority prefixes

RP/0/RP0/CPU0:R1(config-isis-af)#fast-reroute per-prefix priority-limit high ?
  level  Set priority-limit for one level only
  <cr>
```

IOS-XR 7.11.1

SR uLoop avoidance per-prefix filtering

- Usecase:
  - Tight control of ASIC resources
- RPL used for uLoop per-prefix filtering supports:
  - Destination based match
  - Tag based match

```
router isis core
 address-family ipv4 unicast
  microloop avoidance segment-routing route-policy FOO-rpl
 !
 address-family ipv6 unicast
  microloop avoidance segment-routing route-policy BAR-rpl
```

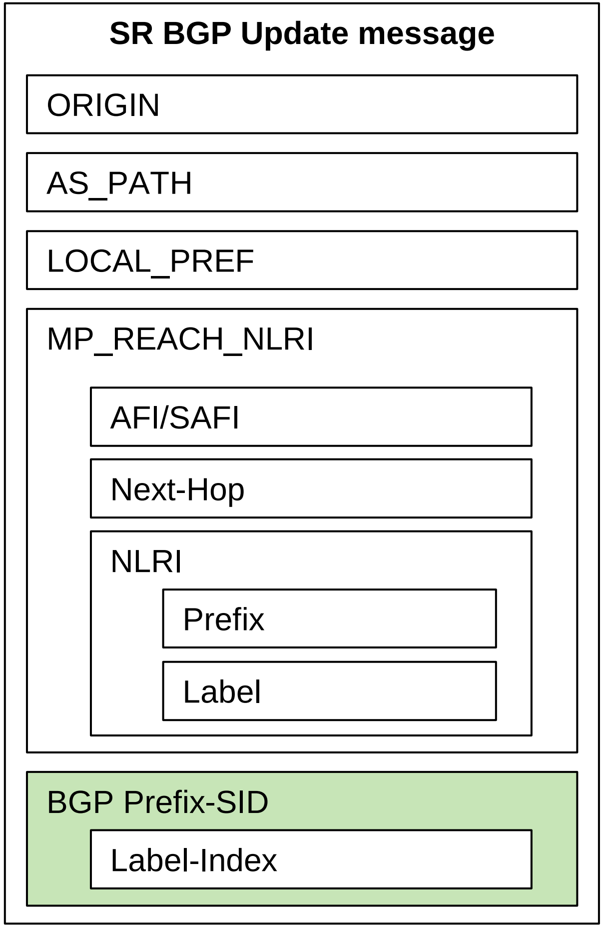

BGP-SR / BGP Prefix-SID

BGP Prefix-SID

- A BGP Prefix-SID is advertised with a prefix in BGP Labeled Unicast (BGP-LU)
  - BGP-LU = IPv4/IPv6 Labeled Unicast Address-families
- BGP Prefix-SIDs are global SIDs
- The instruction of the BGP Prefix-SID is to forward the packet over the ECMP-aware BGP best-path to the associated prefix

BGP Prefix-SID advertisement

- Since the BGP Prefix-SID is a global SID, it is advertised as an index into the SRGB
  - BGP Prefix-SID label value = $SRGB_{base}$ + SID index
- SR BGP uses the BGP-LU address-family
- The Prefix-SID index is advertised in a Label-Index TLV of the BGP Prefix-SID attribute added to the BGP-LU Update message
  - BGP Prefix-SID attribute and Label-Index TLV are specified in draft-ietf-idr-bgp-prefix-sid



SR BGP configuration

- SR BGP is automatically enabled when configuring a global SRGB

```
segment-routing
 global-block 16000 23999
```

  - BGP uses this globally configured SRGB
  - Note 1: there is no default global SRGB
  - Note 2: if a global SRGB is configured, the IGPs use it by default

BGP Prefix-SID config – set label-index

- The Prefix-SID of a locally originated BGP route is set via a route-policy
- A route-policy with **`set label-index <idx>`** can be attached to:
  - a) **`network`** configuration
  - b) **`redistribute`** configuration

a)

```
route-policy SID($SID)
  set label-index $SID
end-policy
!
router bgp 1
 address-family ipv4 unicast
  network 1.1.1.1/32 route-policy SID(1)
  allocate-label all
```

b)

```
route-policy SIDs
  if destination in (1.1.1.1/32) then
    set label-index 1
  endif
end-policy
!
router bgp 1
 address-family ipv4 unicast
  redistribute connected route-policy SIDs
  allocate-label all
```

Increasing Resiliency with BGP-SR

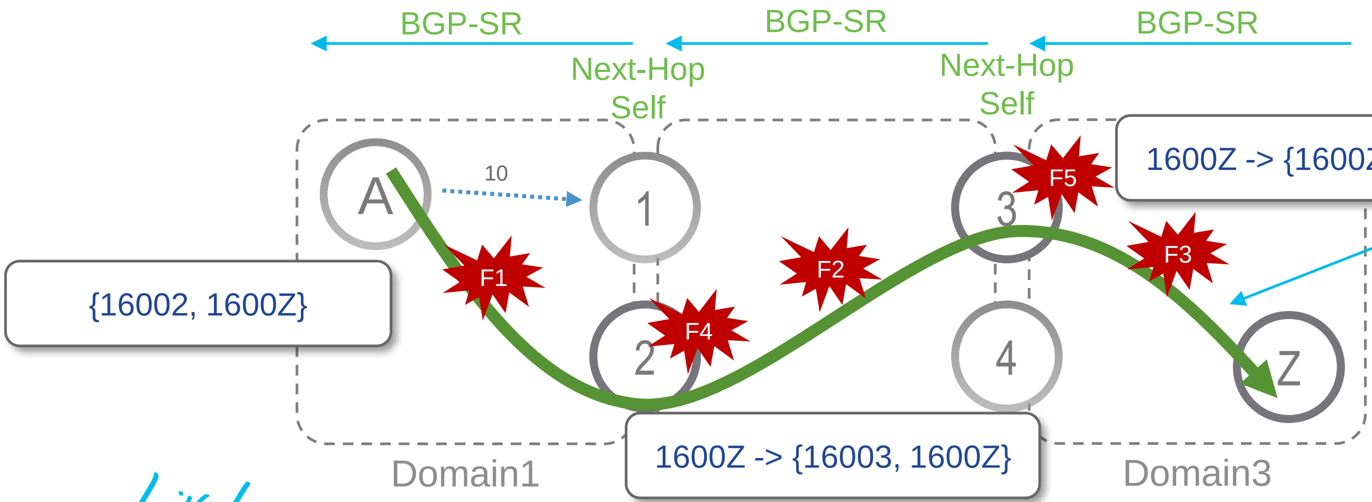
- Better resiliency with BGP-SR !!!
- Thanks for SR Prefix SID global labels, BGP-LU local labels are the same across ASBRs
- As a result, ASBR (inline BGP-LU RRs) can use anycast loopback as NH for BGP-LU prefixes

Without BGP-SR – Operation

- Failures F1, F2, F3
  - TI-LFA FRR
- Failures F4, F5
  - IGP + BGP PIC convergence

Fast convergence

Slower convergence



Optimum path using Node SIDs

Link IGP metric shown

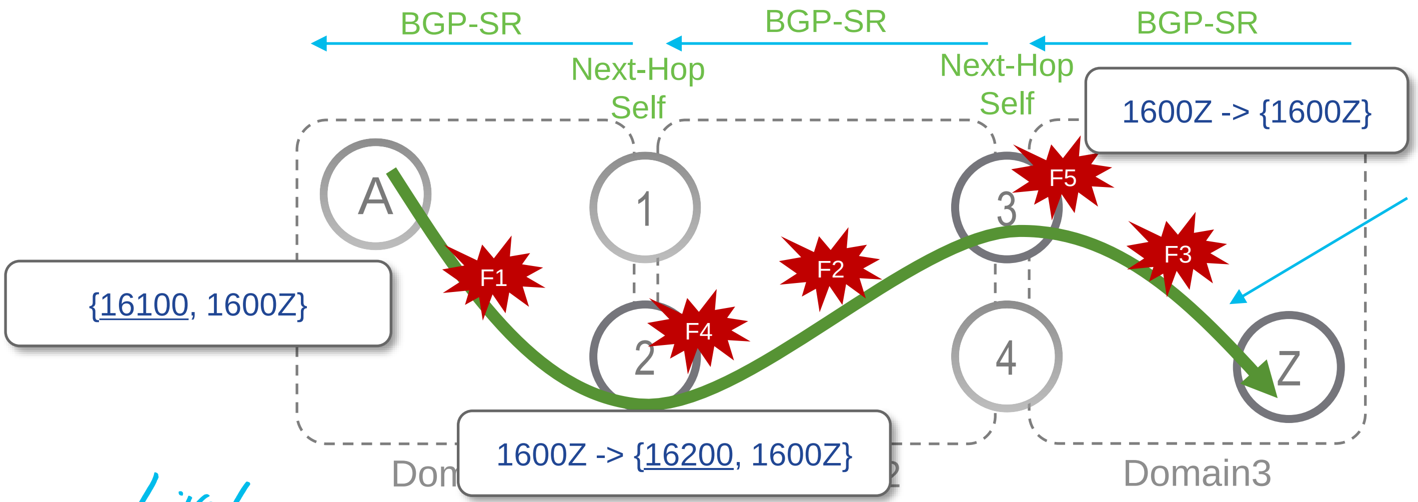
BGP-SR with Anycast NH

- Failures F1, F2, F3
  - TI-LFA FRR
- Failures F4, F5
  - TI-LFA FRR

Fast convergence

Fast convergence

16100 = Anycast SID in R1 and R2
16200 = Anycast SID in R3 and R4



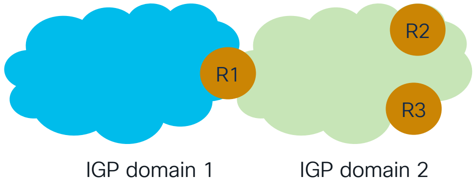
Optimum and Resilient path using weighted Anycast SIDs

Link IGP metric shown

Using Anycast – Operational Considerations

- How do you prevent sub-optimal routing when using Anycast?

ISIS Conditional Prefix Advertisement



- ISIS conditionally advertises a prefix based on RIB reachability to prefixes in a prefix-set
- Example:
  - Router 1 tracks reachability to loopback of R2 and R3
  - If both prefixes become unreachable, R1 stops advertising its loopback1000 (anycast) in Domain1

Router 1:

```
prefix-set domain_2_pfx
 1.1.1.2/32
 1.1.1.3/32
end-set
!
route-policy track_dom_2
 if rib-has-route async domain_2_pfx then
   pass
 endif
end-policy
!
router isis domain_1
 interface Loopback1000
  prefix-attributes anycast
  address-family ipv4 unicast
   advertise prefix route-policy track_dom_2
   prefix-sid absolute 16100 n-flag-clear
```

Object Tracking – BGP Neighbor

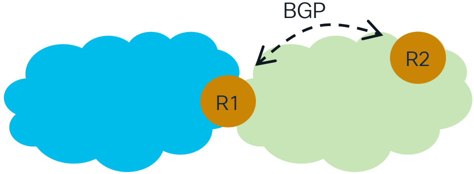
- Requirement: Track BGP neighbor address-family state as an Object in IOS-XR Object Track
- Object == UP
  - BGP Session is in ESTABLISHED state AND EOR is received from Neighbor AND Neighbor routes are installed in FIB
- Object == DOWN
  - BGP Session is NOT in ESTABLISHED state OR Neighbor address is not reachable in FIB

ISIS Conditional Prefix Advertisement

Object Tracking

IOS-XR 24.1.1

- ISIS conditionally advertises a prefix based on Object state
- Example:
  - Router 1 tracks BGP session to neighbor R2
  - If BGP Session is in ESTABLISHED state & EOR is received from Neighbor && Neighbor routes are installed in FIB, R1 advertises its loopback1000 (anycast) in Domain1



Router 1:

```
track bgp_neigh_obj
 type bgp neighbor address-family state
  address-family ipv4 unicast
   neighbor 1.1.1.2
!
route-policy obj-track-rpl
  if track bgp_neigh_obj is up then
    pass
  endif
end-policy
!
router isis domain_1
 interface Loopback1000
  prefix-attributes anycast
  address-family ipv4 unicast
   advertise prefix route-policy obj-track-rpl
   prefix-sid absolute 16100 n-flag-clear
```

Color-Aware Routing Principles

Types of intent

- Topology path selection
  - Minimize expected delay
  - Minimize delay
  - Minimize cost per bit with a delay bound
  - Avoid resource
  - Disjoint paths
  - Disjoint planes
  - Data Sovereignty
- Others
  - Steer traffic along a service chain
- Any combination of the above

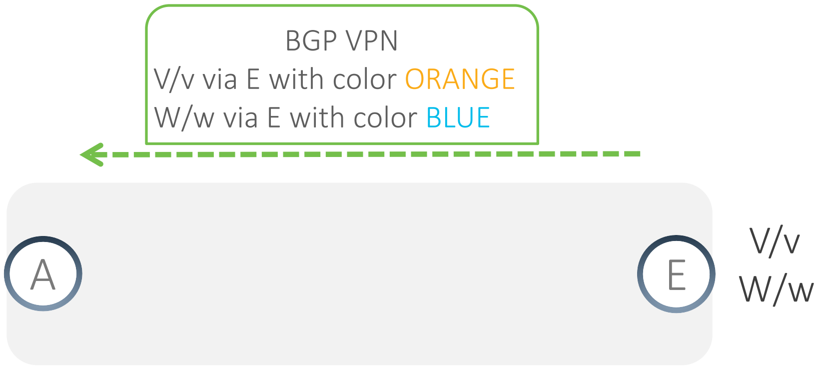
Intent encoded as a color

- Color is a standard way to signal intent
  - A 32-bit number

- Mapping an intent to a color:
  - Low-latency: BLUE
  - Low-cost: ORANGE

- Colored Service Routes – requesting a particular intent
- Color-aware Transport Routes – satisfying a particular intent
- A colored service route is steered over a color-aware route of same intent

Colored service route

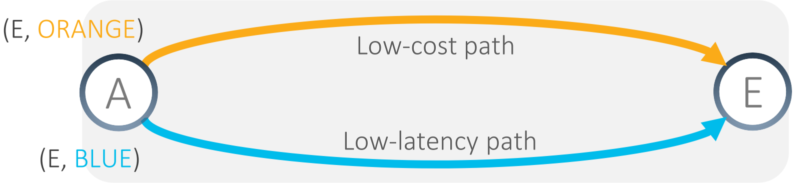


- E sends colored service routes, each requesting a particular intent
- Route coloring is done by using the BGP Color Extended-Community
  - Standard (RFC5512 / RFC9012), supported by all major BGP implementations
- Any service route can be colored (L3VPN, EVPN, Internet routes)

Color-aware transport routes

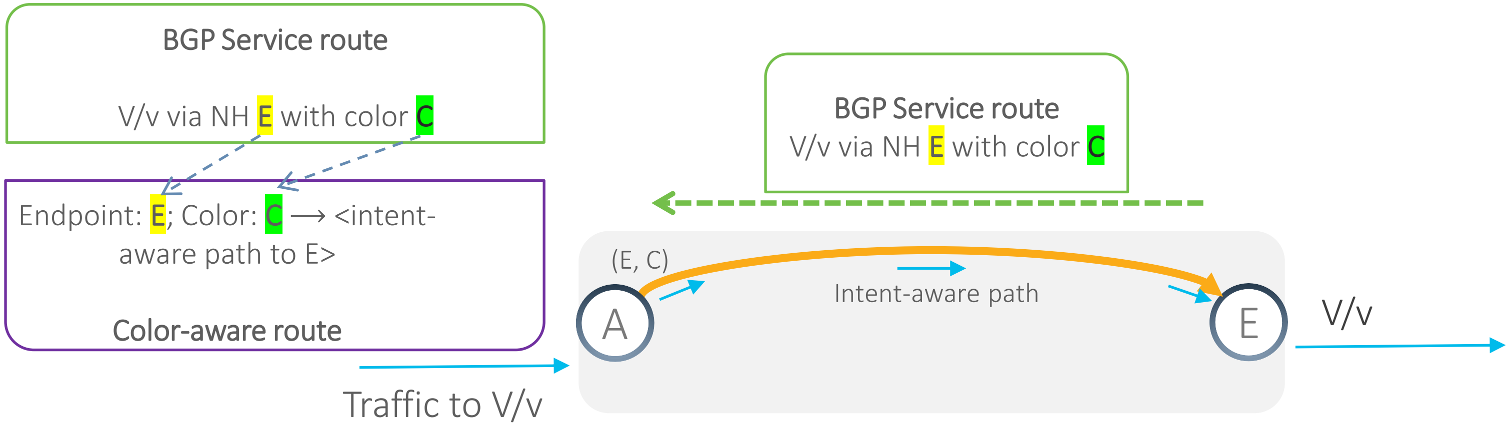
Color-aware routes @ A:

Endpoint	Color	Path
E	ORANGE	<low-cost path A to E>
E	BLUE	<low-latency path A to E>



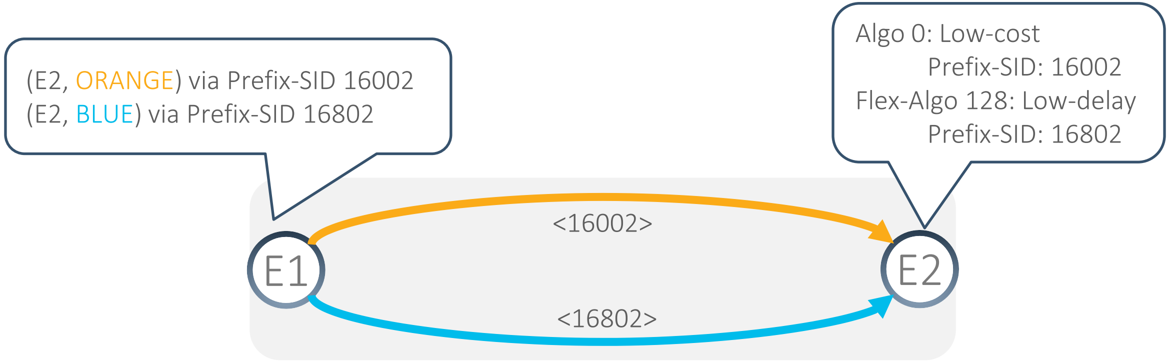
- A color-aware route satisfies a particular intent
- A color-aware route is identified by the tuple (Endpoint and Color); in short (E,C)
- A color-aware route can be signaled/instantiated by different mechanisms

Service routes steered on color-aware route



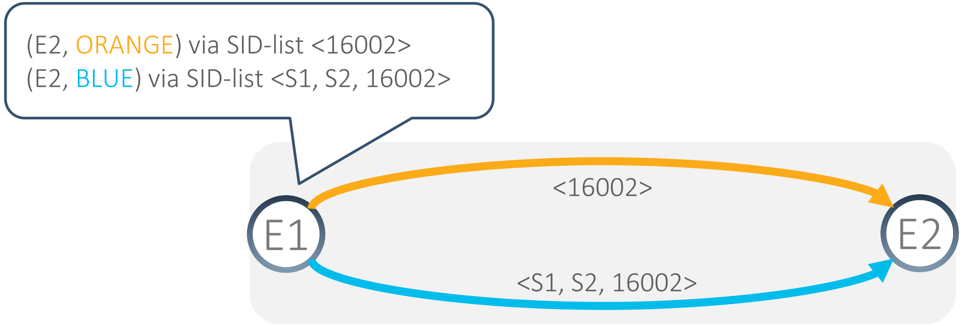
- A colored service route is steered over a color-aware route of same intent
  - Traffic destined to prefix V/v via **E** with color **C** is steered over color-aware route (E, C)
  - This is known as Automated Steering

Color-aware route (E2, C) provided by IGP Flex Algo



- IGP Flex Algo (RFC9350)
- E1 maps color to an IGP algorithm
  - Orange ⟶ Algo 0
  - Blue ⟶ Flex-Algo 128
- IOS-XR implementation available since 2019

Color-aware route (E2, C) provided by SR Policy



- SR Traffic Engineering Policy – in short SR Policy (RFC9256)
- E1 has two SR policies
  - (E2, ORANGE): Dynamic, low cost with SID-list <16002>
  - (E2, BLUE): Dynamic, low delay with SID-list <S1, S2, 16002>
- IOS-XR implementation available since 2017

SR Policy

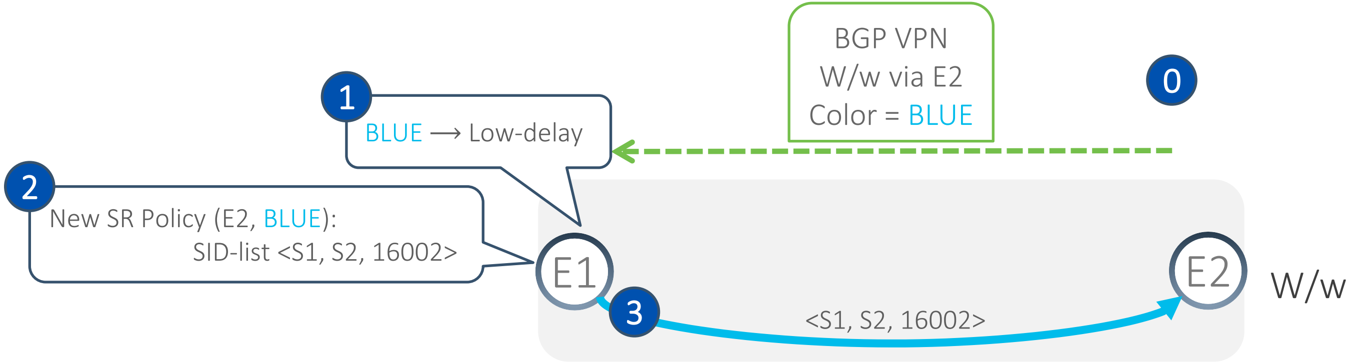
Key IETF document for SRTE

Internet Engineering Task Force (IETF)
Request for Comments: 9256
Updates: 8402
Category: Standards Track
ISSN: 2070-1721

C. Filsfils
K. Talaulikar, Ed.
Cisco Systems, Inc.
D. Voyer
Bell Canada
A. Bogdanov
British Telecom
P. Mattes
Microsoft
July 2022

Segment Routing Policy Architecture

SR Policy pull model: On-Demand Nexthop (ODN)



1. E1 maps color BLUE to the low-delay intent
2. Upon receiving a service route via E2 with color BLUE, E1 automatically instantiates the SR Policy (E2, BLUE)
   - This is called On-Demand Next-hop (ODN)
   - Each PE installs only the SR Policies that it needs
3. E1 steers the traffic for prefix W/w onto SR Policy (E2, BLUE)

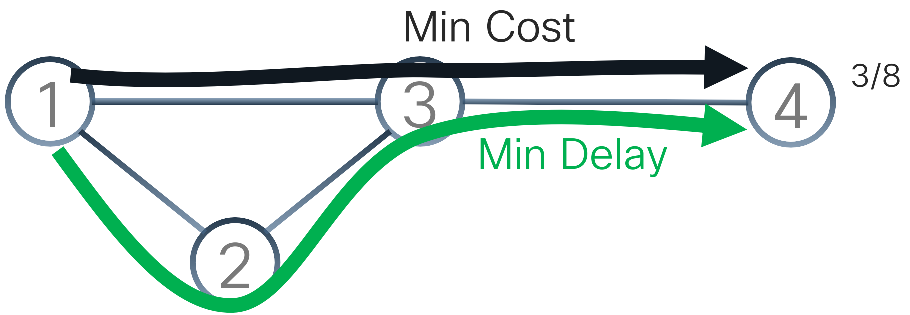
Need for Per-Flow Automated Steering

Same
Destination

F1: 3/8 (DSCP 0) → IGP to 4

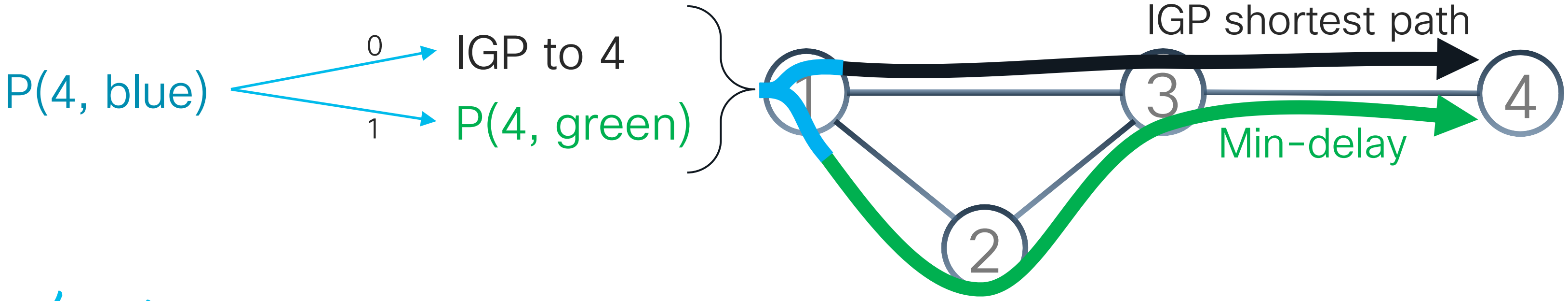
F2: 3/8 (DSCP 46) → P(green, 4)

Different
Flows (flow 1, flow 2)



Per-Flow SR Policy (PFP)

- Per-Flow Policy (Node4, blue) @ Node1
  - FC=0 ➔ IGP shortest path == 16004
  - FC=1 ➔ Per-Destination SR Policy (Node4, green)
- Per-Destination Policy (Node4, green)
  - Defined as Min Delay ➔ <16002, 16004>



SR Path Computation Element (SR-PCE)

SR Path Computation Element (SR-PCE)

SRTE Head-End

Distributed Mode – SR-TE Head-End
Visibility is limited to its own IGP domain

Solution

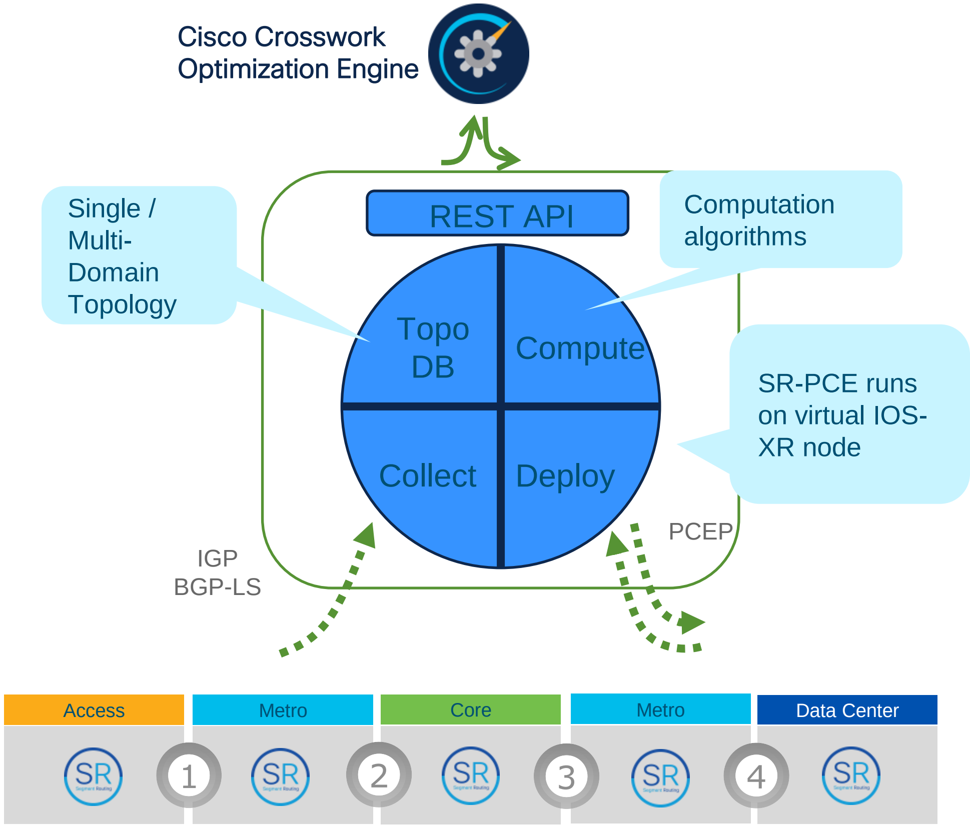
Multi-Domain SRTE Visibility
Centralized SR-PCE for Multi-Domain Topology view

Integration with Applications
North-bound APIs for topology/deployment

Delivers **across the unified SR Fabric** the SLA requested by the service

Benefits

Simplicity and Automation
End-to-End network topology awareness
SLA-aware path computation across network domains



SRTE Use Cases

Distributed or Centralized Path Computation?

Blue = computed by SR-PCE
Green = computed by head-end

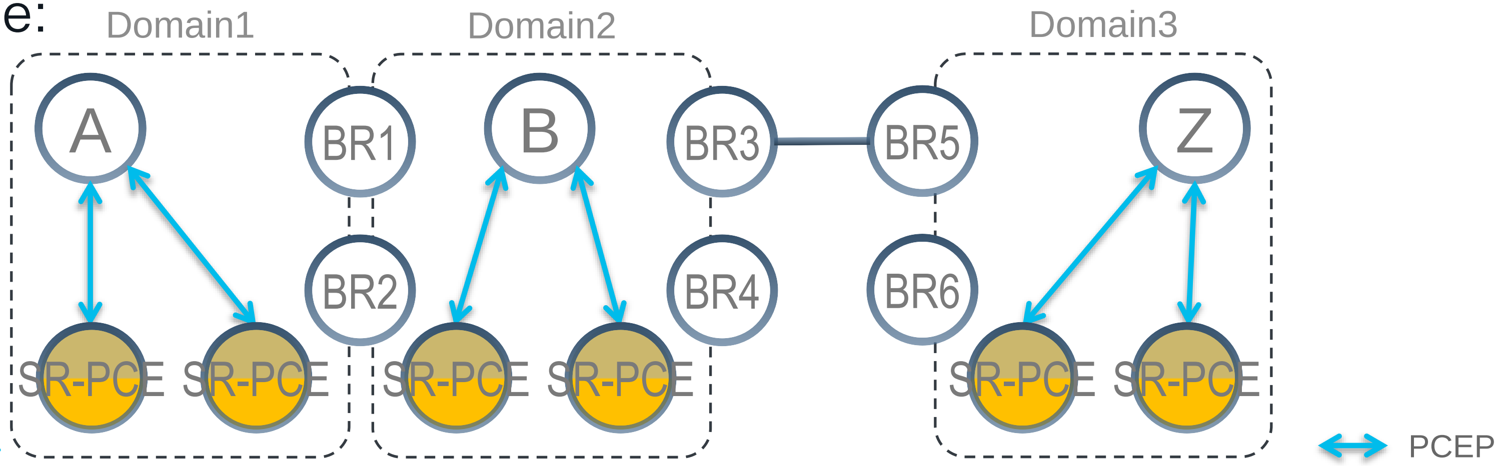
Use Case	Optimization objective / constraints	Single-Domain	Multi-Domain
Reachability	IGP metric + constraints	Distributed or Centralized	Centralized
Low Latency (TE metric)	TE metric + constraints	Distributed or Centralized	Centralized
Low Latency (actual)	Latency + constraints	Distributed or Centralized	Centralized
Path Disjointness	IGP / TE metric + PCEP association group	Centralized	Centralized
Tree-SID	P2MP	Centralized	Centralized

SR-PCE – Fundamentally Distributed

- SR-PCE not to be considered as a single all-overseeing device
- SR-PCE deployment is closer to BGP RR deployment model
- Different service end-points (PEs) can use different pairs of SR-PCEs
- Choice of SR-PCE can either be based on proximity or service-type

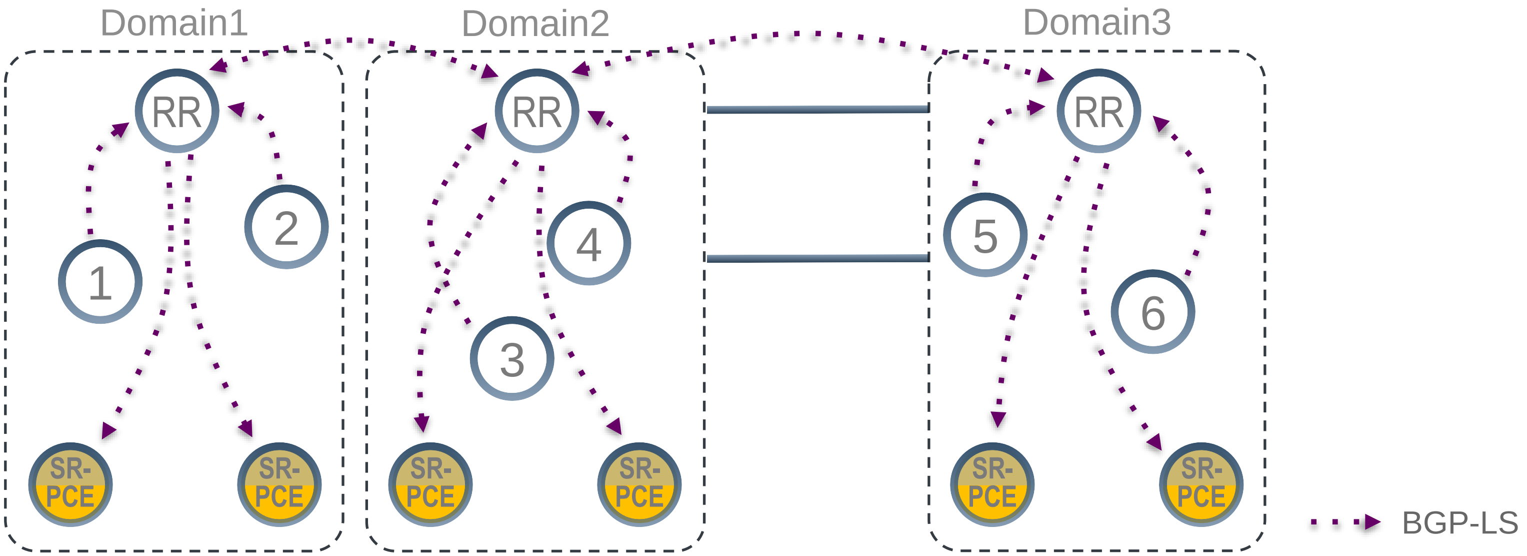
SR-PCE – Fundamentally Distributed

- Add SR-PCE nodes where needed; per geographic region, per service, ...
  - SR-PCE needs to get the required topology information for its task
    - E.g. to compute inter-domain paths SR-PCE needs the topology of all domains
- Example:



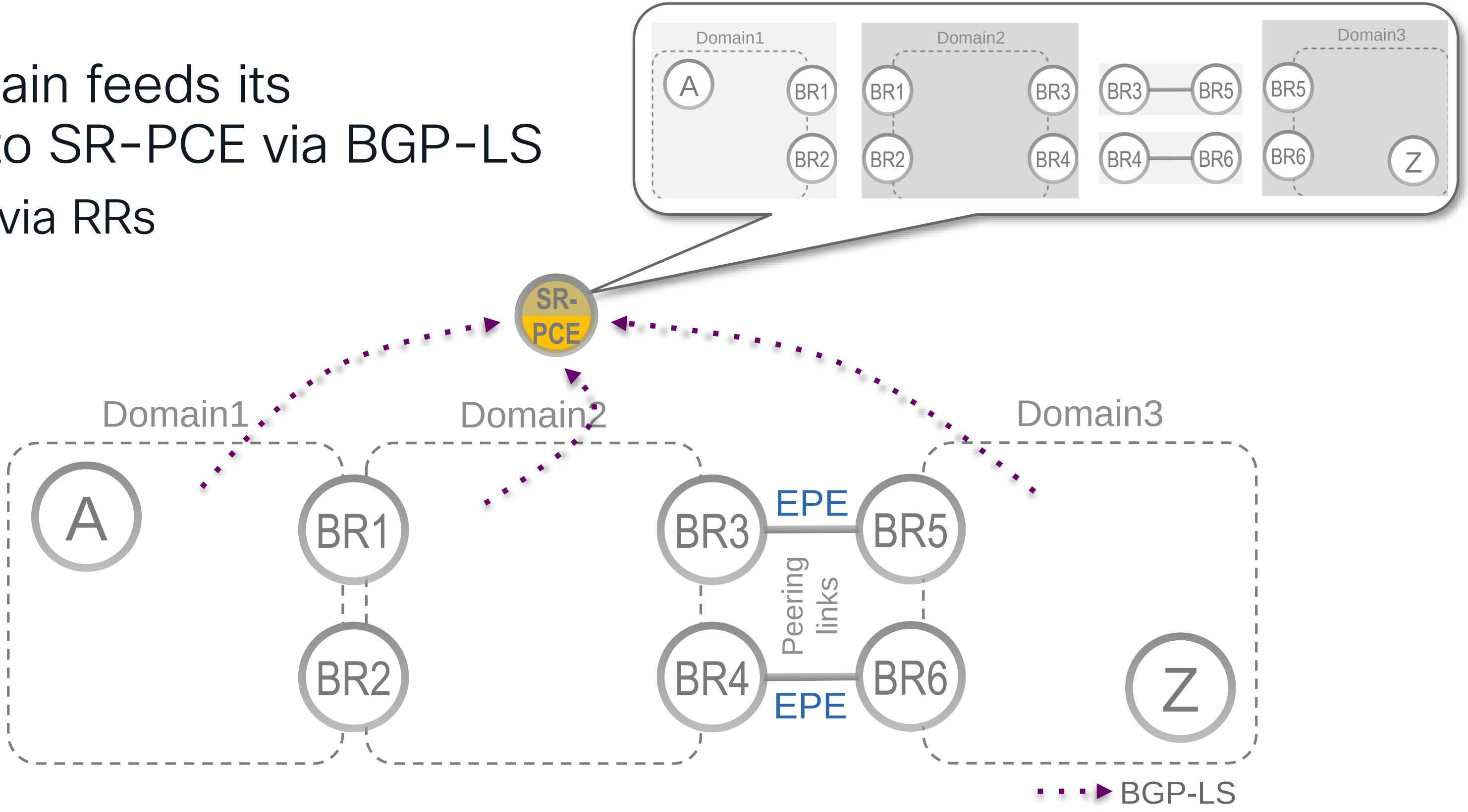
SR-PCE – Fundamentally Distributed

- Using RRs to scale the BGP-LS topology distribution
- Any node can have a BGP-LS session to the RR



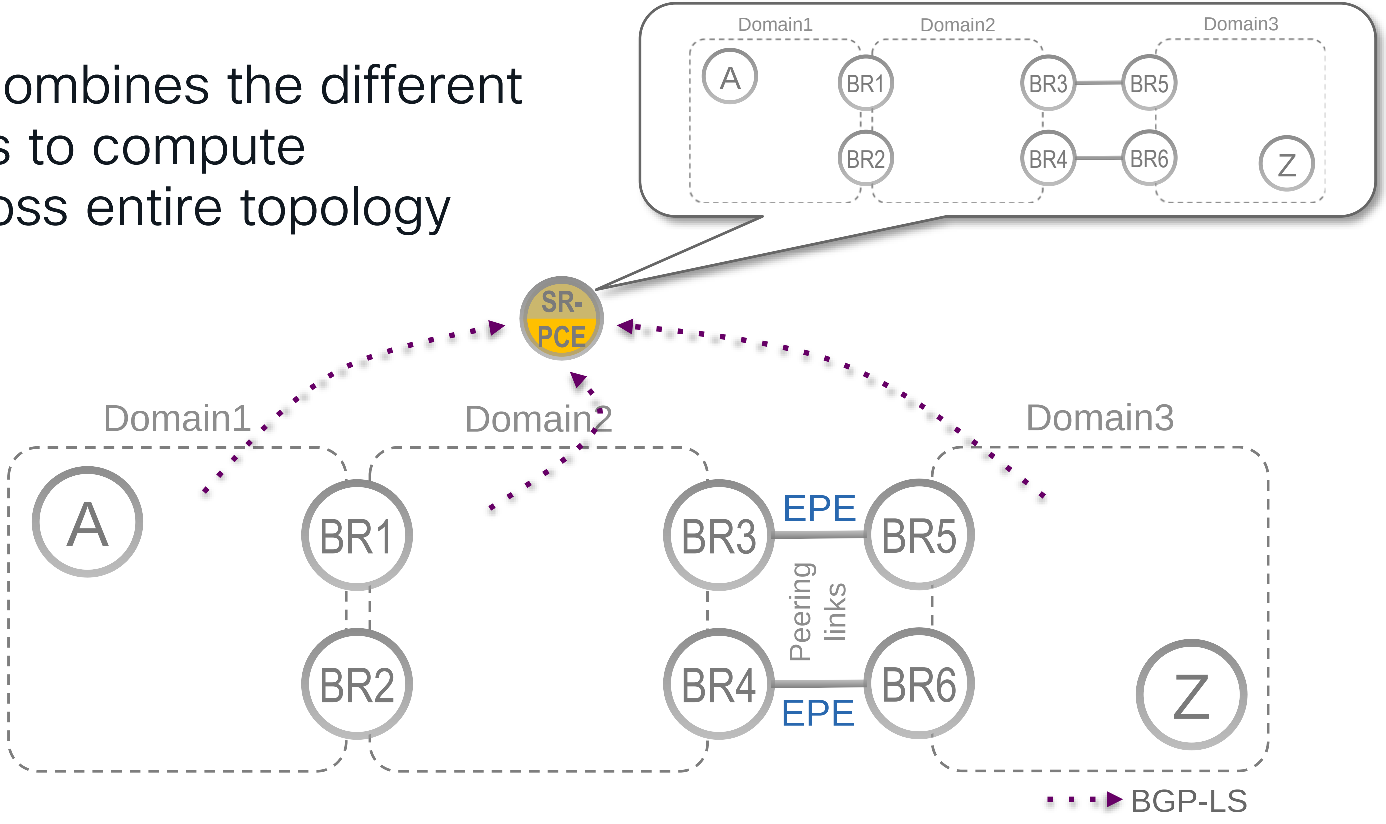
SR-PCE receives topology of all domains

- Each domain feeds its topology to SR-PCE via BGP-LS
  - Typically via RRs



SR-PCE consolidates the topologies

- SR-PCE combines the different topologies to compute paths across entire topology



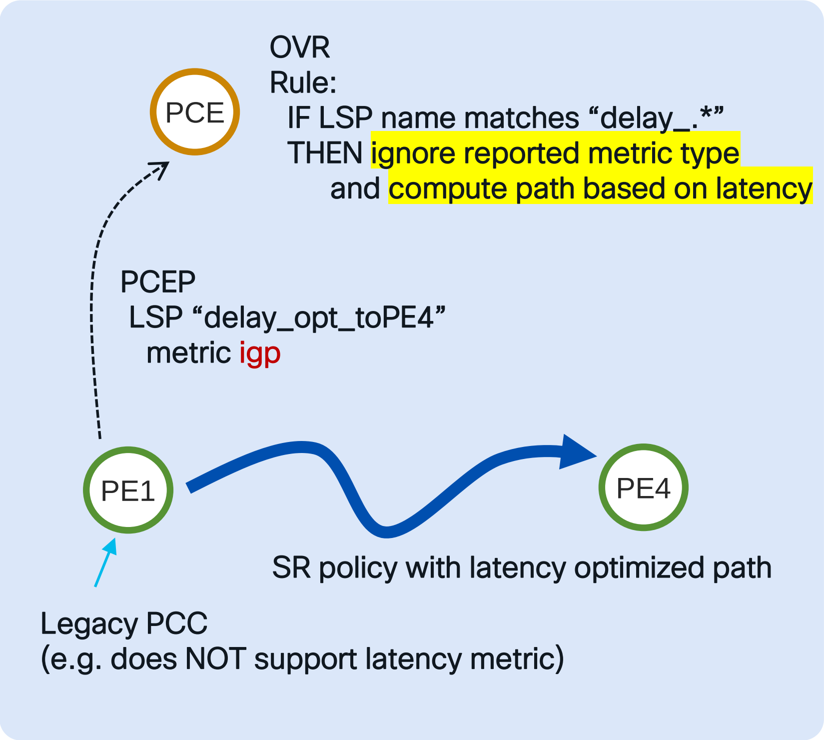
SR-PCE – High Availability (HA)

- SR-PCE leverages the well-known standardized PCE HA
- Head-end sends PCEP Report for its SR Policies to all connected SR-PCE nodes
- Head-end delegates control to its primary SR-PCE
  - Delegate flag (D) is set in PCRept to primary SR-PCE
- Upon failure of the primary SR-PCE, head-end re-delegates control to another SR-PCE

IOS-XR 7.7.1

PCE: Override Rules

- Highlights:
  - New Override Rules configured at PCE that allow PCE to perform path computation based on modified attributes from those signaled by the PCC
- Use Case / Value Proposition:
  - Allow operators to deploy advanced functionality based on PCE capabilities without having to invest / upgrade legacy PCCs



PCE: Override Rules

Cisco IOS-XR Implementation Highlights – Configuration

```
pce
 override-rules
  sequence <sequence-number>
   matching-criteria
    peer
     all
     access-list ipv4 <ipv4-acl-name>
    lsp
     all
     name <lsp-name in form of regex>
     colors <colors and color ranges> ! “0-50,55,70-80”
   override
    metric
     type {igp | te | latency | hopcount}
    constraints
     segments {protection | sid-algorithm}
     bandwidth <1-4294967295>
```

PCE: Override Rules

Cisco IOS-XR Implementation Highlights – Configuration Examples

```
pce
 override-rules
  sequence 10
   matching-criteria
    peer
     all
    lsp
     name “delay_.*”
   override
    metric
     type latency
  sequence 20
   matching-criteria
    peer
     all
    lsp
     name “flex_128_.*”
   override
    constraints
     segments
      sid-algorithm 128
```

- Rule matches LSPs from any PCEP peer and symbolic name matching regex “delay_.*”
- Latency is used as modified optimization objective during path computation

- Rule matches LSPs from any PCEP peer and symbolic name matching regex “flex_128_.*”
- Flex-Algo 128 is used as a constraint during path computation

SRTE with SR IGP Flexible Algorithm

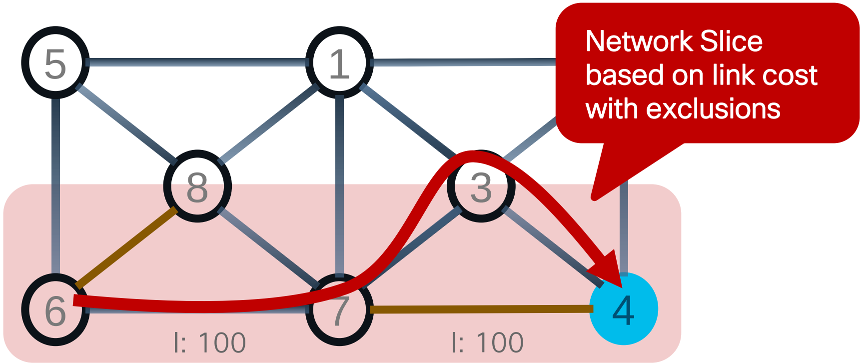
SR IGP Flexible Algorithm (FA)

- Complements the SRTE solution with customizable Prefix-SIDs

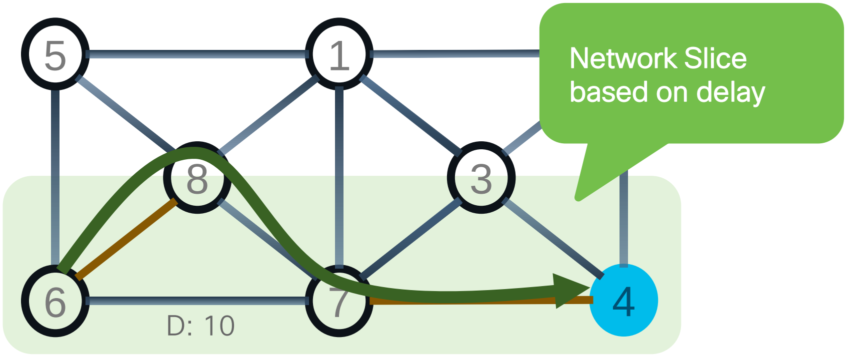
- We call “Flex-Algo”
  - The algorithm is defined by the operator, on a per-deployment basis
- Flex-Algo K is defined as
  - The minimization of a specified metric: IGP, TE or delay
  - The exclusion of certain link properties: link-affinity, SRLG, interface speed, link delay

SR IGP Flexible Algorithms

- Examples
  - Operator defines Flex-Algo 128 as “minimize IGP metric while avoiding links with link-affinity brown”
  - Operator defines Flex-Algo 129 as “minimize delay metric”



Default IGP
link metric: I:10



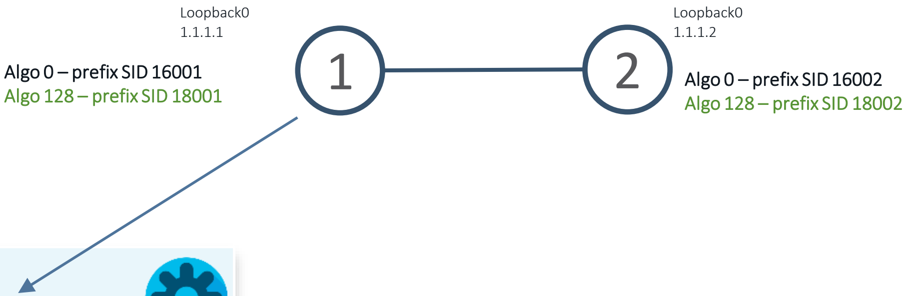
Measured link
Delay: D:1

Flex-Algo and SR data-planes

- Flex-Algo is applicable to SR-MPLS and SRv6

- SR-MPLS
  - A node SID (prefix label) is assigned to the loopback interface
- SRv6
  - A flex-algo locator is assigned to the node

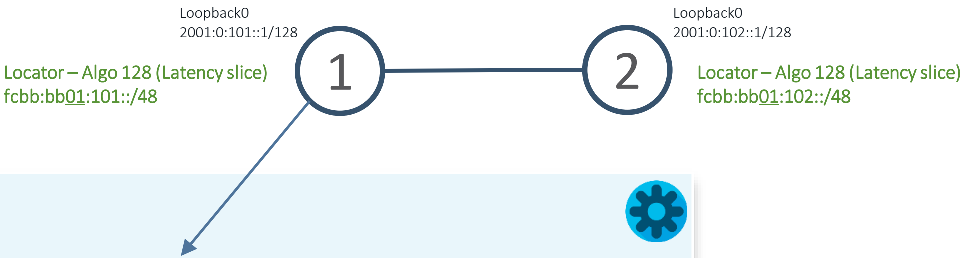
SR-MPLS Flex-Algo



Router 1:

```
router isis core
 flex-algo 128
  metric-type delay
  advertise-definition
 !
 address-family ipv4 unicast
  segment-routing mpls
 !
 interface Loopback0
  passive
  address-family ipv4 unicast
   prefix-sid absolute 16001
   prefix-sid algorithm 128 absolute 18001
```

SRv6 Flex-Algo

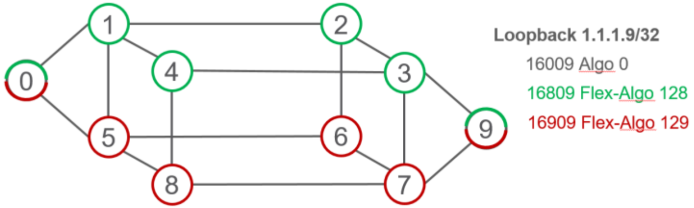



```
segment-routing
 srv6
  locators
   locator LOC_LATENCY
    micro-segment behavior unode psp-usd
    prefix fcbb:bb01:101::/48
    algorithm 128
```

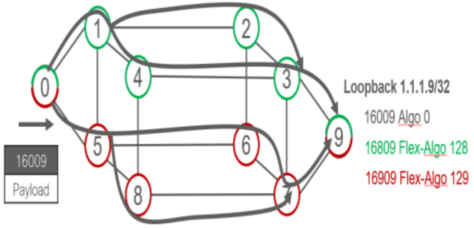
```
router isis core
 flex-algo 128
  metric-type delay
  advertise-definition
 !
 address-family ipv6 unicast
  segment-routing srv6
   locator LOC_LATENCY
```

Multi-Plane Networks

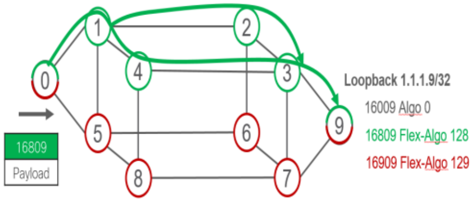
Powered by SR IGP Flex Algo



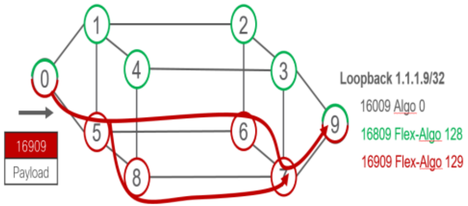
- All the nodes support Algo 0: minimize IGP metric
- Green nodes also support 128: minimize IGP metric
- Red nodes also support 129: minimize Delay



- Path to Node 9 across Algo 0



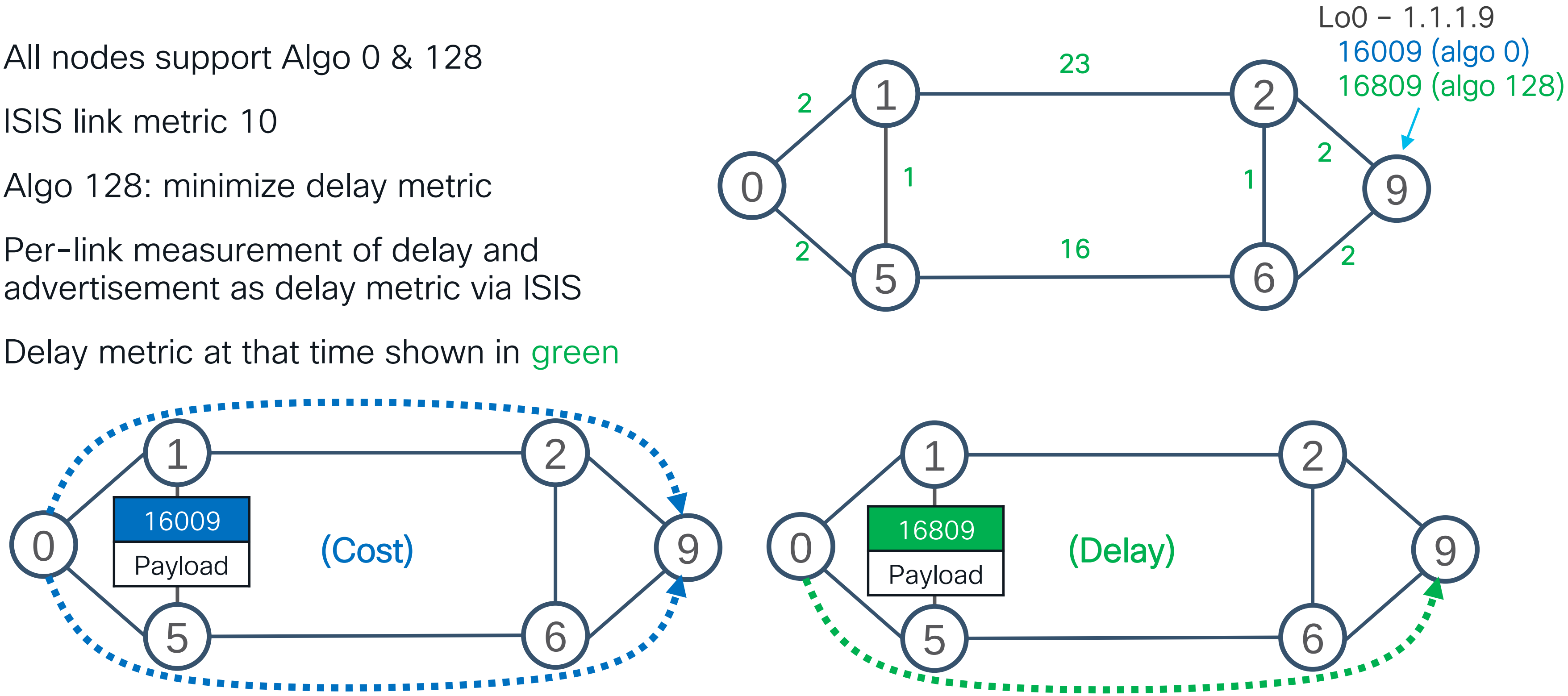
- Path to Node 9 across Flex-Algo 128



- Path to Node 9 across Flex-Algo 129

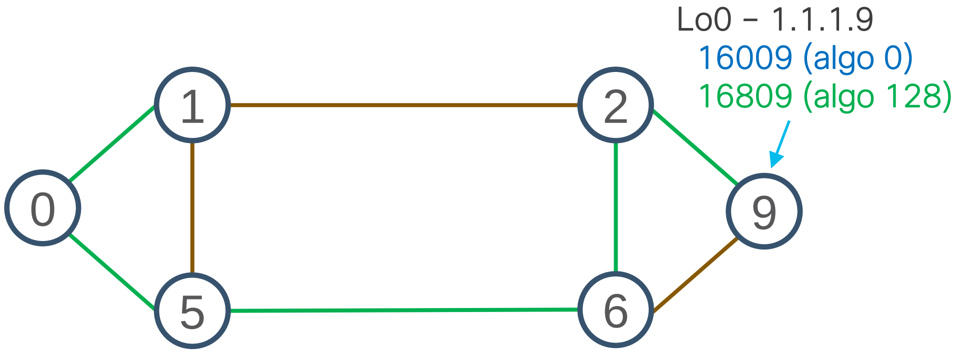
Use-Case – Delay vs Cost of Transport

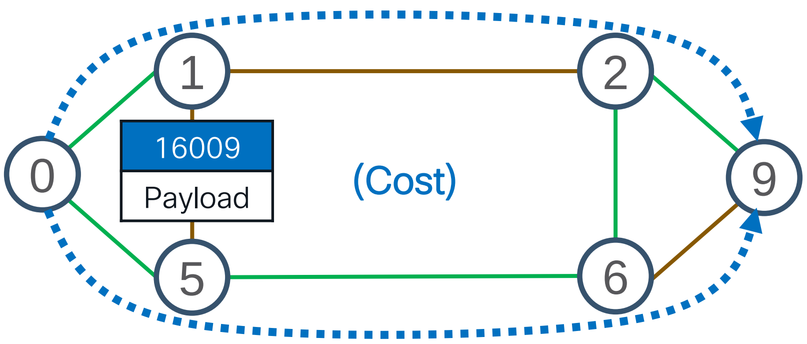
- All nodes support Algo 0 & 128
- ISIS link metric 10
- Algo 128: minimize delay metric
- Per-link measurement of delay and advertisement as delay metric via ISIS
- Delay metric at that time shown in green

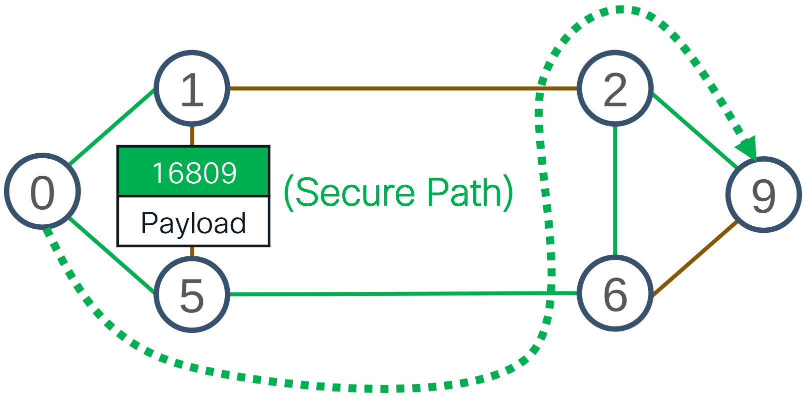


Use-Case – SRTE for Intelligent Secure Paths

- 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



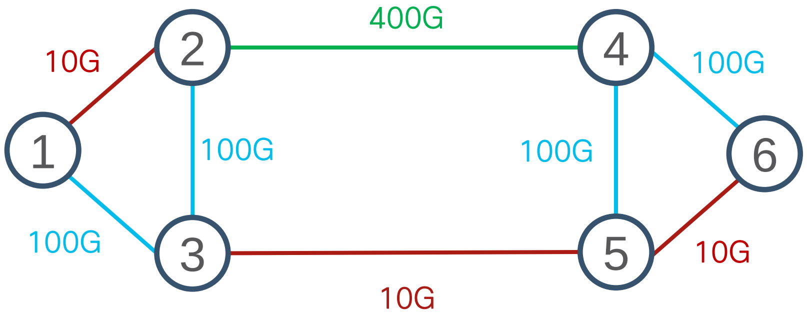




IOS-XR 7.11.1

Flex-Algo - Minimum-BW Constraint

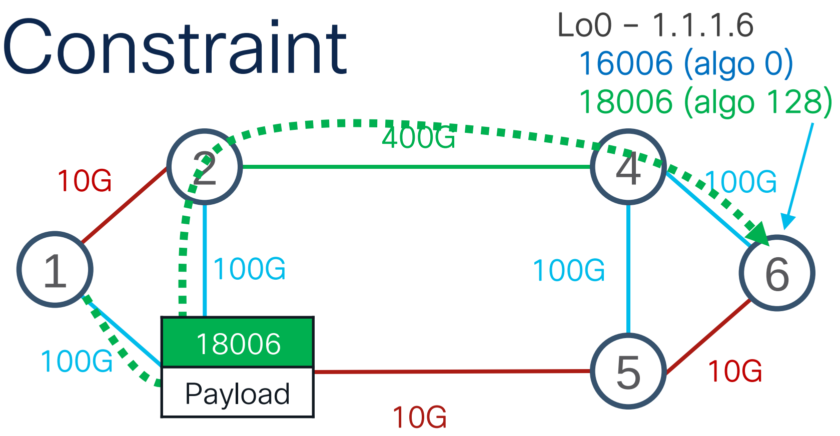
- Usecase
  - Flex-Algo instance consisting of links with a bandwidth exceeding a user-configured minimum value
- Defined in IETF draft-ietf-lsr-flex-algo-bw-con
- Applicable to SR-MPLS and SRv6



Flex-Algo - Minimum-BW Constraint

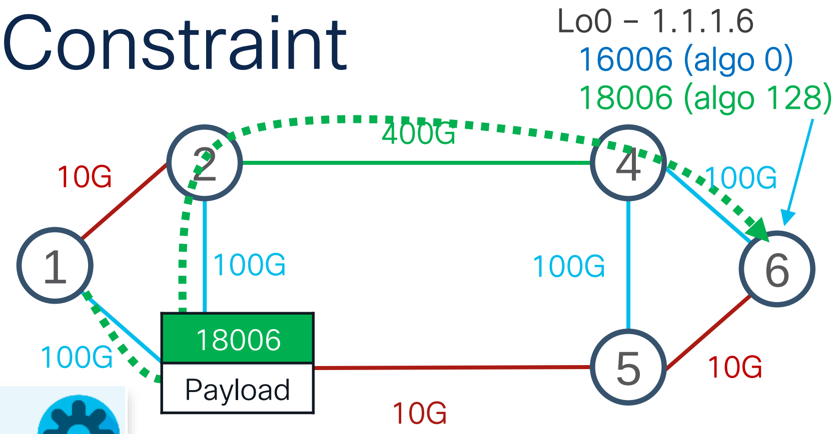
IOS-XR 7.11.1

- Usecase
  - Flex-Algo instance consisting of links with a bandwidth exceeding a user-configured minimum value
- Defined in IETF draft-ietf-lsr-flex-algo-bw-con
- Applicable to SR-MPLS and SRv6



IOS-XR 7.11.1

Flex-Algo - Minimum-BW Constraint



Router 6:

```
router isis core
 flex-algo 128
  advertise-definition
  minimum-bandwidth 100000000
 !
 interface Loopback0
  passive
  address-family ipv4 unicast
   prefix-sid algorithm 128 absolute 18006
```

Metric = IGP (default)

Minimum link bandwidth (kbits/sec); e.g. 100 Gbps

IOS-XR 7.11.1

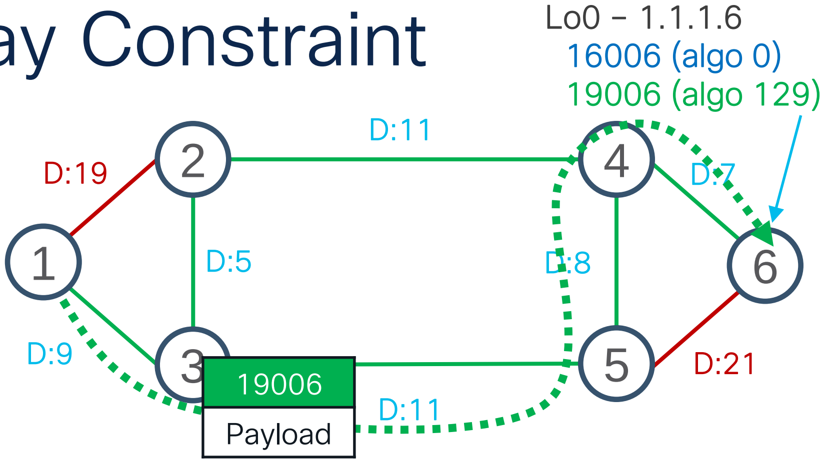
Flex-Algo - Maximum-Delay Constraint

- Usecase
  - Flex-Algo instance consisting of links with a manual / measured propagation delay less than a user-configured maximum value
- Defined in IETF draft-ietf-lsr-flex-algo-bw-con
- Applicable to SR-MPLS and SRv6

Flex-Algo - Maximum-Delay Constraint

IOS-XR 7.11.1

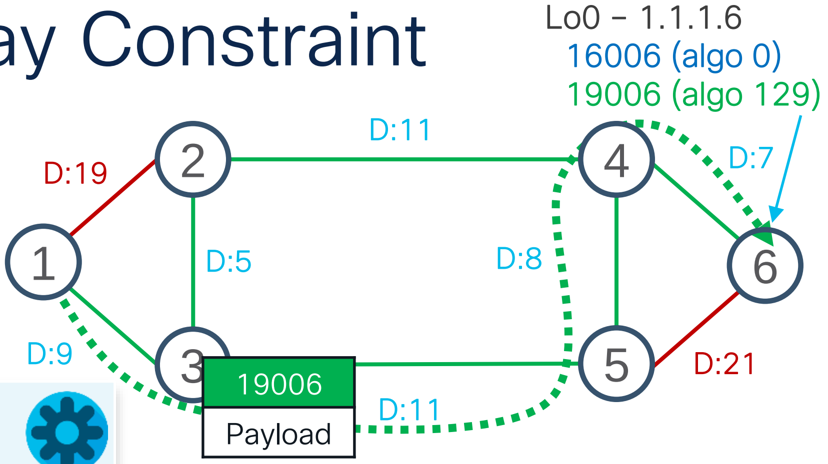
- Usecase
  - Flex-Algo instance consisting of links with a manual / measured propagation delay less than a user-configured maximum value
- Defined in IETF draft-ietf-lsr-flex-algo-bw-con
- Applicable to SR-MPLS and SRv6



IOS-XR 7.11.1

Flex-Algo - Maximum-Delay Constraint

- Note that this is different than FA with delay metric. Instead of minimizing e-2-e delay, it prunes links with a high-delay



Router 6:

```
router isis core
 flex-algo 129
  advertise-definition
  maximum-delay 12
 !
 interface Loopback0
  passive
  address-family ipv4 unicast
   prefix-sid algorithm 129 absolute 19006
```

Metric = IGP (default)

Maximum link delay (usec); e.g. 12 usecs

Flex-Algo Details

Advertisements of Link Attributes for FA

- Link attributes that are to be used during Flex-Algorithm calculation MUST use the Application-Specific Link Attribute (ASLA) advertisements defined in [RFC8919] or [RFC8920]
- The mandatory use of ASLA advertisements applies to link attributes; including:
  - Min Unidirectional Link Delay
  - TE Default Metric
  - Administrative Group / Extended Administrative Group
  - Shared Risk Link Group
  - And any other link attributes that may be used in the future

Flex-Algo – Metric-type

```
RP/0/RP0/CPU0:R1(config-isis)# flex-algo 140
RP/0/RP0/CPU0:R1(config-isis-flex-algo)#?
  <...>
  metric-type           Metric-type used by flex-algo calculation
  <...>

RP/0/RP0/CPU0:R1(config-isis-flex-algo)#metric-type ?
  delay  Use delay as metric
  te     Use Traffic Engineering metric
```

IGP == default metric-type

Flex-Algo – TE metric link attribute

```
RP/0/RP0/CPU0:R1(config)#router isis 100
RP/0/RP0/CPU0:R1(config-isis)#interface tenGigE 0/0/0/0
RP/0/RP0/CPU0:R1(config-isis-if)#address-family ipv4 unicast
RP/0/RP0/CPU0:R1(config-isis-if-af)#?
  <...>
  te-metric    Configure an application specific TE metric for the interface
  <...>

RP/0/RP0/CPU0:R1(config-isis-if-af)#te-metric ?
  flex-algo  Configure a Flex-algo TE metric for the interface

RP/0/RP0/CPU0:R1(config-isis-if-af)#te-metric flex-algo ?
  <1-16777214>  Flex-algo traffic-engineering metric
```

Flex-Algo – Example: metric TE

```
router isis 100
 flex-algo 140
  metric-type te
  advertise-definition
 !
 interface TenGigE0/0/0/0
  address-family ipv4 unicast
   te-metric flex-algo 100
 !
 interface TenGigE0/0/0/0
  address-family ipv4 unicast
   te-metric flex-algo 50
```

Flex-Algo – Example: metric Delay

```
router isis 100
 flex-algo 140
  metric-type delay
  advertise-definition
 !
 interface TenGigE0/0/0/0
  address-family ipv4 unicast
 !
 interface TenGigE0/0/0/1
  address-family ipv4 unicast
 !
!
```

```
performance-measurement
 interface TenGigE0/0/0/0
  delay-measurement
   advertise-delay 12
  !
 !
 interface TenGigE0/0/0/1
  delay-measurement
  !
 !
!
```

Manually configured min unidirectional link delay

Measured min unidirectional link delay

IOS-XR 7.8.1

ISIS max-metric options for TE / Delay

```
RP/0/RP0/CPU0:R1(config-isis)#?
  <...>
  max-metric    Signal other routers to use us as transit option of last resort
  <...>

RP/0/RP0/CPU0:R1(config-isis)#max-metric ?
  <...>
  delay          Apply max-metric to delay metric
  te             Apply max-metric to TE metric
  <...>
```

Max-metric == 16,777,214

Flex-Algo – Example: metric IGP and Affinity constraints

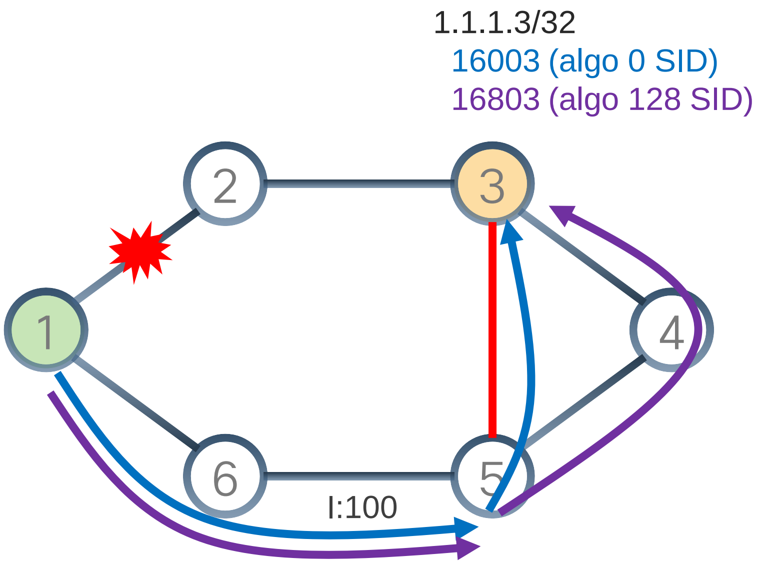
```
router isis 100
 affinity-map foo bit-position 0
 affinity-map baa bit-position 2
 !
 flex-algo 140
  advertise-definition
  affinity include-any foo baa
 !
 interface TenGigE0/0/0/0
  affinity flex-algo foo baa
 !
 interface TenGigE0/0/0/1
  affinity flex-algo baa
 !
```

Topology Independent LFA (TI-LFA)

- TI-LFA algorithm is performed within the topology of a given Flex-Algo instance
- Backup path is expressed with Prefix-SIDs of the Flex Algo

- Benefits: the backup path is optimized per Flex-Algo !!!

Example – TI-LFA Backup path per Algo (SR-MPLS)



At node 1 for destination 3

```
16003 => 16003 via 2
  backup: <24065, 16003> via 6

16803 => 16803 via 2
  backup: <24065, 16803> via 6
```

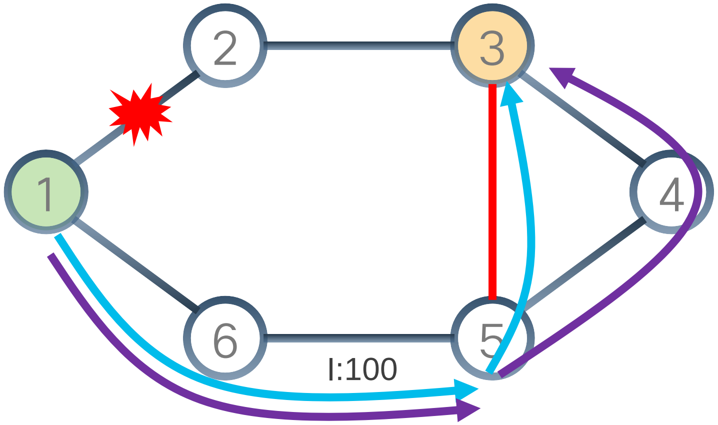
Usage of Algo-128 Prefix-SID 16803 ensures that the Algo 128 backup path also avoids the red link

IGP link metric: l:10 (default)
Adj SID convention: 240XY Adj SID from node X to node Y

Example – TI-LFA Backup path per Algo (SRv6)

SRv6 Locators @ 3:

FCBB:BB00:3::/48 (algo 0)
FCBB:BB08:3::/48 (algo 128)



FIB at node 1 for Locators of 3

```
FCBB:BB00:3::/48 via 2
 backup: Repair nodes R6, R5
 <FCBB:BB00:6:F6A5::, via 6

FCBB:BB08:3::/48 via 2
 backup: Repair nodes R6, R5
 <FCBB:BB08:6:F6A5::, via 6
```

Algo-128 Locator of 3 ensures that the Algo-128 backup path also avoids the red link

IGP link metric: I:10 (default)
uA SID convention (FXAY) – uA SRv6 Function at Router X associated with L3 neigh. Y

Flex-Algo – Disabling TI-LFA

- Usecase
- A Flex-algo instance for low-tier services without any fast-reroute guarantees

Flex-Algo – Disabling TI-LFA

```
RP/0/RP0/CPU0:R1(config)#router isis 100
RP/0/RP0/CPU0:R1(config-isis)# flex-algo 128
RP/0/RP0/CPU0:R1(config-isis-flex-algo)#?
  <...>
  fast-reroute          Configure Fast ReRoute
  <...>
RP/0/RP0/CPU0:R1(config-isis-flex-algo)#fast-reroute disable
```

```
router isis 100
 flex-algo 128
  fast-reroute disable
  metric-type delay
  advertise-definition
```

ISIS – Max-Paths enhanced granularity

- Usecase:

- Tight control of ASIC resources used for ECMP programming
  - Max-path = 1 => eliminates ECMP and programs unipath
- Various granularity:
  - max-paths per-algo + per-address-family — IOS-XR 7.8.1
  - max-paths per-algo + per-address-family and per-prefix — IOS-XR 7.11.1
- Applicable to SR-MPLS and SRv6

IOS-XR 7.8.1

IOS-XR 7.11.1

ISIS – Max-Paths enhanced granularity

```
router isis tag
 algorithm 0
  address-family {ipv4 | ipv6} unicast
   maximum-paths {max_path_val | route-policy rpl_name}
```

max_path_val == 1 programs single path (unipath)

```
router isis tag
 flex-algo algo_num
  address-family {ipv4 | ipv6} unicast
   maximum-paths {max_path_val | route-policy rpl_name}
```

IOS-XR 7.8.1

IOS-XR 7.11.1

ISIS – Max-Paths enhanced granularity

Example – per-prefix granularity (SR-MPLS)

```
prefix-set sample-pfx-set
  1.1.1.100/32
end-set
!
route-policy sample-rpl
  if destination in sample-pfx-set then
    set maximum-paths 1
  endif
end-policy
!
router isis 100
 flex-algo 128
  address-family ipv4 unicast
   maximum-paths route-policy sample-rpl
```


Flex-Algo – Example: No ECMP, no LFA/TI-LFA

- Usecase:
- Provide low-tier services with minimal transport SLA guarantees
  - No ECMP
  - No LFA / TI-LFA

```
router isis 100
 flex-algo 128
  fast-reroute disable
  advertise-definition
  address-family ipv4 unicast
   maximum-paths 1
```

Flex-Algo Steering

Flex-Algo Steering

- SR-MPLS
  - Leverages an SRTE policy (on-demand / manual) with a sid-list of a single label (flex algo prefix-sid of intended destination)
  - Automated steering is used to steer service routes with color over matching SR policies

Flex-Algo Steering

- SRv6
  - Ingress PE directly encapsulates traffic based on the Service SID advertised in BGP
    - Service SID = algo locator + decap function
    - Service SID directly encodes the transport intent (algo locator)
  - Does not require an SR-TE policy
  - Does not require to color service routes

Flex-Algo Steering – SR-MPLS

```
policy sample-POLICY
   color 100 end-point ipv4 1.1.1.4
   candidate-paths
    preference 100
     dynamic
     !
     constraints
      segments
       sid-algorithm 128
      !
     !
    !
   !
  !
```

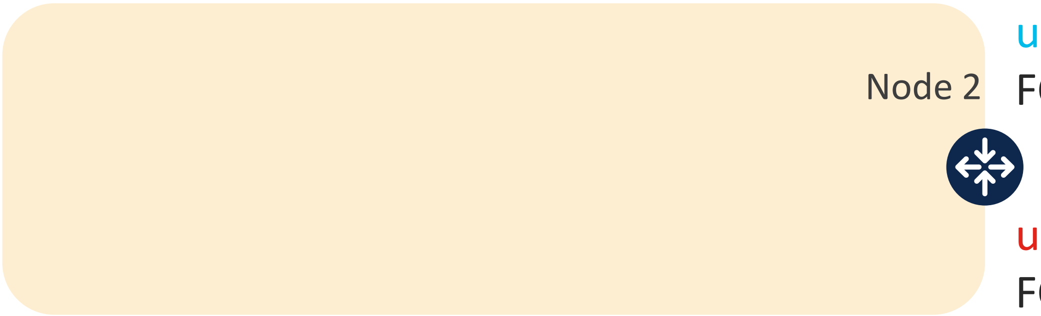
Flex-Algo Steering – SR-MPLS

- SRTE Automated Steering is leveraged for IGP Flex-Algo

```
segment-routing
  traffic-eng
    on-demand color 100
      dynamic mpls
         sid-algorithm 128
```

"Any 100-colored BGP route should be steered
via the prefix-SID(ALGO 128) of the BGP nhop"

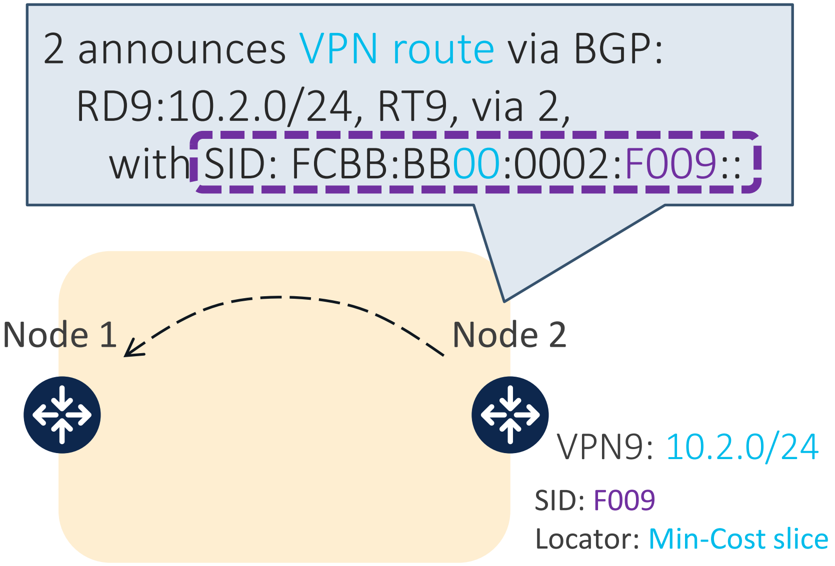
Flex-Algo Steering - SRv6



- A node gets a Shortest-Path Endpoint uSID (uN) from each slice
- A uN is a /48 off the /32 of the related slice
- Classic Prefix-Based Routing (CIDR)

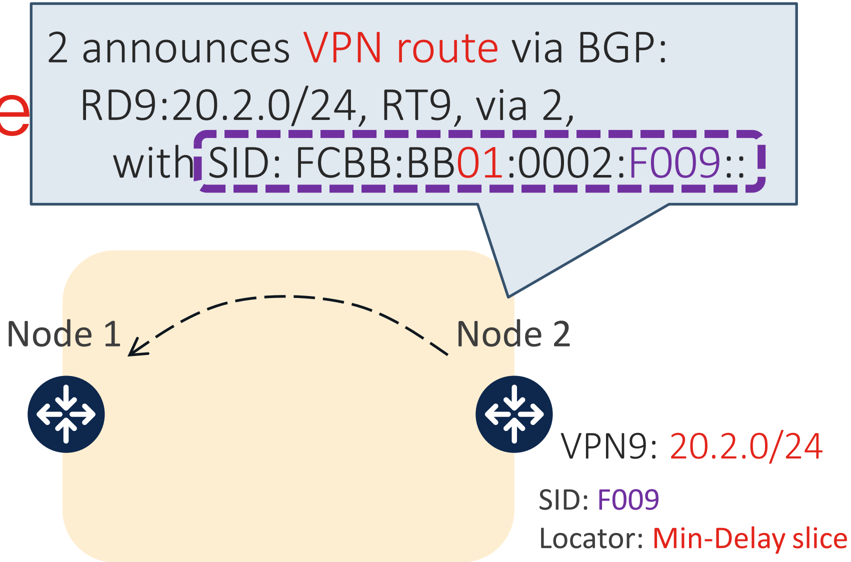
BGP Advertisement

- Intuitive uSID program:
  - Within the Min-Cost Slice (FCBB:BB00)
  - Follow the shortest-path to 2 (0002)
  - Execute VPN9 Decaps at 2 (F009)
- Seamless Deployment
  - Any transit node (SRv6 capable or not) routes on a classic /48
- Hardware Efficiency
  - Egress PE 2 processes multiple uSIDs with a single /64 lookup
  - FCBB:BB00:0002:F009/64



BGP Advertisement per Slice

- Intuitive uSID program:
  - Within the Min-Delay Slice (FCBB:BB01)
  - Follow the shortest-path to 2 (0002)
  - Execute VPN9 Decaps at 2 (F009)
- Hardware Efficiency
  - Egress PE 2 processes multiple uSIDs with a single /64 lookup
  - FCBB:BB01:0002:F009/64



Flex-Algo Operational Considerations

Flex-Algo – Operational Considerations

- Flex-Algo with metric-type == TE
  - Explicitly configure the FA TE metric link attribute under the participating interfaces, independently of the legacy TE metric
  - If not done, IOS-XR implementation, by default, would advertise the legacy TE metric (configured under SRTE) as a FA ASLA link attribute

Flex-Algo – Operational Considerations

- Number of Flex-Algo instances in the network
  - Treat a FA instance as a network-wide transport SLA intent
  - All services associated with this transport intent share a FA instance
  - Do not treat a FA as specific for one service

- Number of Flex-Algo instances in a node
  - Recommended to be in the single-digit range

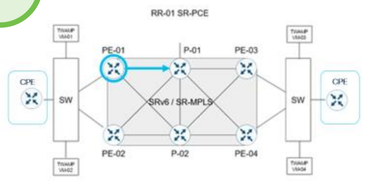
Flex-Algo – Operational Considerations

- SRGB considerations (SR-MPLS)
  - A node participating in a FA instance has a Node SID allocated for the FA
  - Consider the expected number of Flex-Algo instances in the network when planning the SRGB size

```
router isis core
 interface Loopback0
  passive
  address-family ipv4 unicast
   prefix-sid absolute 16006
   prefix-sid algorithm 128 absolute 18006
```

SR Performance Monitoring

1



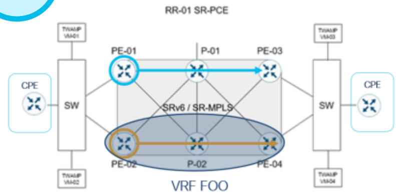
Link Measurement

Use Case: Measure the distance between PE-01 and P-01 (optical distance via HW timestamping)

Behaviors:

1. Provide min, max, avg, variance via telemetry
2. Update IGP with min

2



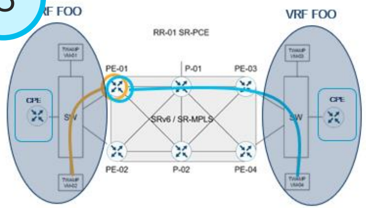
IP Endpoint Measurement

Use Case: Measure delay to endpoint (using best IGP path) can be within VRF. Note that measurement for multiple VRF will be the same

Behaviors:

1. Provide min, max, avg, variance via telemetry
2. Endpoint is any TWAMP reflector**

3



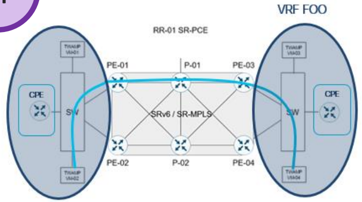
VRF IP Endpoints Measurement

Use Case: Measure delay to endpoint inside *VRF FOO.* Needs two probe sessions for remote and local

Behaviors:

1. Provide min, max, avg, variance via telemetry
2. Endpoint is any TWAMP reflector*

4



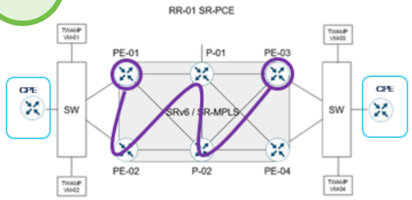
External Probes Model

Use Case: Measure Delay / Jitter / Packet Loss end to end

Behaviors: Provide min, max, avg, variance via telemetry GNMI or Rest.

Opportunity: Collect SR-PM data via telemetry collector

5



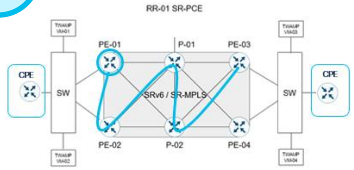
SR-TE Endpoint Measurement

Use Case: Measure delay for specific SR-TE policy

Behaviors:

1. Provide min, max, avg, variance via telemetry
2. Endpoints any TWAMP reflector**

6



IP Endpoint Meas. via SID list

Use Case: Measure delay to endpoint <via> "some" SID list (mimic an SR policy behavior)

Behaviors:

1. Provide min, max, avg, variance via telemetry
2. Endpoint is any TWAMP reflector**

*multi-vendor including Accedian **multi-vendor, including XR

SRv6 uSID

The Path to Ultimate Simplicity

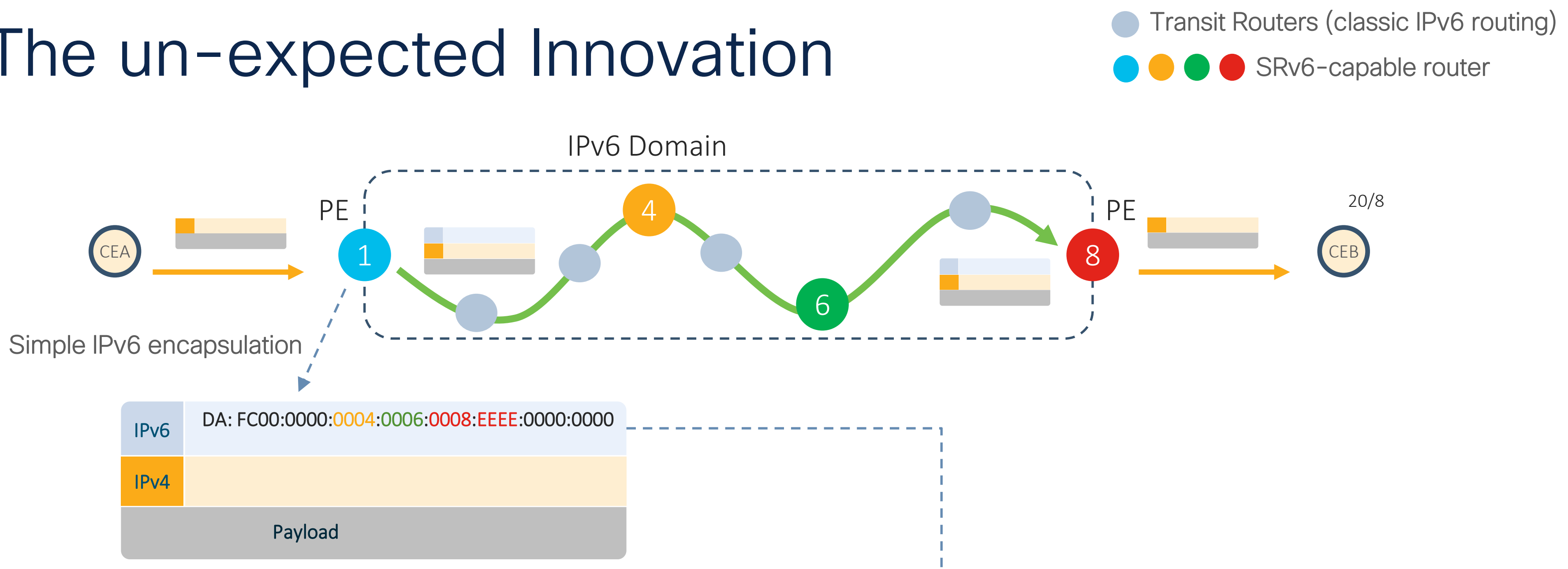
The un-expected Innovation

Using IP protocol differently
than anyone else imagined in the past

Any service without any shim (MPLS, VxLAN…)

With Better Scale, Reliability, Cost
and Seamless Deployment in Brownfield

The un-expected Innovation



A **source-routed path encoded in a single IPv6 address**!

- follow igp shortest-path to node 4,
- then shortest-path to node 6,
- then shortest-path to node 8,
- then decapsulate and lookup in VPN table

Simple IPv6 encapsulation – IPv6 uSID

- Most of the nodes just perform classic IPv6 routing as defined 25 years ago
- Some of the nodes enable and use the uSID network programming by simply using the available space in the outer IPv6 DA
- IPv6 Segment Routing Header (SRH) is rarely used but available for ultra-scale use-case

Grand Architecture with HW-Efficiency

- Revolutionary Network Programming Model (Turing Complete)
  - The IPv6 Destination Address (DA) holds multiple instructions
    - E.g., up to 6 instructions with 4-byte SRv6 block, 2-byte uSID's
    - E.g., up to 14 instructions with 2-byte SRv6 block, 1-byte uSID's
  - SRH IPv6 extension header holds additional instructions (rarely needed)
- Any behavior can be bound to the instruction
  - Shortest path according to cost, latency with exclusion of unsecured links
  - TDM-alike behavior (one instruction per hop/interface)
  - VPN Service
  - TE, FRR, NFV, Cryptography…
- Linerate across our entire portfolio

Novel Architecture with Brownfield

- Classic Longest-Match at Legacy IP node
- The network program is opaque to legacy node
- Alibaba, Swisscom, Bell … are all brownfield deployments

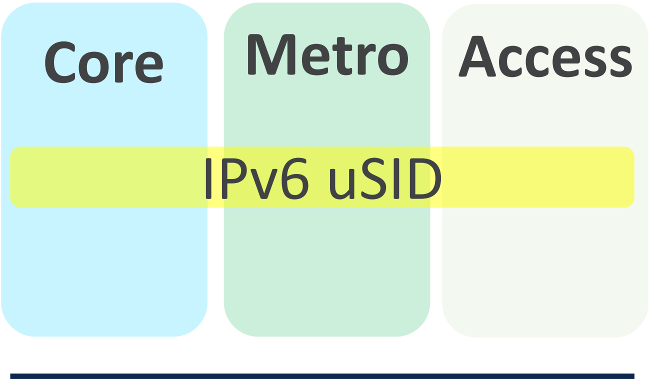
swisscom (video)

Alibaba.com (video)

Bell (video)

Operator Endorsement across Unified Solution

The Obvious Solution



Dan Voyer
Bell Canada
Paris 2022

Operator Endorsement across Unified Solution

The Universal Solution !

Core	Metro	Access	NFV	DC	Cloud	Host

IPv6 uSID

Dan Voyer
Bell Canada
Paris 2022

Dan Bernier
Bell &
NoviFlow
Paris 2022

Gyan
Mishra
Verizon
Paris 2023

Dan Bernier
Bell Canada
Paris 2023

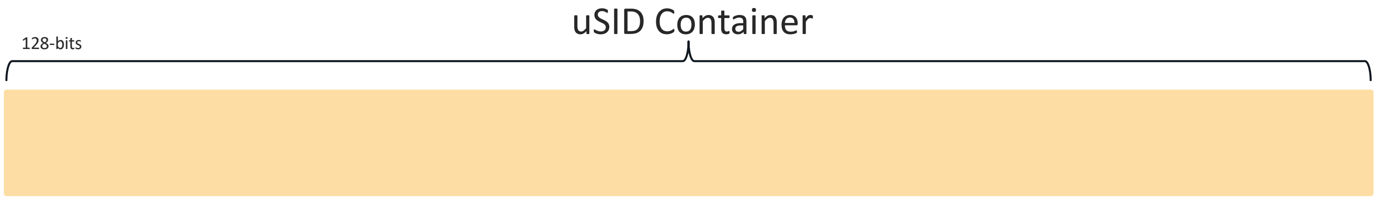
MPLS WC 2023 Session Recordings and Demos: http://segment-routing.net/conferences/Paris23

SRv6 uSID Terminology

- Industry terms:
  - SRv6 Micro Segment
  - SRv6 uSID
  - Abbreviation: uSID
- IETF terms:
  - Next Compressed-SID (NEXT-C-SID)
  - Abbreviation: Next
  - IETF document: draft-ietf-spring-srv6-srh-compression

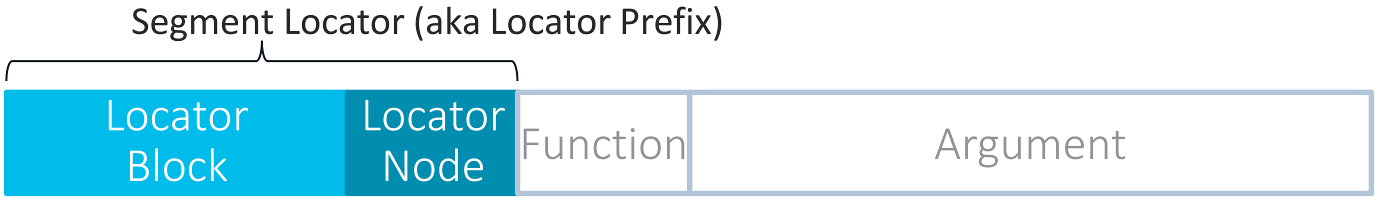
SRv6 uSID Segment Locators

SRv6 uSID Format



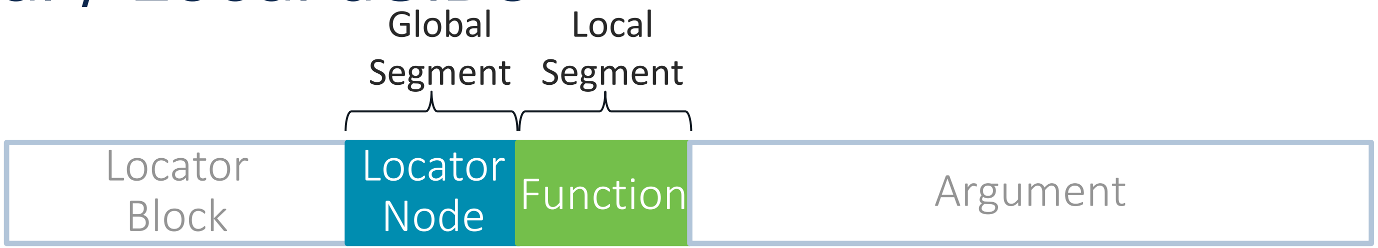
- uSID Container is a 128-bit SRv6 SID containing a sequence of uSIDs
- It can be encoded in the destination address (DA) field of an IPv6 header or at any position in the Segment List of a Segment Routing Header (SRH)

SRv6 uSID Format



- A Segment Locator is composed of:
  - uSID Locator Block – a block of uSIDs allocated from an IPv6 prefix available to the provider
  - uSID Locator Node – an identifier of the parent node instantiating the uSID
- The Locator leads traffic to the endpoint node which instantiates the SID
- It provides a context for the execution of the function

Global / Local uSIDs



- Global Segments – learnt and programmed by all nodes in the SR domain
- Local Segments – programmed only by the advertising node

Locator Block	Locator Node	Function	Argument

uSID – An Illustration

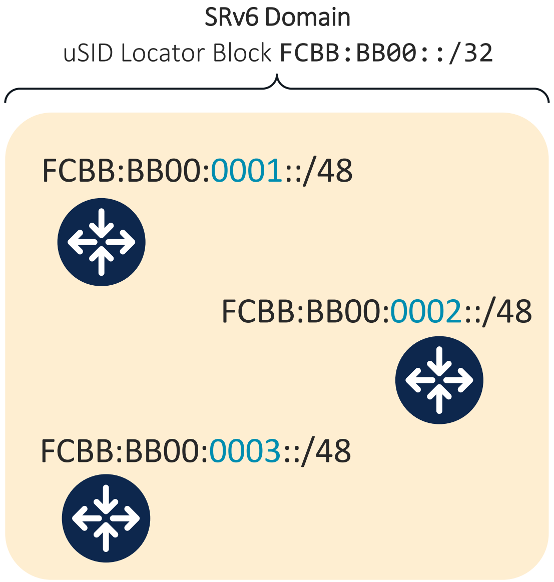
- For illustration, we will use:
  - uSID Locator Block length: 32 bits
  - uSID Locator Block:

FCBB:BB00::/32 with **B** a nibble value picked by operator

  - uSID length (Locator Node ID / Function ID): 16 bits
  - uSID:

FCBB:BB00:XYWZ::/48 with **XYWZ** variable nibbles

- A uSID FCBB:BB00:XYWZ::/48 is said to be allocated from its block (FCBB:BB00::/32)



SRv6 uSID Configuration

```
segment-routing
 srv6
  locators
   locator MAIN
    micro-segment behavior unode psp-usd
    prefix fcbb:bb00:1::/48
```

Name to reference

uSID

Locator Prefix

SRv6 ISIS Configuration

```
router isis 1
 address-family ipv6 unicast
  segment-routing srv6
   locator MAIN
```

Name of the Locator

This will result in:

- Locator is advertised
- uN function is advertised
- uA for each ISIS interface is allocated and advertised

uSID Locator Addressing

Separation between SIDs and addresses

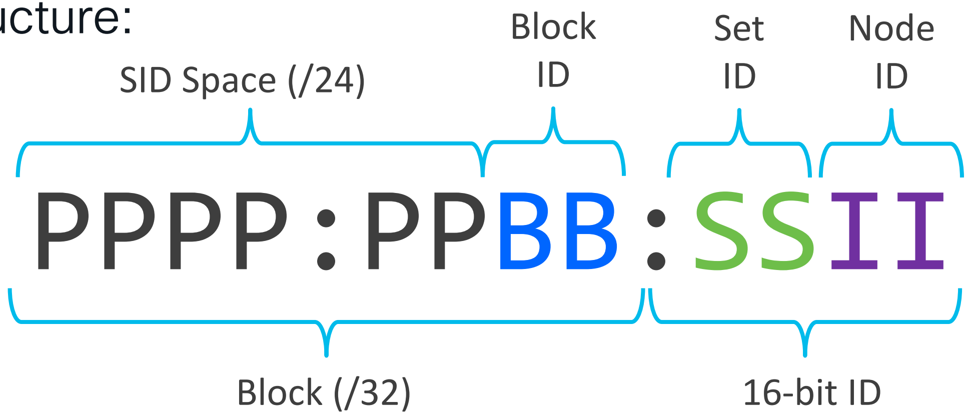
- Infrastructure addressing and SRv6 SID allocation belong to two different planes and are different
  - Infrastructure IP addresses (e.g., link interfaces, loopbacks) are allocated on the management plane
  - SRv6 SIDs are allocated on the service plane
- SRv6 SIDs are assigned to a node independently from the IP addressing of that node
- Even if they are both represented as IPv6 addresses, infrastructure addresses and SIDs cannot be merged and should be allocated off different blocks.

An existing IPv6 address plan is not a constraint
for a future SRv6 SID allocation plan

Terminology – uSID F3216

- uSID F3216: uSID format with
  - uSID Block size: 32 bits
  - ID size: 16 bits

- uSID F3216 structure:



SRv6 Space allocation recommendation

- Private range allocation ✓
  - Recommended allocation
  - Use /24 sub-range from ULA `FC00::/7` space
  - `FCBB:BB00::/24`, with `B` indicating a nibble value picked by operator

- Public range allocation ⚠
  - Supported, not advised
  - From allocated public GUA range

uSID Block per slice (Flex Algo) if possible

- 256 Blocks are available in the SRv6 Space:

  `FCBB:BBTT::/32`, with `TT` = slice ID

  - Multiple Blocks can be concurrently used on a node

- We assume 2 slices (Blocks), e.g.:
  - `FCBB:BB00::/32` Low-cost slice (algo 0) ← focus, other Blocks are similar
  - `FCBB:BB01::/32` Low-delay slice (algo 128)

Set Allocation Example

Block: FCBB:BB00::/32

Region 1	Core Region	Region 2
600 nodes ↓ 3 Sets	200 nodes ↓ 1 Set	400 nodes ↓ 2 Sets
FCBB:BB00:0200::/40 FCBB:BB00:0300::/40 FCBB:BB00:0400::/40	FCBB:BB00:0100::/40	FCBB:BB00:0500::/40 FCBB:BB00:0600::/40

- If a region outgrows its allocated Sets, then allocate more Sets to this region

uSID Allocation Example

Block: FCBB:BB00::/32

Region 1

600 nodes
3 Sets (02, 03, 04)

FCBB:BB00:0207::/48

FCBB:BB00:0301::/48

FCBB:BB00:0406::/48

Core Region

200 nodes
1 Set (01)

FCBB:BB00:0107::/48

Region 2

400 nodes
2 Sets (05, 06)

FCBB:BB00:0507::/48

FCBB:BB00:0601::/48

- Remaining unallocated uSIDs in Sets are for future growth

Summarization

Block: FCBB:BB00::/32

Region 1	Core Region	Region 2
600 nodes 3 Sets (02, 03, 04)	200 nodes 1 Set (01)	400 nodes 2 Sets (05, 06)

summarize →

FCBB:BB00:0200::/40
FCBB:BB00:0300::/40
FCBB:BB00:0400::/40

← summarize

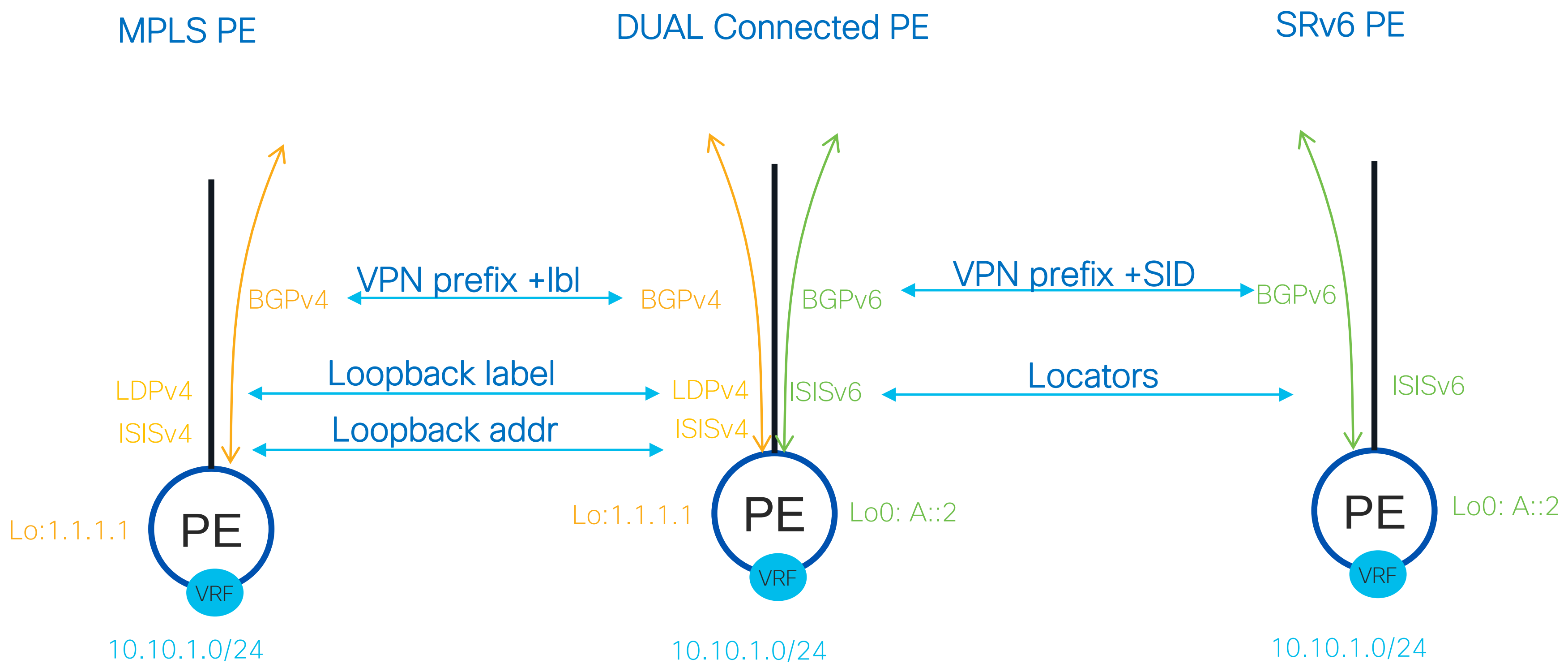
FCBB:BB00:0500::/40
FCBB:BB00:0600::/40

- Metro to Core – /40 summary routes provide a scaling gain factor of 256
- Core to Metro – just a single summary route
- Compare to MPLS where no summarization is possible – /32 route per node required

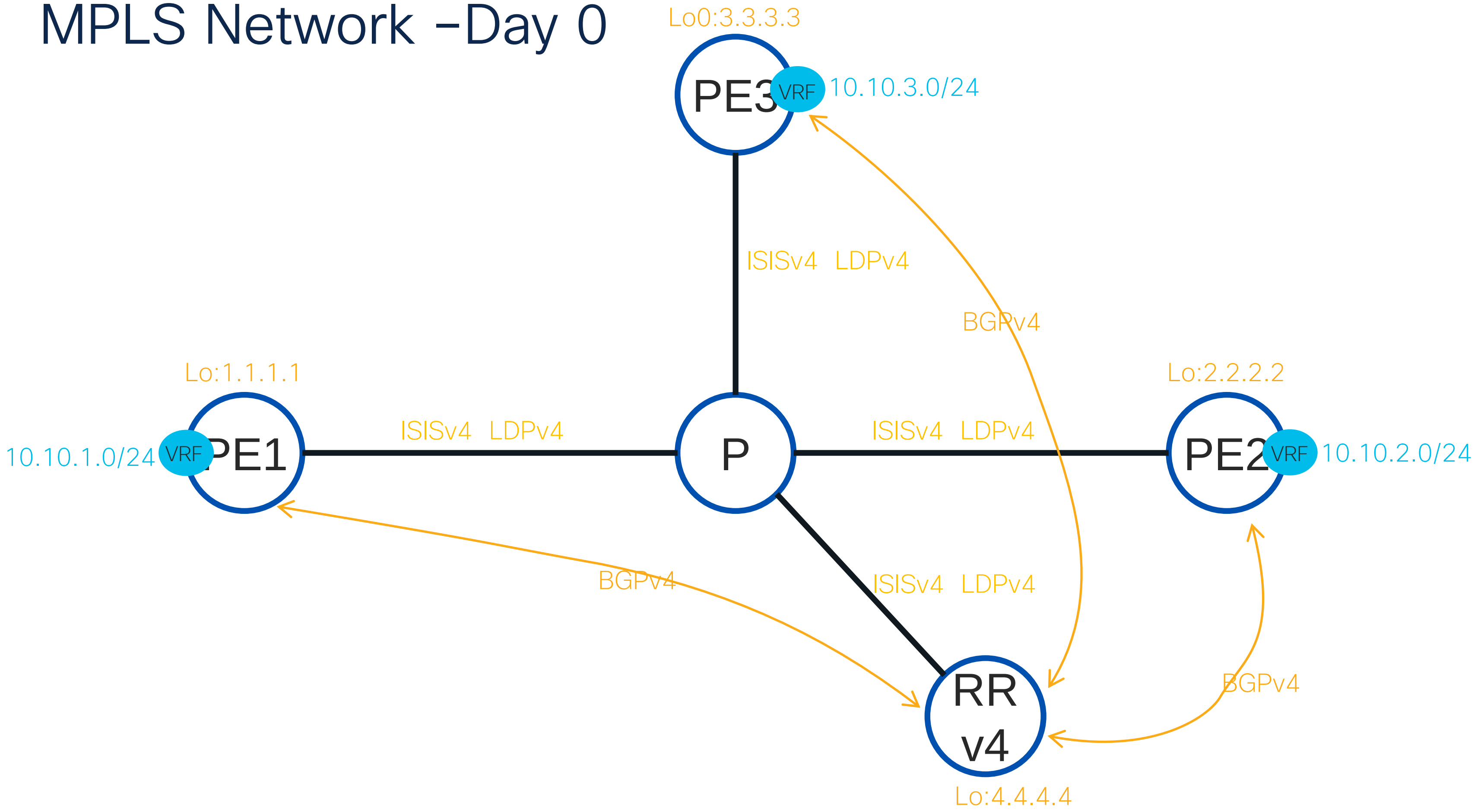
Lossless Migration

https://www.linkedin.com/pulse/lossless-migration-from-srmpls-usid-srv6-possible-frank-geldner/

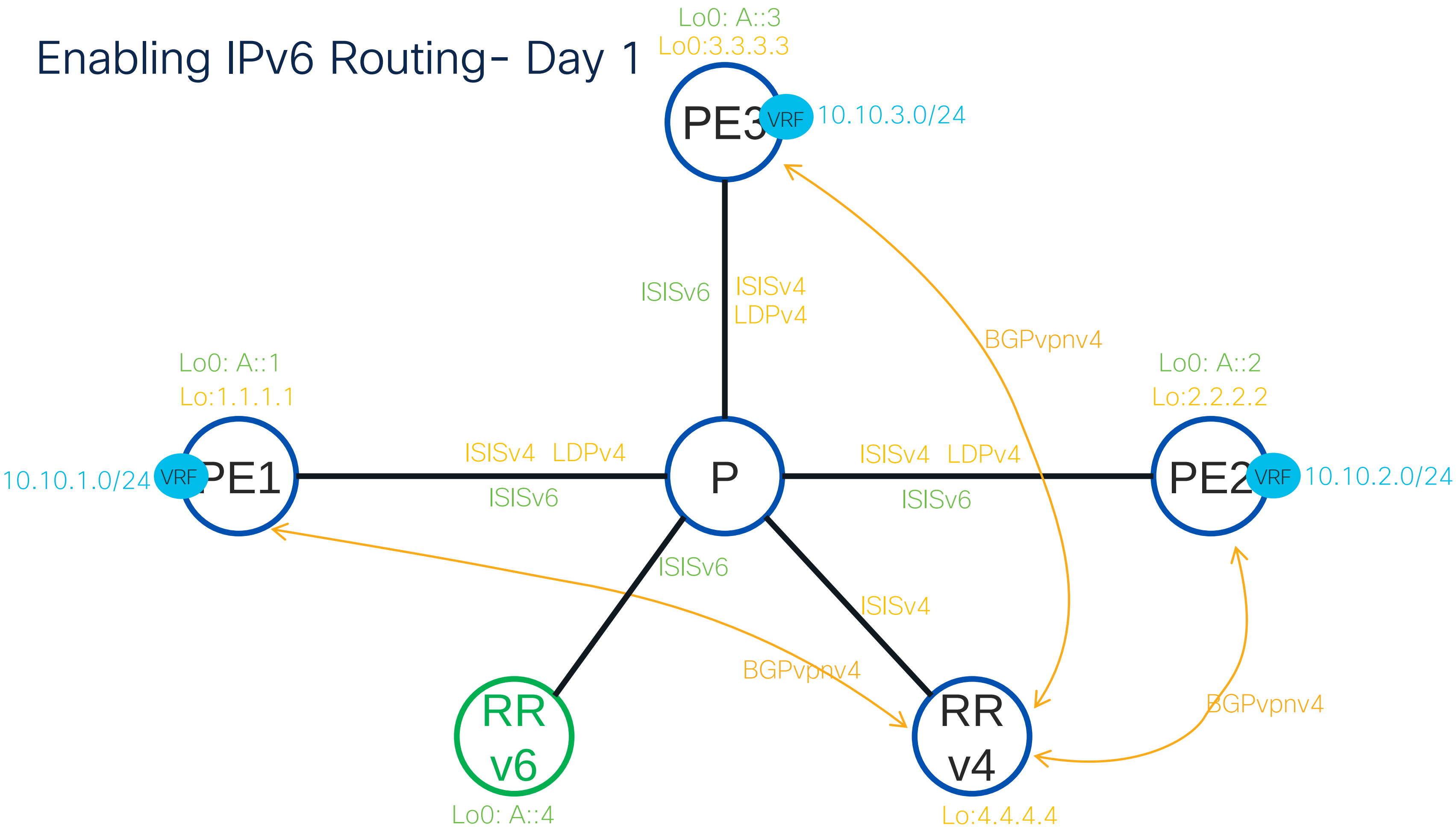
Dual Connected PE



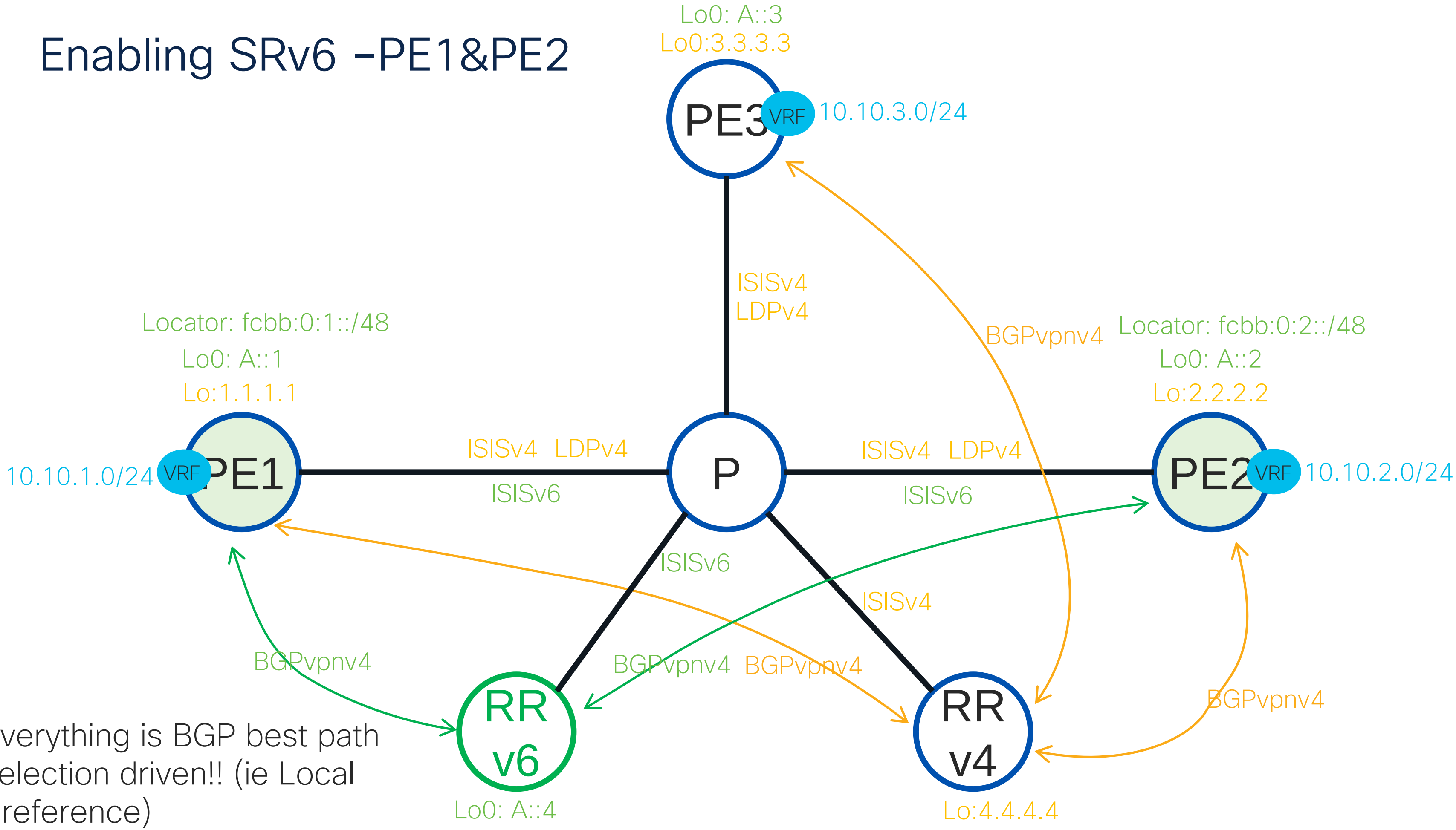
MPLS Network –Day 0



Enabling IPv6 Routing- Day 1

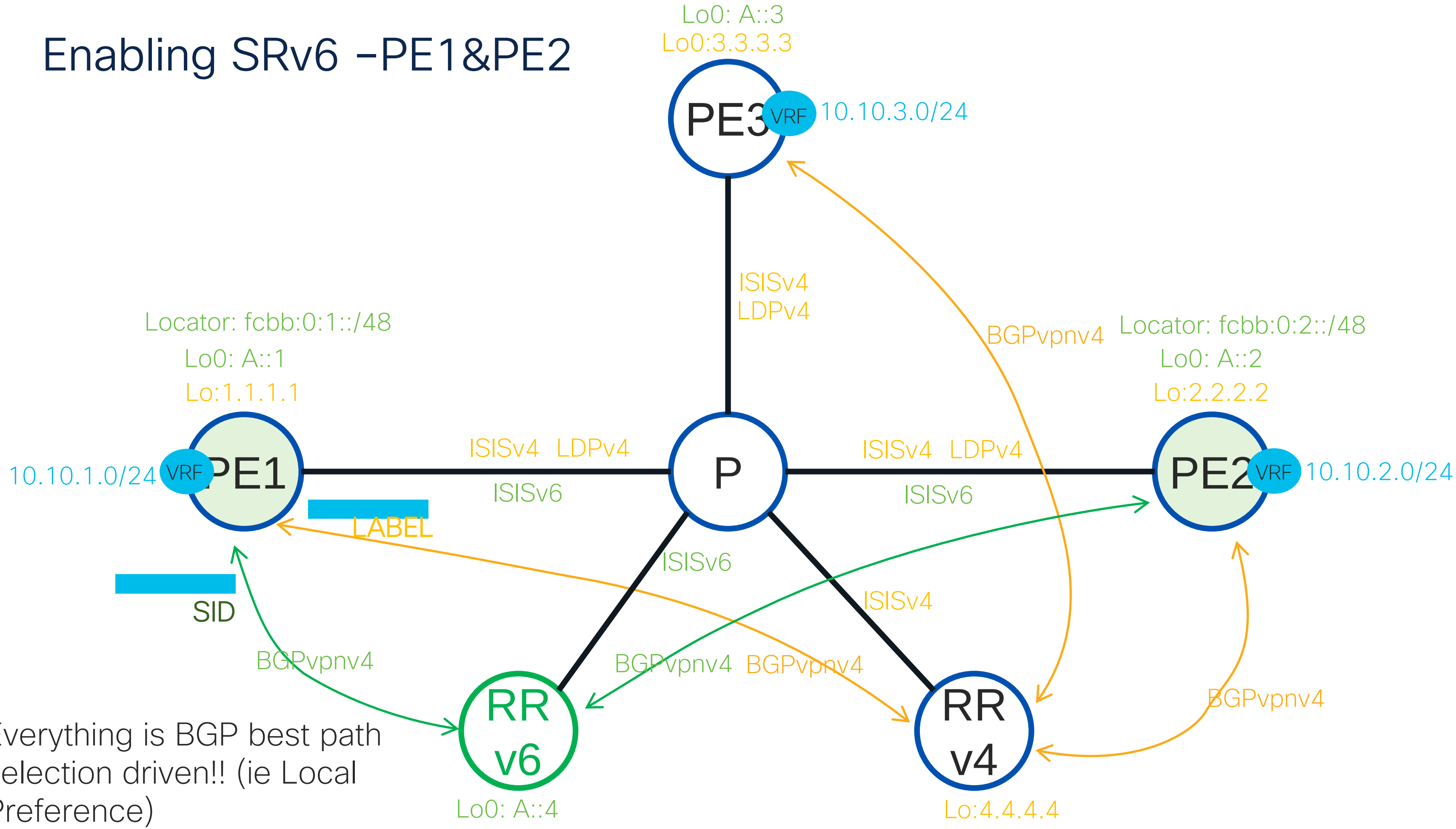


Enabling SRv6 –PE1&PE2



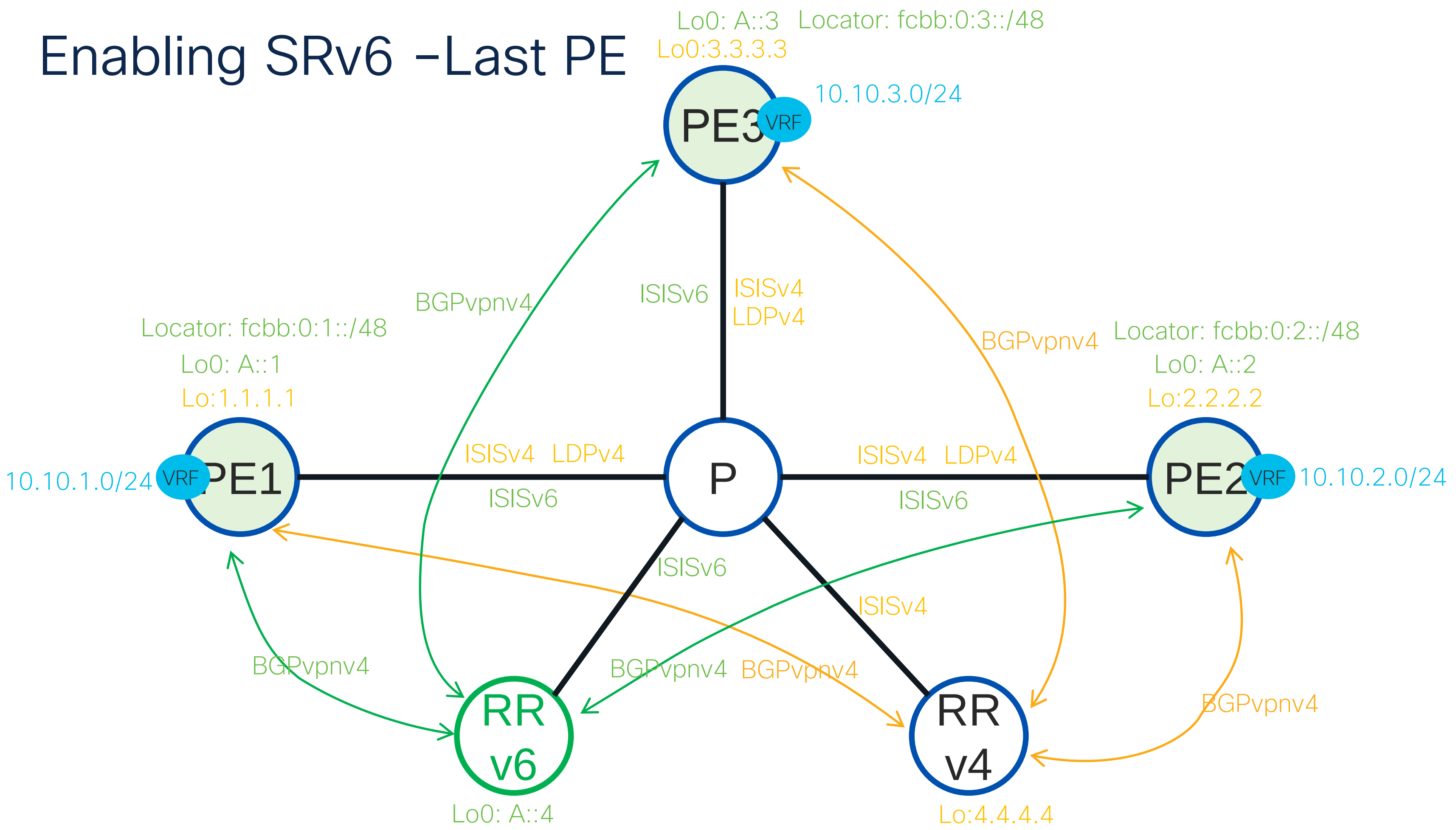
Everything is BGP best path selection driven!! (ie Local Preference)

Enabling SRv6 –PE1&PE2

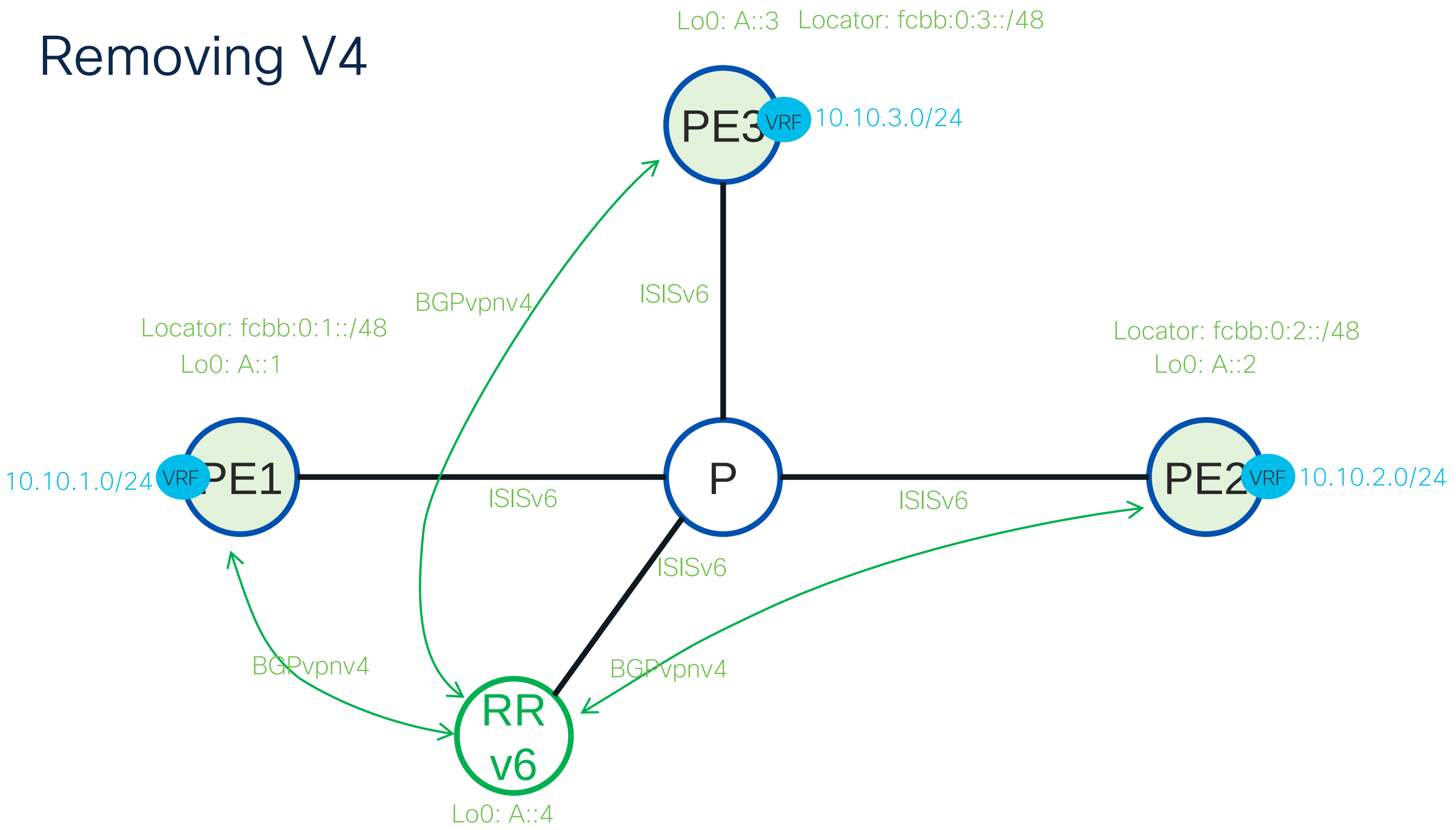


Everything is BGP best path selection driven!! (ie Local Preference)

Enabling SRv6 –Last PE



Removing V4



Lo0: A::3
Locator: fcbb:0:3::/48
PE3
VRF
10.10.3.0/24
BGPvpnv4
ISISv6
Locator: fcbb:0:1::/48
Lo0: A::1
Locator: fcbb:0:2::/48
Lo0: A::2
10.10.1.0/24
VRF
PE1
P
PE2
VRF
10.10.2.0/24
ISISv6
ISISv6
ISISv6
BGPvpnv4
BGPvpnv4
RR
v6
Lo0: A::4

SRv6 Dual PE Configuration

```
router bgp 1
 neighbor A::4
  address-family vpnv4 unicast
   encapsulation-type srv6
   route-policy RRv6 out
 neighbor 4.4.4.4
  address-family vpnv4 unicast
   route-policy RRv4 out
vrf 1
  address-family ipv4 unicast
   mpls alloc enable
   segment-routing srv6
    locator MAIN
    alloc mode per-vrf
```

Via RPL we set specific BGP attributes to to prefixes ie Local Preference towards RRv6 and RRv4

← Policy towards v6 RR (route-policy RRv6 out)

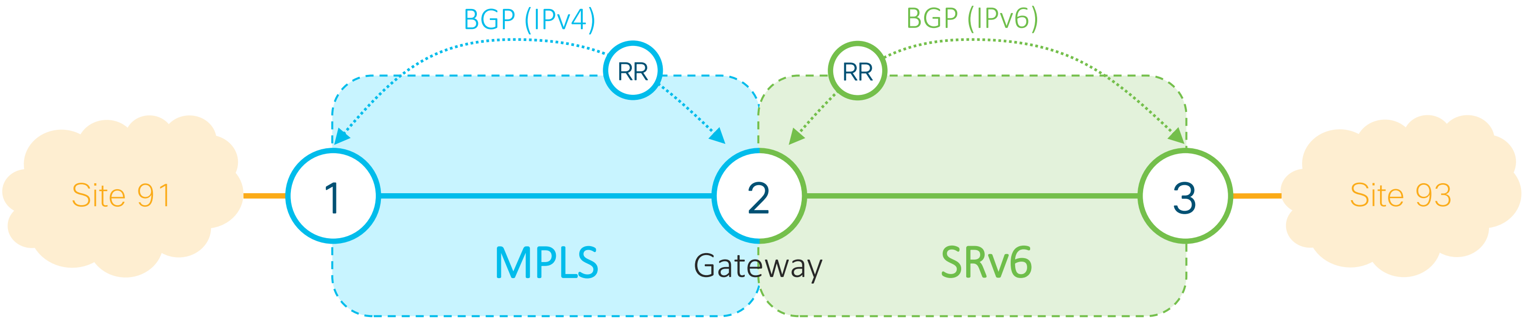
← Policy towards v4 RR (route-policy RRv4 out)

← Allocates Labels for all prefixes in VRF (mpls alloc enable)

← Allocates SIDs for all prefixes in VRF from Locator MAIN (locator MAIN)

SR-MPLS/SRv6 L3 Service Interworking Gateway

SRv6/MPLS L3 Service Interworking Gateway



- Gateway acts as intermediary for L3 services between SRv6 and MPLS domains
- Gateway provides service interworking on the control and data planes
  - Control plane: import service routes and re-advertise
  - Data plane: pop/decaps, IP lookup, encaps/push

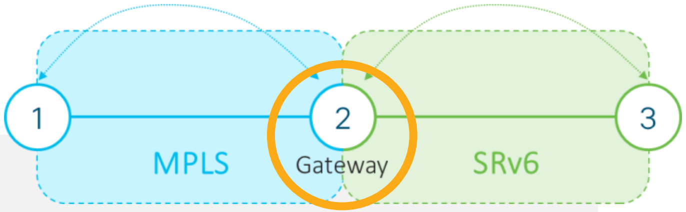
L3VPN service interworking

- Gateway acts as intermediary for interworked L3VPN services
  - GW terminates the L3VPN services
- GW has the VRFs configured that need SRv6/MPLS interworking with 2 sets of RTs
  - MPLS L3VPN RTs
  - SRv6 L3VPN RTs (called “stitching RTs”)
- GW imports service routes received from one domain (MPLS | SRv6)
- GW re-advertises exported service routes to the other domain (next-hop-self)
- GW stitches the service on the data plane (using VRF IP lookup of service route)

- Illustration shows VPNv4 example – also applies to VPNv6

Gateway configuration example

VPNv4 unicast




```
vrf ACME
 address-family ipv4 unicast
  import route-target
   1:1                ; MPLS
   2:2 stitching      ; SRv6
  !
  export route-target
   1:1                ; MPLS
   2:2 stitching      ; SRv6
```

Stitch MPLS domain RTs to SRv6 domain ("stitching") RTs (note: MPLS RTs can also be stitching with SRv6 RTs non-stitching)

Allocate VPN label and SRv6 SID

```
router bgp 100
 segment-routing srv6
  locator LOC1
 !
 neighbor 1.1.1.1
  address-family vpnv4 unicast
   import re-originate stitching-rt
   route-reflector-client
   advertise vpnv4 unicast re-originated
 !
 neighbor b:3::1
  address-family vpnv4 unicast
   import stitching-rt re-originate
   route-reflector-client
   encapsulation-type srv6
   advertise vpnv4 unicast re-originated stitching-rt
 !
 vrf ACME
  address-family ipv4 unicast
   mpls alloc enable
   segment-routing srv6
```

MPLS VPN peer or RR

Import "MPLS" RT and re-originate with "SRv6" "stitching" RT

Advertise with "MPLS" RT

SRv6 VPN peer or RR

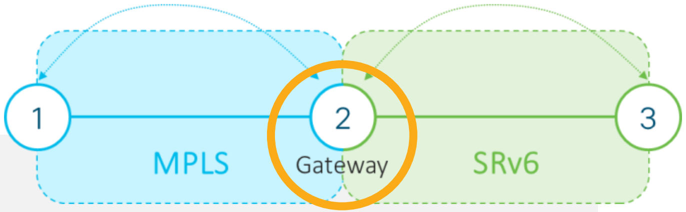
Import "SRv6" "stitching" RT and re-originate with "MPLS" RT

send the SRv6 SID to the neighbor

Advertise with "SRv6" "stitching" RT

Gateway configuration example

VPNv6 unicast




```
vrf ACME
 address-family ipv6 unicast
  import route-target
   1:1                ; MPLS
   2:2 stitching      ; SRv6
  !
  export route-target
   1:1                ; MPLS
   2:2 stitching      ; SRv6
```

Stitch MPLS domain RTs to SRv6 domain ("stitching") RTs (note: MPLS RTs can also be stitching with SRv6 RTs non-stitching)

```
router bgp 100
 segment-routing srv6
  locator LOC1
 !
 neighbor 1.1.1.1
  address-family vpnv6 unicast
   import re-originate stitching-rt
   route-reflector-client
   advertise vpnv6 unicast re-originated
 !
 neighbor b:3::1
  address-family vpnv6 unicast
   import stitching-rt re-originate
   route-reflector-client
   encapsulation-type srv6
   advertise vpnv6 unicast re-originated stitching-rt
 !
 vrf ACME
  address-family ipv6 unicast
   mpls alloc enable
   segment-routing srv6
```

MPLS VPN peer or RR

Import "MPLS" RT and re-originate with "SRv6" "stitching" RT

Advertise with "MPLS" RT

SRv6 VPN peer or RR

Import "SRv6" "stitching" RT and re-originate with "MPLS" RT

send the SRv6 SID to the neighbor

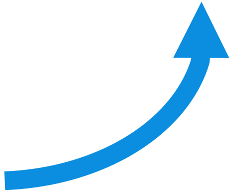
Advertise with "SRv6" "stitching" RT

Allocate VPN label and SRv6 SID

Conclusion

Simplicity Always Prevails

Furthermore with more scale and functionality



Stay up to date with...

SR | Segment Routing
www.segment-routing.net/

in | LinkedIn
www.linkedin.com/groups/8266623

Twitter
www.twitter.com/SegmentRouting

f | Facebook
www.facebook.com/SegmentRouting/

Segment Routing, Part I / II Textbooks
Available on Kindle and in paperback

cisco Live!

Did you know?

You can have a
one-on-one session with
a technical expert!

Visit Meet the Expert in The HUB
to meet, greet, whiteboard & gain
insights about your unique questions
with the best of the best.

Meet the Expert Opening Hours:

Tuesday	**3:00pm – 7:00pm**
Wednesday	**11:15am – 7:00pm**
Thursday	**9:30am - 4:00pm**
Friday	**10:30am - 1:30pm**

Session Surveys

We would love to know your feedback on this session!

- Complete a minimum of four session surveys and the overall event surveys to claim a Cisco Live T-Shirt

Continue your education

- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Expert meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand

SR Learning Path

- SRv6 Universality and E2E Network Transformation
  - TECSPG-3052, Dec 05 9:00-1:00PM AEDT
- A Guide to a Successful SR Deployment
  - BRKSPG-2510, Dec 06 1:00-2:30PM AEDT
- Protect Your Network Using Programmability and Segment Routing
  - BRKSPG-2125, Dec 06 2:40-3:40PM AEDT
- SRv6 for Next-Generation Transport Networks
  - BRKSPG-1676, Dec 06 4:00-5:30PM AEDT
- Exploring SR MPLS: Unleashing Network Scalability and Efficiency
  - LTRSPG-2762, Instructor-led Lab, Dec 07 8:30-10:30AM AEDT

SR Learning Path

- Circuit-Style SR with Crosswork Network Controller
  - DEVWKS-2063, DevNet Workshop, WED Dec 06 12:00-12:45PM AEDT, Dec 07 2:30-3:15PM AEDT
- Cisco IOS-XR Segment Routing 101
  - LABSPG-1785, Walk-in Lab
- Cisco IOS-XR EVPN VPWS Service over SR-TE
  - LABSPG-2782, Walk-in Lab
- Cisco IOS-XR SR Flexible Algorithm
  - LABSPG-2783, Walk-in Lab
- Cisco SR BGP Prefix-SID in Inter-AS Network Lab
  - LABSPG-2784, Walk-in Lab

The bridge to possible

Thank you

Let's go

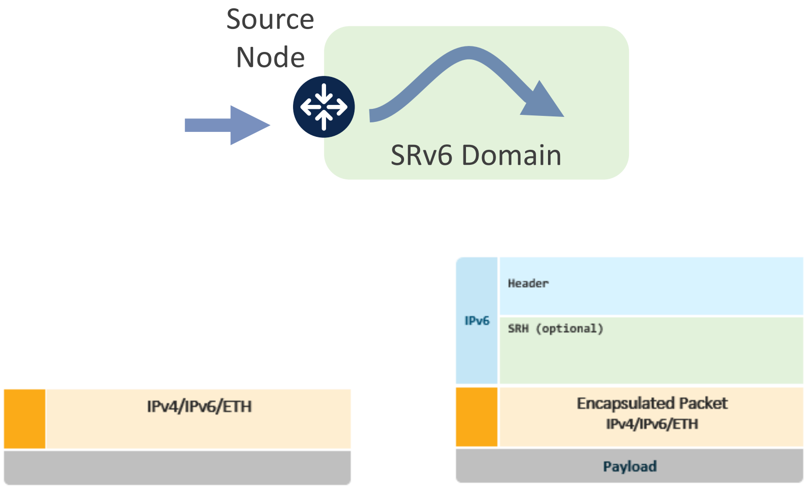
Appendixes

SRv6 Fundamentals

The Beginning

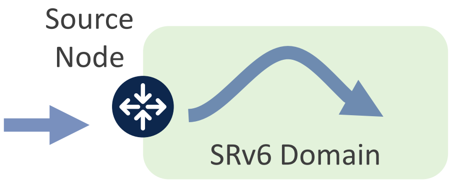
Segment Routing over IPv6 (SRv6)

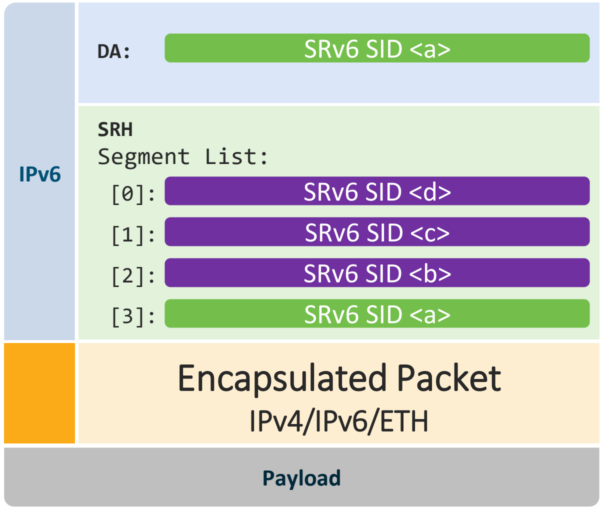
- SR applies to the IPv6 data plane with a native IPv6 extension
- A new Routing Header (RH) type called the Segment Routing Header (SRH)
- A source-routed path is encoded as an ordered list of Segments
- Source Node encapsulates incoming packet/frame in an outer IPv6 header, followed by an optional SRH



Segment Routing over IPv6 (SRv6)

- When a packet is steered onto a source-routed path, and if an SRH is required, the related SRH is added to the packet
- The Active Segment is encoded in the destination address (DA) of the outer IPv6 header
- SRH carries the ordered list of SRv6 SIDs associated with the source-routed path





Introducing SRv6 uSID

SRv6 uSID Terminology

- Industry terms:
  - SRv6 Micro Segment
  - SRv6 uSID
  - Abbreviation: uSID
- IETF terms:
  - Next Compressed-SID (NEXT-C-SID)
  - Abbreviation: Next
  - IETF document: draft-ietf-spring-srv6-srh-compression

Efficient Compressed Encoding

- In an SRv6 domain, the SIDs are allocated from a particular IPv6 prefix – the SRv6 SID Locator Block
- All SRv6 SIDs instantiated from the same Locator Block share the same most significant bits (Block bits (B))
- Furthermore, when the combined length of the SRv6 SID Locator, Function and Argument is smaller than 128 bits, the trailing bits are set to zero

Efficient Compressed Encoding

- When a sequence of consecutive SIDs in a Segment List shares a common Locator-Block, a compressed SRv6 Segment List encoding can optimize the packet header length by avoiding the repetition of the Locator-Block and trailing bits with each individual SID

Example
4 segment segment-list

```
fcbb:bb00:0002::
fcbb:bb00:0003::
fcbb:bb00:0004:e001::
fcbb:bb00:0005:e000::
```

Compressed SID-list encoding

```
fcbb:bb00:0002:0003:0004:e001:0005:e000
```

Compressed SRv6 Segment List Encoding

- Compressed Segment List Encoding – A segment list encoding that reduces the packet header length thanks to one or more uSID sequences

Uncompressed Segment List Encoding

```
IPv6  | SA: fcbb:bb00:0001::1                     ] 40B
      | DA: fcbb:bb00:0002::
      | NH: RH
      |--------------------------------------
      | RH Type: 4 (SRH)                       ] 72B*
      | NH: IPv4
      | Segments Left: 3
      | Segment List:
      | [0]: fcbb:bb00:0005:e000::
      | [1]: fcbb:bb00:0004:e001::
      | [2]: fcbb:bb00:0003::
      | [3]: fcbb:bb00:0002::
IPv4  | SA: 1.1.1.1
      | DA: 2.2.2.2
      | Port: UDP
Payload
```

Compressed SID-list encoding

Compressed Segment List Encoding

```
IPv6  | SA: fcbb:bb00:0001::1                                    ] 40B
      | DA: fcbb:bb00:0002:0003:0004:e001:0005:e000
      | NH: IPv4
IPv4  | SA: 1.1.1.1
      | DA: 2.2.2.2
      | Port: UDP
Payload
```

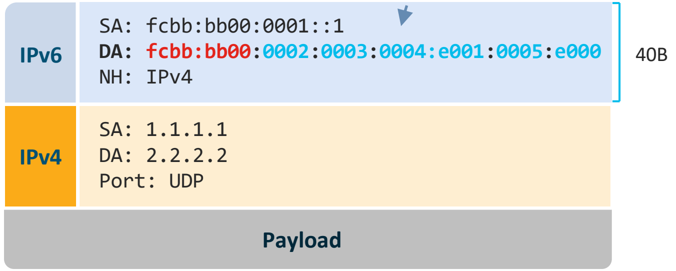
Note (*): MTU of SRH with “n” SIDs = ([n x 16] + 8) Bytes

Compressed SRv6 Segment List Encoding

A network program @ **R1**:
Encapsulate a packet for a destination in VPN acme at R5 following the min-cost slice, and that:

- Takes **shortest-path** to **R2**, then
- Takes **shortest-path** to **R3**, then
- Takes **shortest-path** to **R4**, then
- Takes **interface if1** to **R5**, then
- **Decapsulates and performs a table lookup** in VRF acme

A network program of six (6) 16-bit instructions encoded with only 40B of overhead!



Compressed SRv6 Segment List Encoding

A network program @ **R1**:
Encapsulate a packet for a destination in VPN acme at R5 following the min-cost slice, and that:

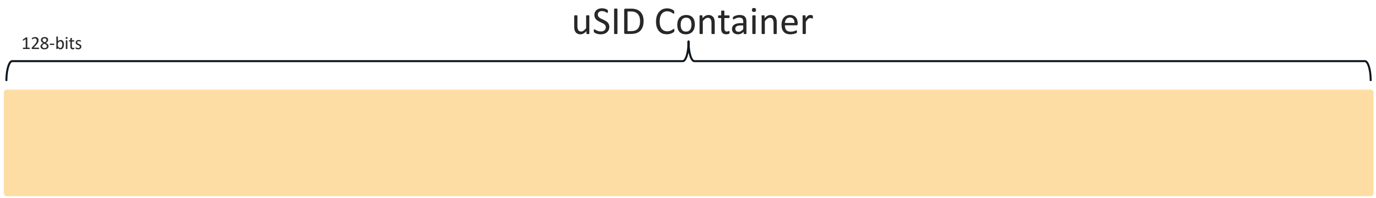
- Takes **shortest-path** to **R2**, then
- Takes **shortest-path** to **R3**, then
- Takes **shortest-path** to **R4**, then
- Takes **interface if1** to **R5**, then
- Gets **decapsulated and performs a table lookup** in VRF acme

Furthermore, a micro-program of less than 6 uSIDs only requires IP-in-IP encapsulation behavior at the source node
Most use-cases do not need an SRH!

IPv6	SA: fcbb:bb00:0001::1 **DA: fcbb:bb00:0002:0003:0004:e001:0005:e000** NH: IPv4	40B
IPv4	SA: 1.1.1.1 DA: 2.2.2.2 Port: UDP	
Payload		

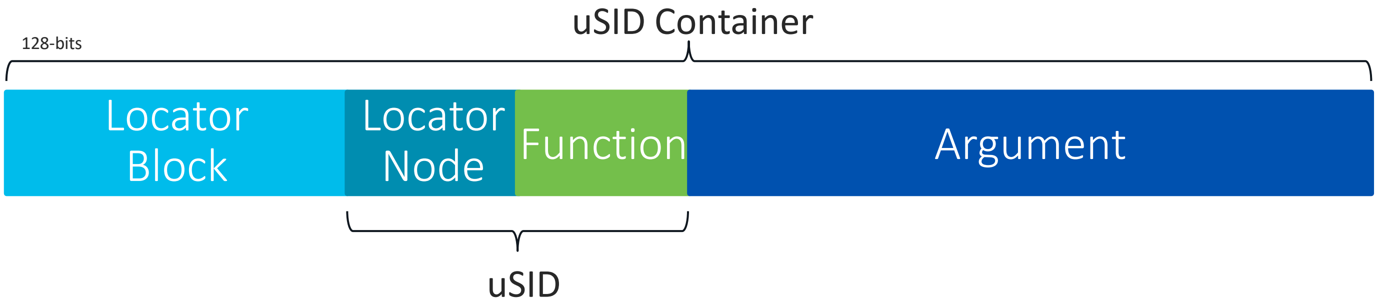
SRv6 uSID Format

SRv6 uSID Format



- uSID Container is a 128-bit SRv6 SID containing a sequence of uSIDs
- It can be encoded in the destination address (DA) field of an IPv6 header or at any position in the Segment List of a Segment Routing Header (SRH)

SRv6 uSID Format



- uSID Container consists of LOC-BLCK:LOC-NODE:FUNCT:ARG,
  - where a uSID Locator Block (LOC-BLCK) is encoded in the B most significant bits of the uSID Container
  - followed by NF bits of Locator Node ID and Function ID (LOC-NODE:FUNCT) and
  - followed by A bits of Argument (ARG)
  - Flexible bit-length format → B + NF + A <= 128 bits

SRv6 uSID Format

- A Segment Locator is composed of:
  - uSID Locator Block – a block of uSIDs allocated from an IPv6 prefix available to the provider
  - uSID Locator Node – an identifier of the parent node instantiating the uSID
- The Locator leads traffic to the endpoint node which instantiates the SID
- It provides a context for the execution of the function

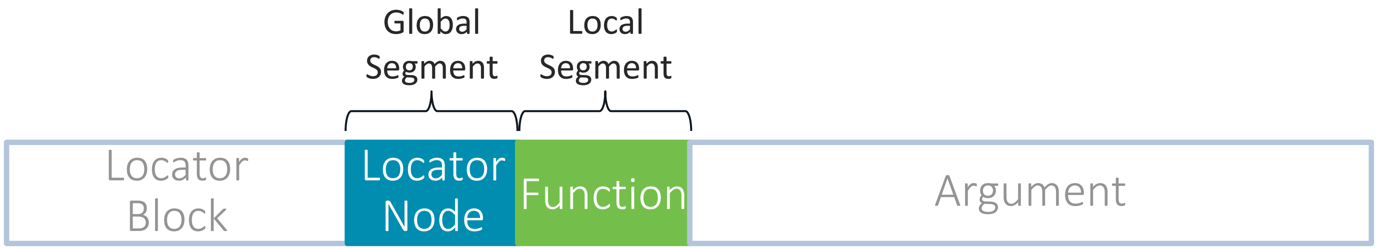
SRv6 uSID Format

- Function represents an opaque identification of a local behavior bound to the SID – aka SRv6 Behaviors
- It identifies the action to be performed by the endpoint node, in the context of a given locator
- A function can represent ANY action; e.g., topological or service-based or user-defined

SRv6 uSID Format

- Argument carries the remaining uSIDs (LOC-NODE:FUNCT) in the uSID Container

Global / Local uSIDs



- Global Segments – learnt and programmed by all nodes in the SR domain
- Local Segments – programmed only by the advertising node

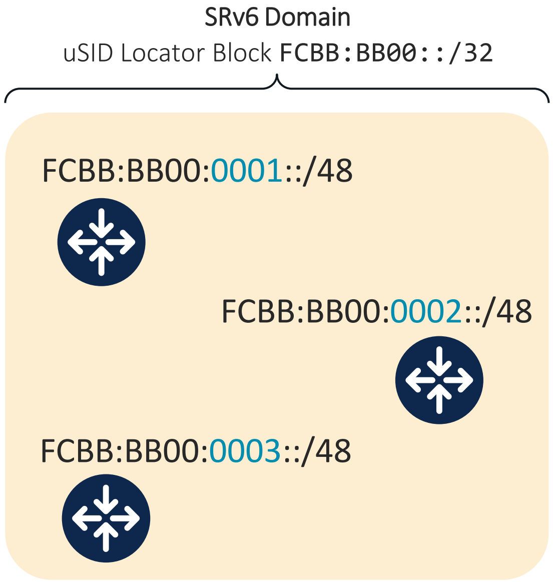
uSID – An Illustration

- For illustration, we will use:
  - uSID Locator Block length: 32 bits

`FCBB:BB00::/32` with **B** a nibble value picked by operator

  - uSID length (Locator Node ID / Function ID): 16 bits

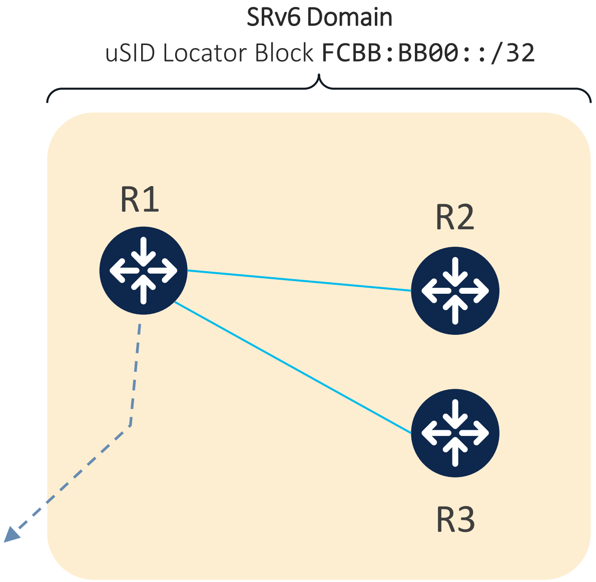
`FCBB:BB00:XYWZ::/48` with **XYWZ** variable nibbles



- A uSID `FCBB:BB00:XYWZ::/48` is said to be allocated from its block (`FCBB:BB00::/32`)

uSID – An Illustration (cont.)

- @ Node R1
- Global uSID
  - Locator Node ID - 0x0001
- Local uSIDs
  - Function ID 0xE000 – cross-connect to L3 neighbor R2
  - Function ID 0xE001 – cross-connect to L3 neighbor R3

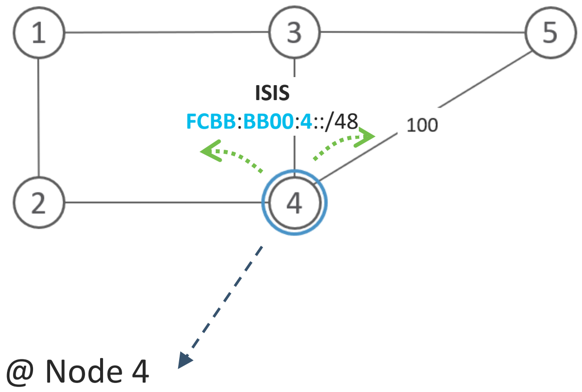


uSIDs: FCBB:BB00:0001::/48
FCBB:BB00:0001:E000::/64
FCBB:BB00:0001:E001::/64

SRv6 uSID Segment Locators

Segment Locator

- A parent node advertises its SRv6 Locator(s) as an IPv6 route(s) in the routing protocol
- From a routing perspective, the Locator summarizes all underlaying segments associated with it
- The rest of the network follows this route for any packets destined to the parent node. What the parent node does with the packet, according to the function, is purely a local behavior



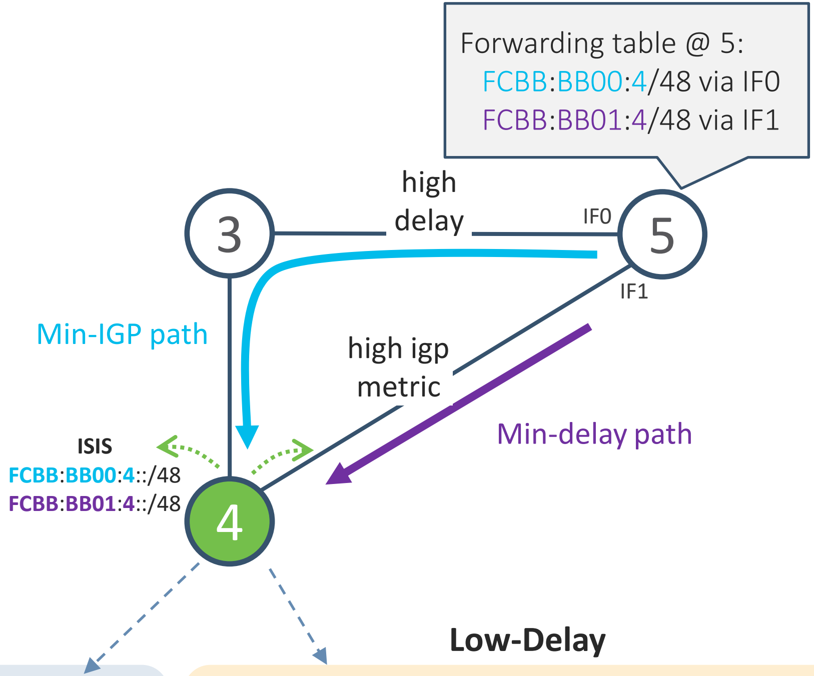
Locator Block: FCBB:BB00::/32

Locator: FCBB:BB00:4::/48

Functions:
FCBB:BB00:4:E000::/64
FCBB:BB00:4:E001::/64
:
:
FCBB:BB00:4:E007::/64

Segment Locator

- A node can have multiple locators
- Each Locator represents a routing policy to a node; e.g.:
  - low-cost slice locator
  - low-delay slice locator
- Locators can be assigned from one or multiple (recommended) SID locator blocks



Low-Cost

Locator Block: FCBB:BB00::/32

Locator 1: FCBB:BB00:4::/48

Functions:
FCBB:BB00:4:E000::/64
FCBB:BB00:4:E001::/64
:
:
FCBB:BB00:4:E007::/64

Low-Delay

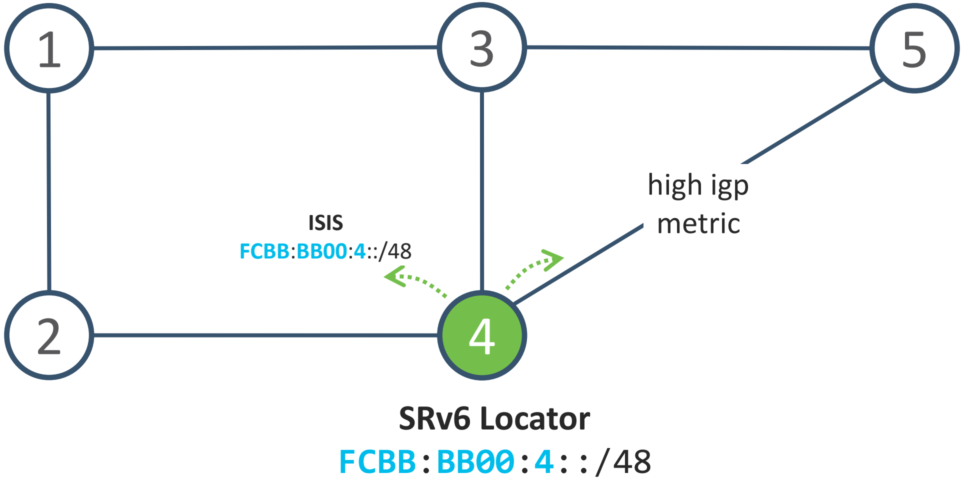
Locator Block: FCBB:BB01::/32

Locator 2: FCBB:BB01:4::/48

Functions:
FCBB:BB01:4:F00A::/64
FCBB:BB01:4:F00B::/64
:
:
FCBB:BB01:4:F010::/64

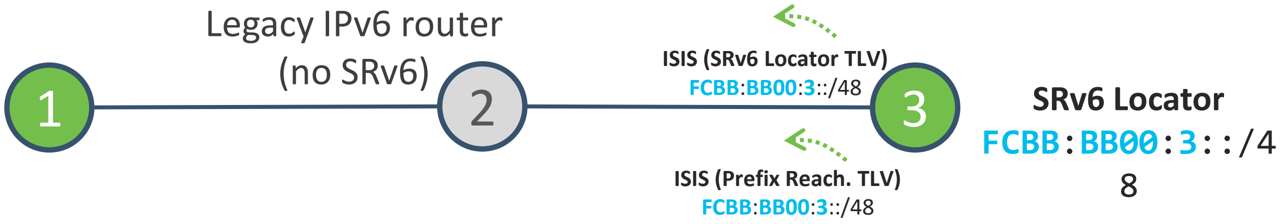
Segment Locator

- Advertising the locator as an IPv6 route in the routing protocol provides reachability to all the segments instantiated from that locator
- For example, all the segments of Node 4 that should be routed along the low-cost path to Node 4 are instantiated from locator `FCBB:BB00:4::/48`



Segment Locator Reachability – ISIS

- ISIS advertises a locator in both the SRv6 Locator TLV and in the Prefix Reachability TLV
  - When receiving both advertisements, SRv6 capable routers prefer Locator's Prefix Reachability TLV advertisement when installing entries in the FIB
- Locators MUST be advertised in the SRv6 Locator TLV
  - Forwarding entries for the locators advertised in the SRv6 Locator TLV MUST be installed in the forwarding plane of receiving SRv6 capable routers when the associated topology/algorithm is supported by the receiving node
- Locators associated with algorithm 0 and 1 (for all supported topologies) SHOULD also be advertised in a Prefix Reachability TLV (236 or 237)
  - So that legacy routers (i.e., routers which do not support SRv6) will install a forwarding entry for algorithm 0 and 1 SRv6 traffic



Segment Locator Reachability – ISIS

- Locators associated with Flexible Algorithms SHOULD NOT be advertised in Prefix Reachability TLVs (236 or 237). Advertising the Flexible Algorithm locator in regular Prefix Reachability TLV (236 or 237) would make the forwarding for it to follow algo 0 path

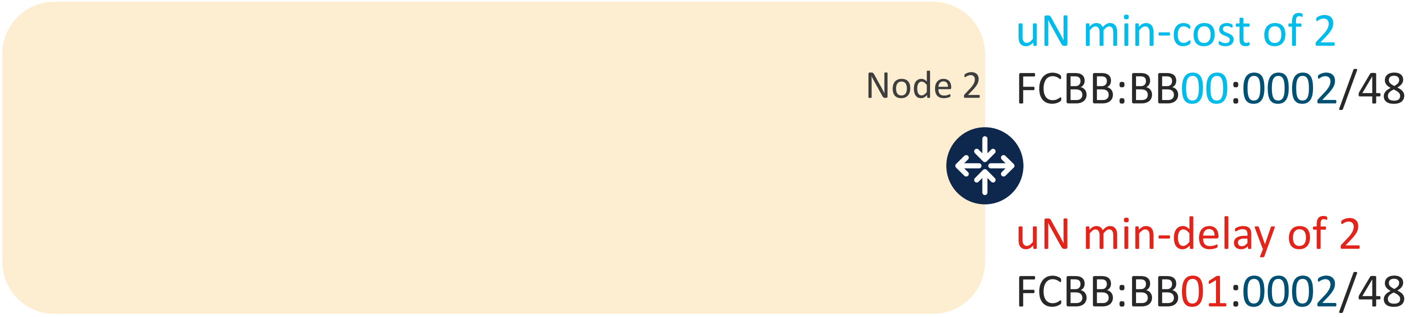
SRv6 uSID Advertisement

SR Domain

Min-Cost Slice:	FCBB:BB00/32
Min-Delay Slice:	FCBB:BB01/32

- Each slice is assigned a /32 uSID Locator Block
- Slices realized with user-defined Flex-Algo instances (e.g., FA 128 min-delay)

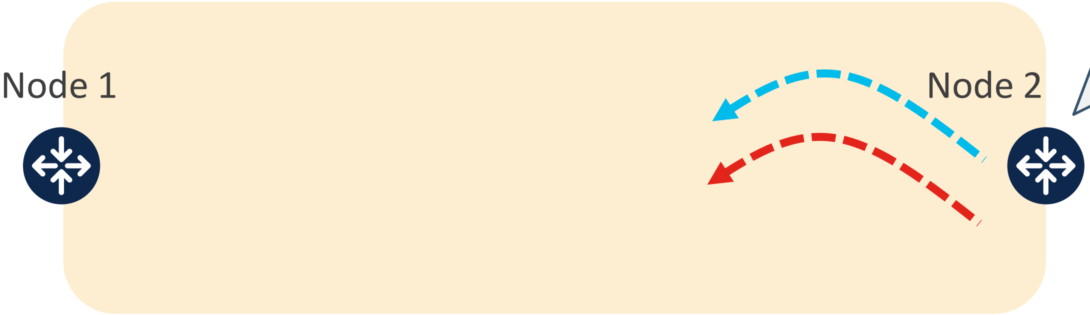
SR Node



- A node gets a Shortest-Path Endpoint uSID (uN) from each slice
- A uN is a /48 off the /32 of the related slice
- Classic Prefix-Based Routing (CIDR)

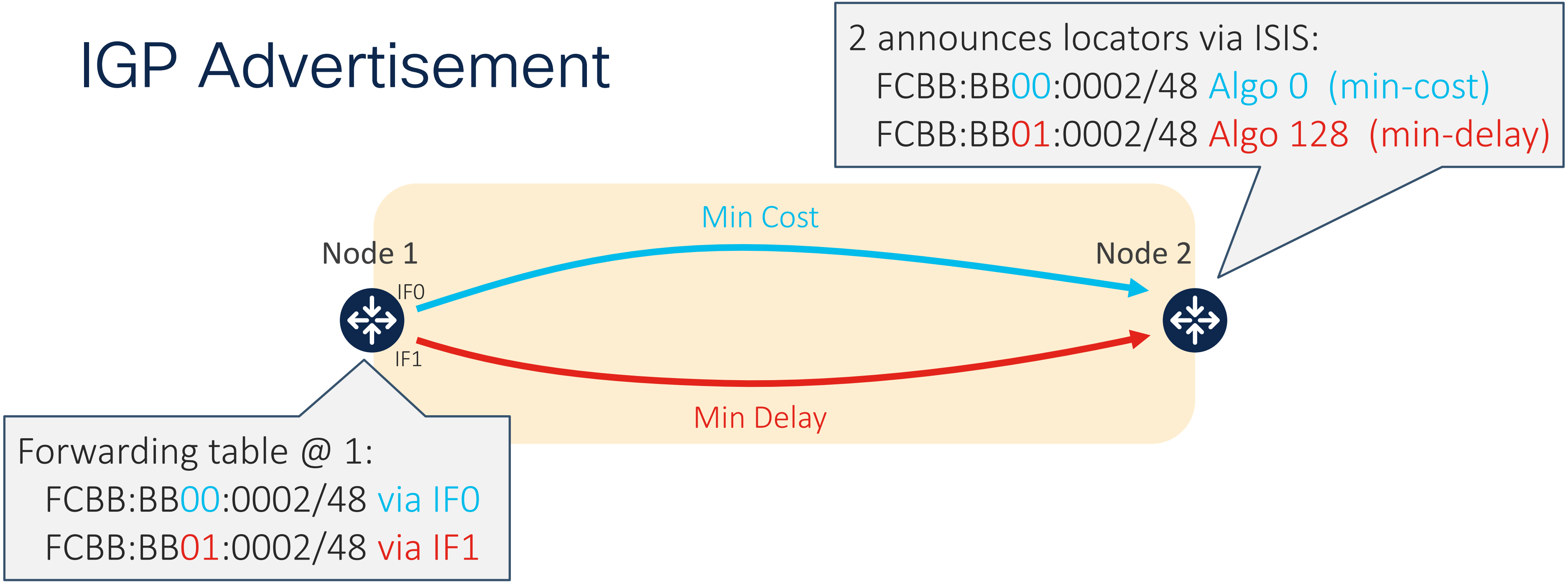
IGP Advertisement

2 announces locators via ISIS:
FCBB:BB00:0002/48 Algo 0 (min-cost)
FCBB:BB01:0002/48 Algo 128 (min-delay)



- An SRv6 SID is said to be routed if its SID belongs to an IPv6 prefix advertised via a routing protocol. An SRv6 SID that does not fulfill this condition is non-routed

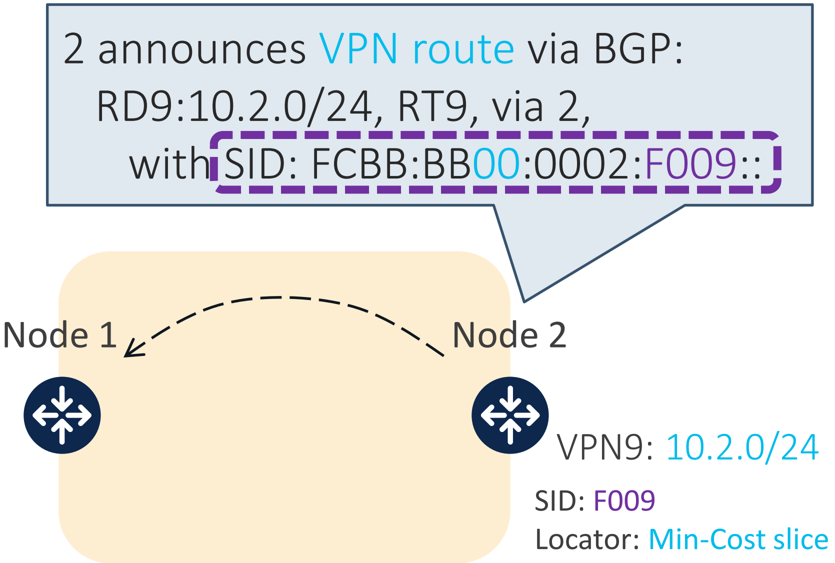
IGP Advertisement



- Classic IP Routing
  - Flex-Algo based routing to a /48

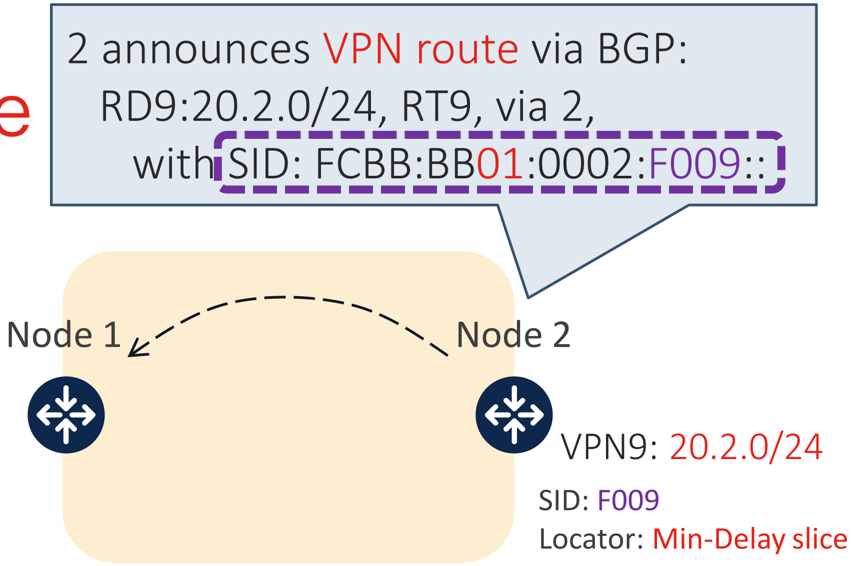
BGP Advertisement

- Intuitive uSID program:
  - Within the Min-Cost Slice (FCBB:BB00)
  - Follow the shortest-path to 2 (0002)
  - Execute VPN9 Decaps at 2 (F009)
- Seamless Deployment
  - Any transit node (SRv6 capable or not) routes on a classic /48
- Hardware Efficiency
  - Egress PE 2 processes multiple uSIDs with a single /64 lookup
  - FCBB:BB00:0002:F009/64



BGP Advertisement per Slice

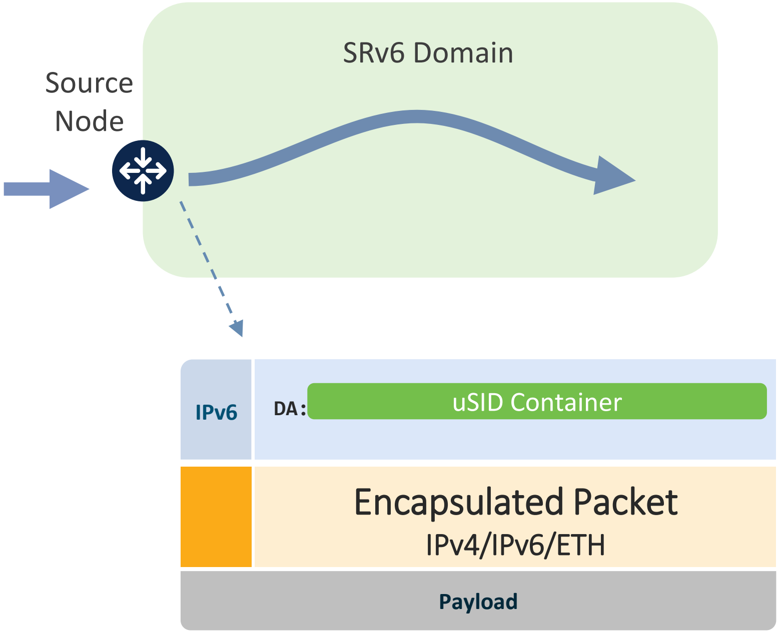
- Intuitive uSID program:
  - Within the Min-Delay Slice (FCBB:BB01)
  - Follow the shortest-path to 2 (0002)
  - Execute VPN9 Decaps at 2 (F009)
- Hardware Efficiency
  - Egress PE 2 processes multiple uSIDs with a single /64 lookup
  - FCBB:BB01:0002:F009/64



SRv6 uSID Processing and Forwarding

uSID Processing

- An uSID Container can be encoded in the destination address (DA) field of an IPv6 header or at any position in the Segment List of a Segment Routing Header (SRH)

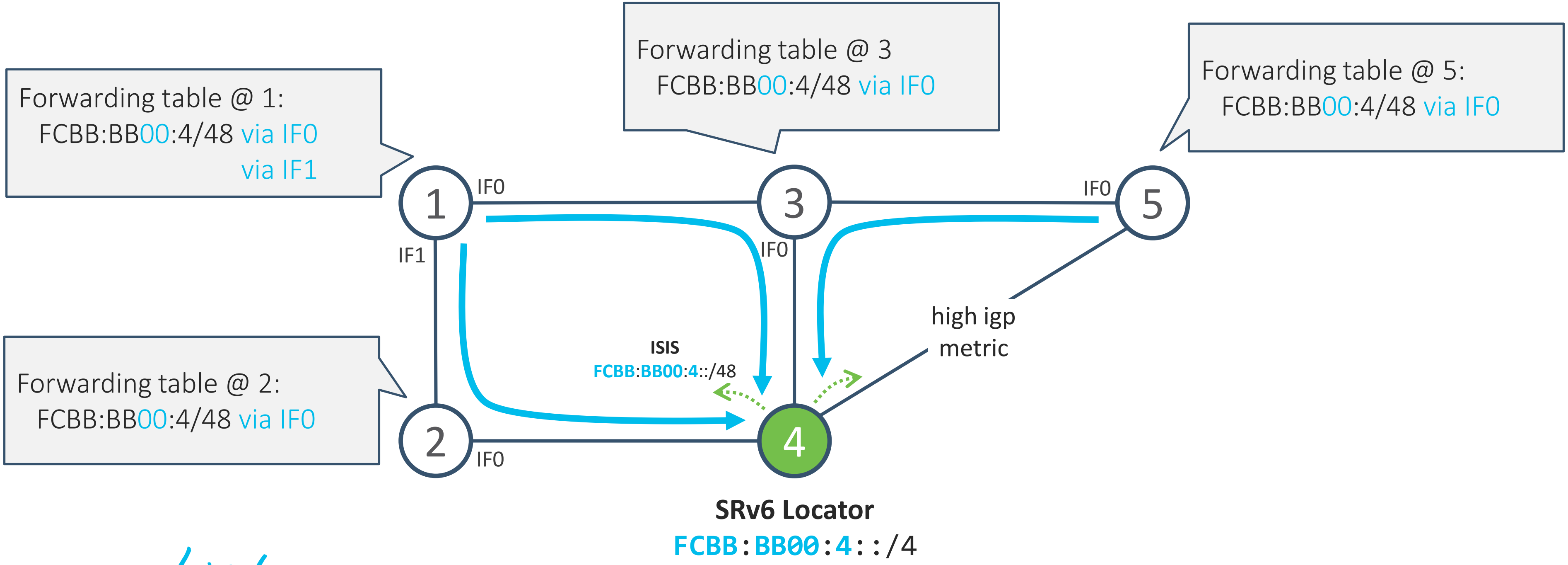


uSID Processing

- New flavors of the base SRv6 endpoint behaviors are used to decode uSID's compressed Segment List encoding
- These behaviors are based on the "Shift and Forward" operation

uSID Forwarding

- When an SRv6 SID is in the destination address field of an IPv6 header of a packet, the packet is routed through transit nodes in an IPv6 network based on its IPv6 address



uSID Processing – Legacy node

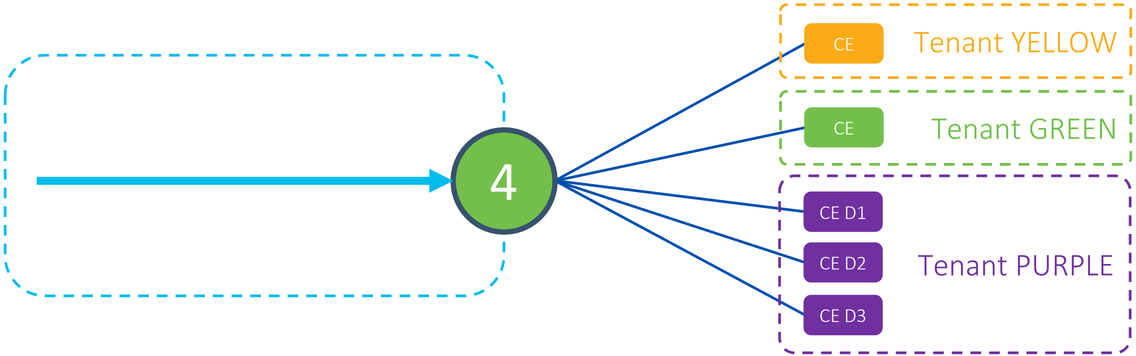
- When legacy IPv6 router (non-SRv6-capable) node receives an IPv6 packet, it performs a longest prefix match lookup on the packet's destination address. This lookup can return any of the following:
  - A FIB entry that represents a local interface
  - A FIB entry that represents a non-local route
  - No Match

uSID Processing – SRv6-capable node

- When an SRv6-capable node receives an IPv6 packet, it performs a longest prefix match lookup on the packet's destination address. This lookup can return any of the following:
  - A FIB entry that represents a locally instantiated SRv6 SID
  - A FIB entry that represents a local interface, not locally instantiated as an SRv6 SID
  - A FIB entry that represents a non-local route
  - No Match

uSID Processing – SRv6-capable node

- A SR segment endpoint node creates FIB entries for its local SIDs
- A single longest prefix match used resolve across services at different scales



FIB entries@4:

fcbb:bb00:4::/48	uNode(4)
fcbb:bb00:4:f100::/64	uDT4 (L3VPN-YELLOW)
fcbb:bb00:4:f200::/64	uDT4 (L3VPN-GREEN)
fcbb:bb00:4:fff0:D1::/80	uDX2 (L2VPN-PW-PURPLE-D1)
fcbb:bb00:4:fff0:D2::/80	uDX2 (L2VPN-PW-PURPLE-D2)
fcbb:bb00:4:fff0:D3::/80	uDX2 (L2VPN-PW-PURPLE-D3)

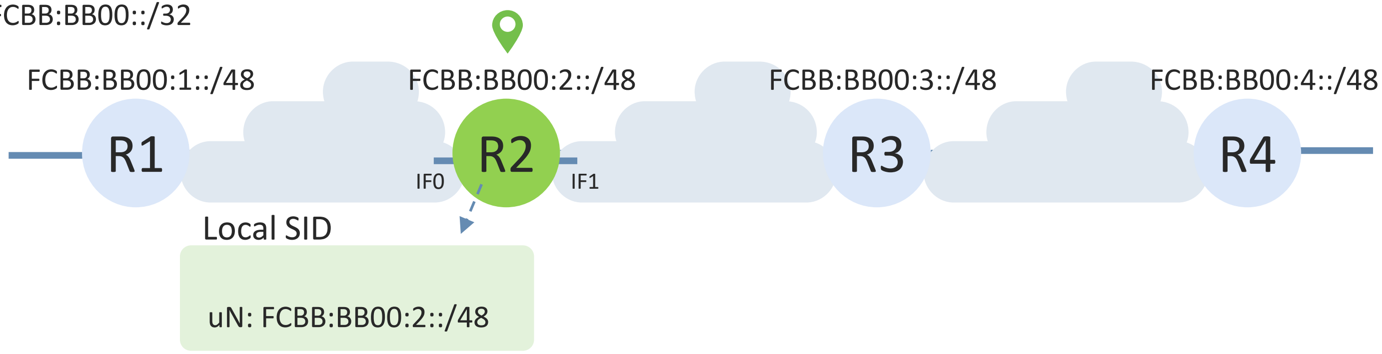
/48: A few unicast/anycast personalities @ PE4

/64: A few hundreds/thousands of VRF's @ PE4

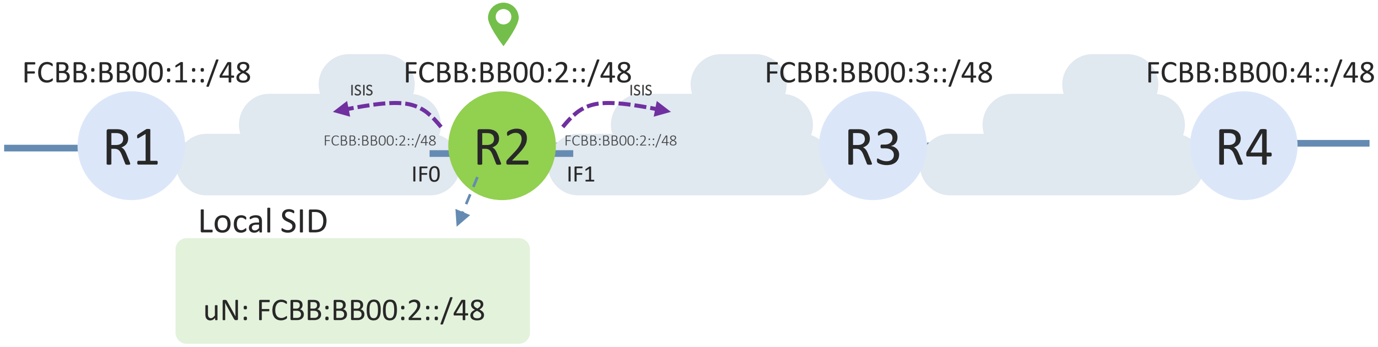
/80: 100k of PW's @ PE4

Shift and Forward Operation

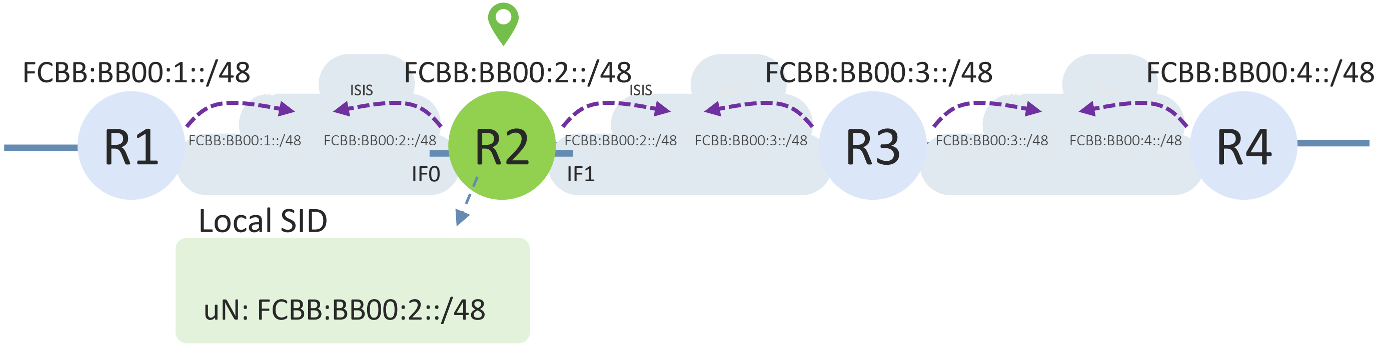
Locator Block: FCBB:BB00::/32



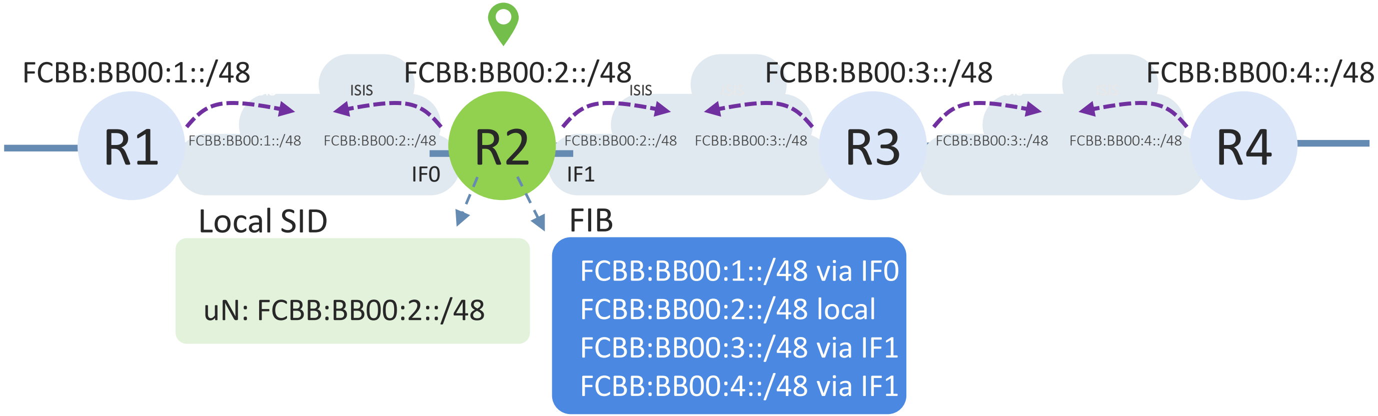
Shift and Forward Operation



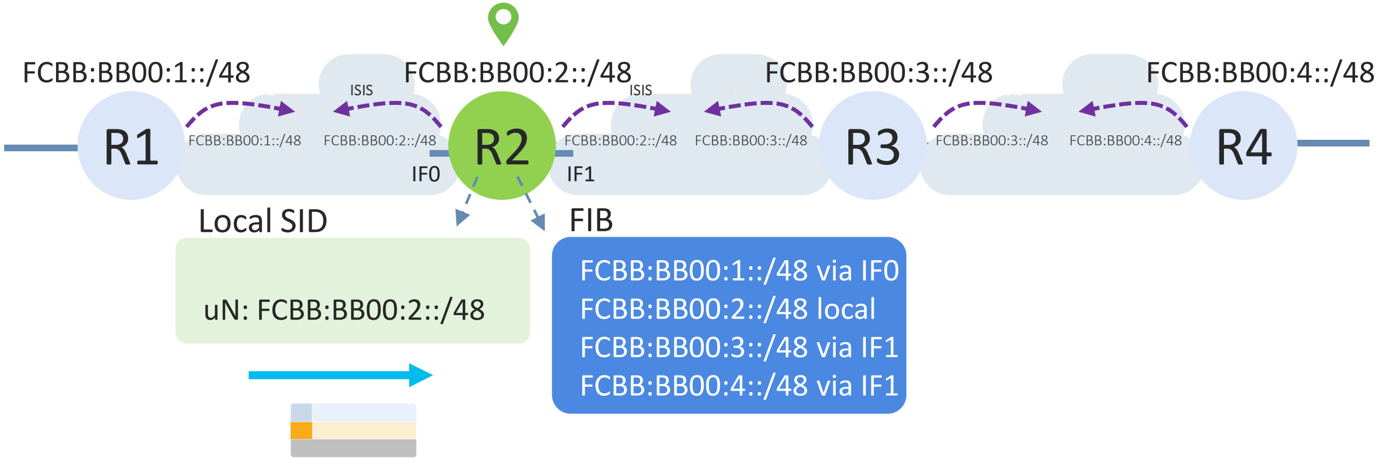
Shift and Forward Operation



Shift and Forward Operation



Shift and Forward Operation



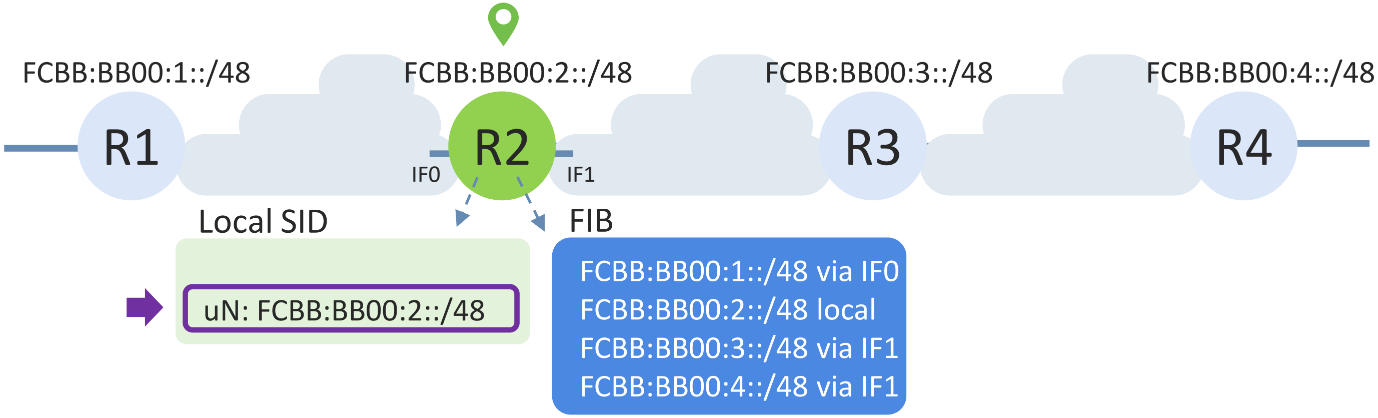
EXAMPLE

Incoming DA:

FCBB:BB00:0002:0003:0004:0000:0000:0000

IPv6	SA: FCBB:BB00:1::1 **DA: FCBB:BB00:2:3:4::**
IPv4	**Payload**

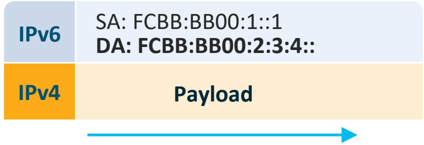
Shift and Forward Operation



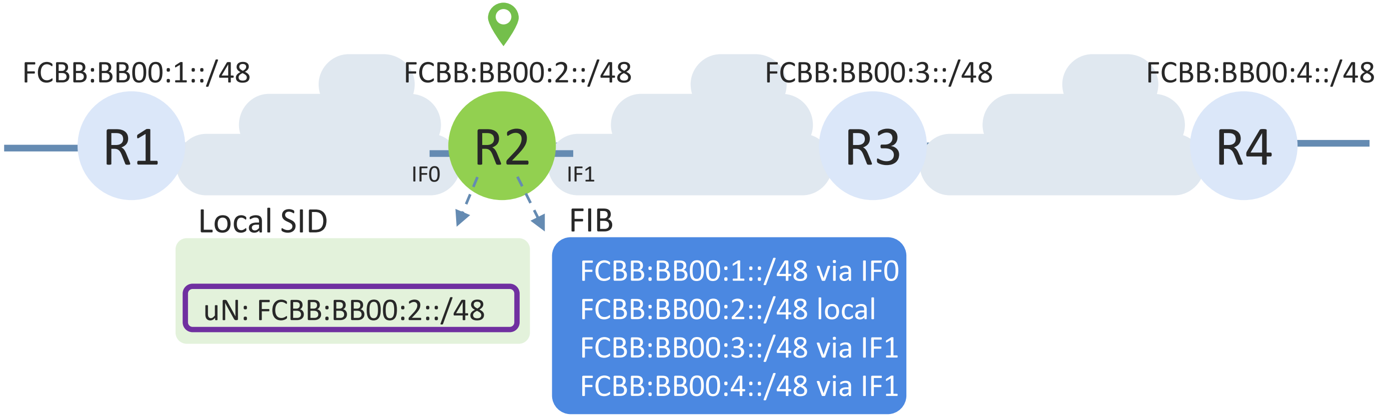
A FIB longest prefix match lookup matches local SID FCBB:BB00:2::/48 → apply SRv6 uN instruction:

Incoming DA:

FCBB:BB00:0002 0003:0004:0000:0000:0000



Shift and Forward Operation



A FIB longest prefix match lookup matches local SID FCBB:BB00:2::/48 → apply SRv6 uN instruction:

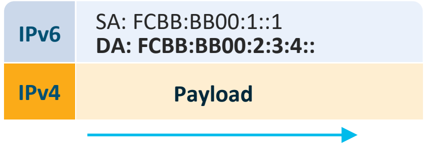
B Shift micro-program by one micro-instruction

Incoming DA:

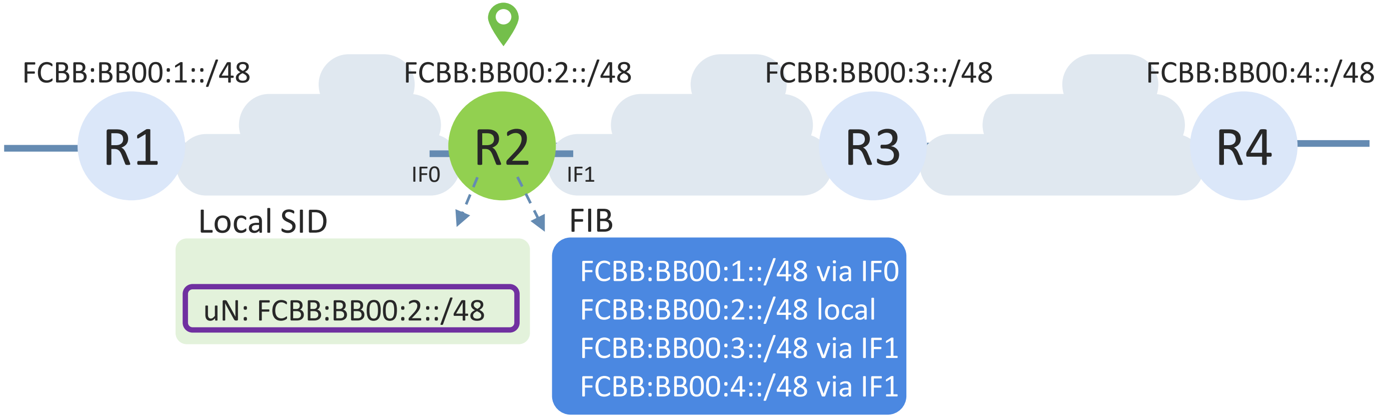
FCBB:BB00:0002 0003:0004:0000:0000:0000 **[SHIFT << 16]**

Outgoing DA:

FCBB:BB00:0003:0004:0000:0000:0000



Shift and Forward Operation



A FIB longest prefix match lookup matches local SID FCBB:BB00:2::/48 → apply SRv6 uN instruction:

B Shift micro-program by one micro-instruction

C Set last micro-instruction to "End-of-Container"

Incoming DA:

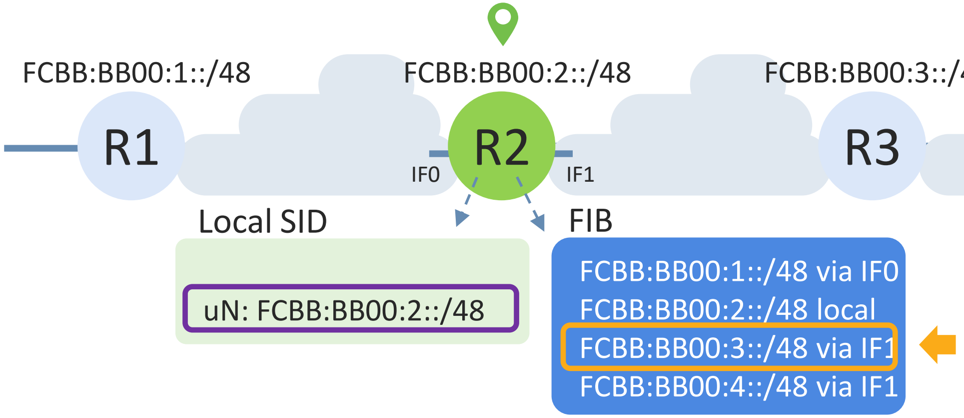
FCBB:BB00:0002 0003:0004:0000:0000:0000 **[SHIFT << 16]**

Outgoing DA:

FCBB:BB00:0003:0004:0000:0000:0000:0000

IPv6	SA: FCBB:BB00:1::1 **DA: FCBB:BB00:2:3:4::**
IPv4	**Payload**

Shift and Forward Operation




```
A FIB longest prefix match lookup matches local SID FCBB:BB00:2::/48 → apply SRv6 uN instruction:
    B Shift micro-program by one micro-instruction
    C Set last micro-instruction to “End-of-Container”
    D Lookup the updated DA and forward
```

Incoming DA:

FCBB:BB00:0002 0003:0004:0000:0000:0000 **[SHIFT << 16]**

Outgoing DA:

FCBB:BB00:0003:0004:0000:0000:0000:0000