RIPE NCC
RIPE NETWORK COORDINATION CENTER

Introduction to the Segment Routing for IPv6 (SRv6)

Alex Semenyaka | July 2026 | RIPE Community Day, Odesa

Challenges

Not all network elements are equal…

- Different types of traffic
- Different application requirements
- Different types of channels

Pre-defined routing

Why?

- Necessity of the simple and explicit Traffic Engineering (TE)
  - Better capacity planning
  - Rich routing metrics
  - Eliminating local route fluctuations during recalculations
- Need to better partition and structure the network
  - Robust FRR
- Effortless management and diagnostics with explicit path control

The Dark Ages

Taking swings at the pre-defined routing

- IPv4: Option LSRR and SSRR (Loose and Strict)
  - Source routing
  - Existed from the beginning of IPv4
  - Had the security and productivity issues
  - Deprecation started at early 2000s

The Dark Ages

Taking swings at the pre-defined routing

- IPv6: Routing Header type 0 (RH0)
  - Source routing
  - Existed from the beginning of IPv6
  - Had the security issues
  - Deprecated in 2007

The Dark Ages

Taking swings at the pre-defined routing

- Path Computational Elements Approach
  - Predefined routing
  - Invented in 2006
  - Moving the Constrained Shortest Path First (CSPF) onto the external controller with the unified Traffic Engineering Database
    - And then this controller can instruct routers
  - Needs a source of the topology information (like IGP or BGP-LS)
  - Limited usage (MPLS RSVP-TE and GMPLS)

The Dark Ages

Taking swings at the pre-defined routing

- OpenFlow SDN
  - Routing defined by a controller with the full visibility and knowledge of everything
  - Invented in 2008
  - Complete physical separation of the control plane from the data plane
    - From posh and smart branded boxes to cheap stupid white boxes
  - It had been written off by the industry by 2018 due to internal conflicts
    - The need for a massive TCAM, and NP instead of ASIC
    - Control Plane latency and the responsiveness issue
    - The complexity of versioning multi-level tables

The Renaissance

MPLS-based Segment Routing

- The very first successful and universal [enough] solution
- Invented in 2013
- Moved away from the micromanagement
  - In favor of a combination of distributed and pre-defined decision-making
- Moved beyond the traditional routing paradigm
  - And redefined the meaning of labels in MPLS

The Modern Era

Segment Routing in IPv6

- Adaptation of core concepts from SR/MPLS
- Simplified deployment
- Easy adaptation
  - Not without drawbacks though

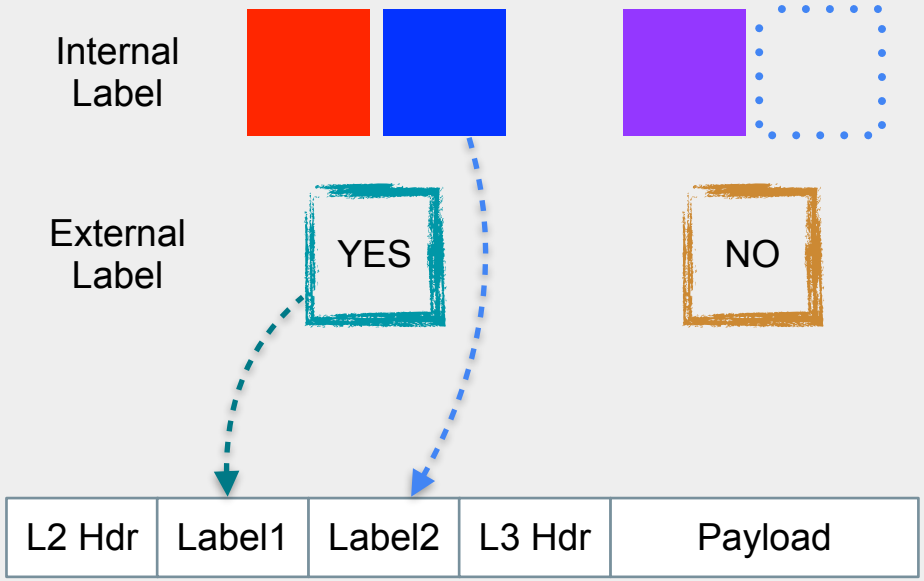
A short recap

- The **simple concept** of hierarchical labelling of IP packets
- At each level, a single label groups together all “equivalent” packets
  - Forming classes of equivalence (completely arbitrary)
  - The logic behind equivalence is arbitrary and is determined by the network architect
- Labels has the local meaning
- **Levels are stacked**
- A router works with the topmost label
- Label switching instead of routing

- And then we pile an **over-engineered control plane (LDP, RSVP-TE)** onto this

A [twisted] example: two levels

- The deepest label: the color of an IP packet
  - Red
  - Blue
  - Purple
  - Transparent (oops)
- The topmost level: the necessity
  - At the level of the meaning of life
  - Better not to ever meet



Segment Routing

The initial concepts

- Labels are no longer local
- Labels are not overwritten
- A label represents a network entity: **a segment**
  - **Label value = Segment ID (SID)**
- These entities (segments) may be of different kinds
- And a label stack is the sequence in which these segments pass through

What on earth are these *entities*?

- The simplest ones are **nodes** and **links**
  - For example, a SID sequence “NodeA, NodeB, LinkX” means “go from Node A to Node B by whichever route they prefer, and then to Node C take exactly Link X”
  - If we use only loopback addresses, the basic structure becomes very similar to RH0
- But segments can also be more complex
  - **Any operation** that could be performed on an IP packet on the given router can be referred to as a segment
    - For example, directing the packet to the NAT64 box, into a tunnel, or to the scrubbing center

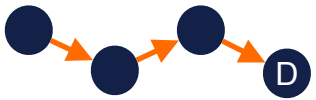
Programmatic network

- Each router can work with different segments
  - These capabilities should be known in advance
- Then, selecting a **segment** means choosing the instruction (**functions**) what is to be done with the packet at that stage
  - Therefore, **a sequence of segments is a program**
- **Function** is what is encoded in SID. **Behaviour** is the local action that the node has associated with this function.

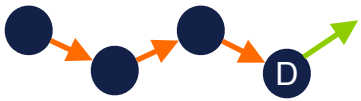
Segment Routing: functions

Some examples

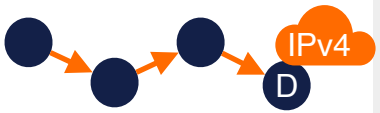
End Instruction
the shortest path to a given node



End.X Instruction
the shortest path to a given node followed by a forwarding over a specific link



End.DT4
the shortest-path to a selected egress node followed by a IPv4-VPN table lookup



End.DT6
the shortest-path to a selected egress node followed by a IPv6-VPN table lookup

Segment Routing concepts

SR Domain

- Segment Routing Global Block (SRGB): the range of labels reserved for SR-MPLS
  - Other labels are handled outside the SR-MPLS logic
  - Can be combined with the traditional MPLS
- **The part of the infrastructure that supports Segment Routing is called an SR Domain**

Segment Routing implementation

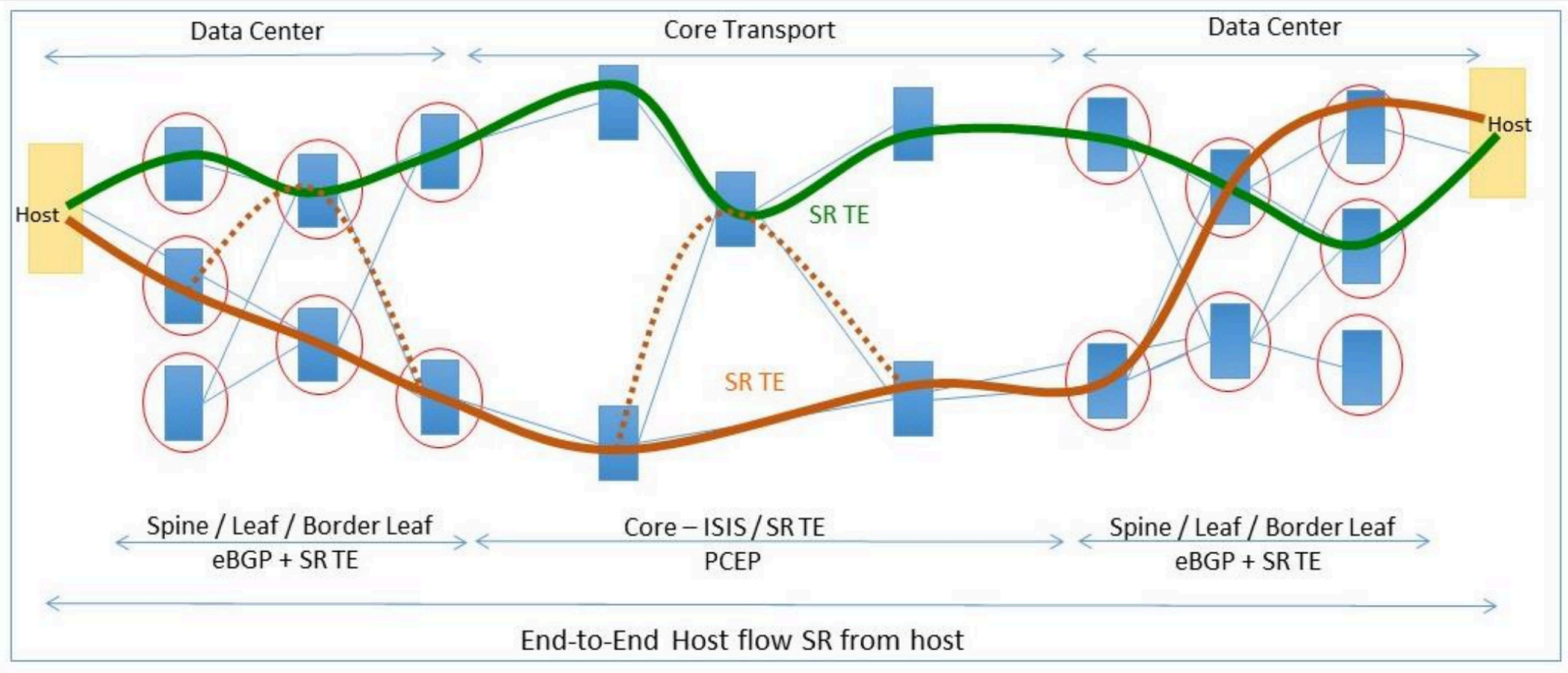
The missing part: sources of SIDs?

Segment Routing implementation

The missing part: sources of SIDs

- Statically pre-configured SIDs
- IGP with extensions (“I have a loopback address 10.0.0.1 and it is SID=10001”)
  - IS-IS
  - OSPF
- SDN Controller
  - Still IGP or BGP-LS
  - PCEP/NETCONF etc
- …or any mix of them.

Example



by Bell Canada (2016), via https://www.segment-routing.net/images/lightreading_report.pdf

SR/MPLS Issues

Why the deployment could be challenging

- You need to have an MPLS infrastructure in place everywhere
- All your equipment must support SR/MPLS
- ASICs do not support deep label stacks
  - Scalability would demand some quirks and tweaks
  - ECMP would be broken

Ready-to-be-used

- RH0 is deprecated, but we still have a Routing Extension Header
- It is large
- It is easy to use
- And we always will have a backup
  - Default routing based on the Basic Header

Translate MPLS into IPv6

- We introduce Segment Routing Header (SRH)
  - It has Type 4 (RH4)
- The SIDs are coded now by 128-bits IDs
  - They are valid IPv6 address inside your numeration but with additional properties

128-bit SRv6 SID

Locator: the route to the node performing the function

Function: the forwarding behaviour to be executed by the node

Arguments (optional): if the function needs (not shown in the drawing)

SRH Structure

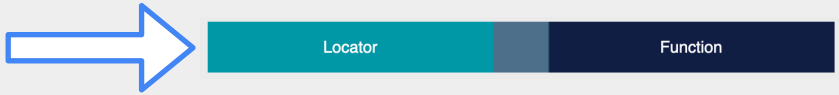
SRH Base (first 8 octets)			
Bits 0-7	**Bits 8-15**	**Bits 16-23**	**Bits 24-31**
Next Header 8 bits	Hdr Ext Len 8 bits	Routing Type 8 bits	**Segments Left** 8 bits
Last Entry 8 bits	Flags 8 bits	Tag 16 bits	

Segment List area
Segment List[0] (128-bit IPv6 address)
...
Segment List[n] (128-bit IPv6 address)

Optional TLV area (variable length)
TLVs start after Segment List[n]

Routing Type:
- 4 for SRH

Flags:
- O-flag (0x4): OAM support



Generic SRH TLV format		
Bits 0-7	**Bits 8-15**	**Bits 16...**
Type 8 bits	Length 8 bits	Variable-length data
Type MSB: 0 = immutable TLV data, 1 = mutable TLV data		

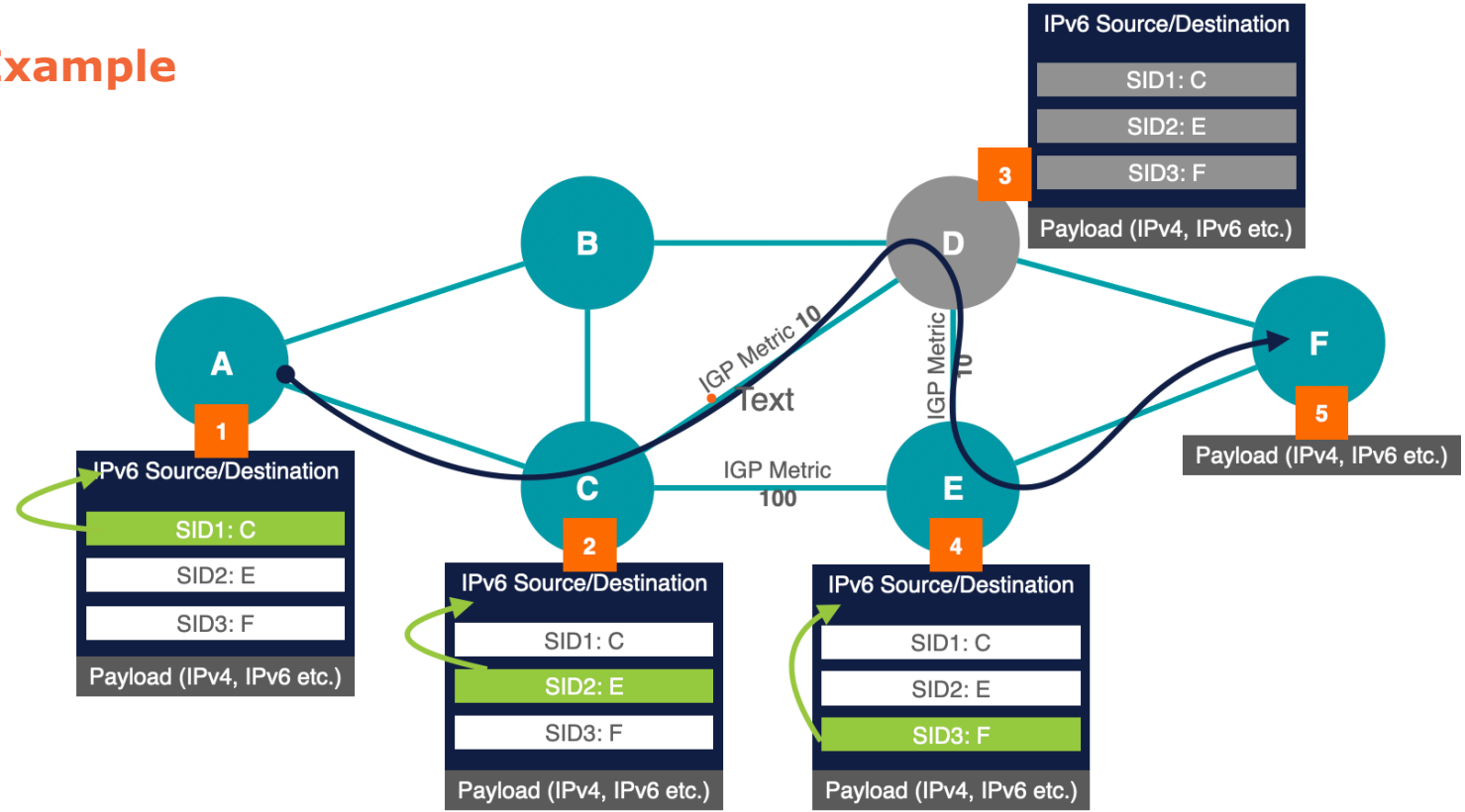
SRv6 vs SR/MPLS

Differences

- We will use IPv6 addresses instead of MPLS labels (obviously)
- We copy the next SID into the Destination IP field of the basic IPv6 header
- We do not pop the current SID from SRH, just move the pointer (the Segments Left field)
  - If the pointer=0, the SR domain is over
    - But we still *can keep* SRH in place
    - Or *can drop* it off if we want
- Network functions can have arguments (optional)
- **Intermediate routers do not need any SRv6 support**
  - Non-SRv6 node forwards the packet as usual, based on the current Dst IP and LPM

SRv6

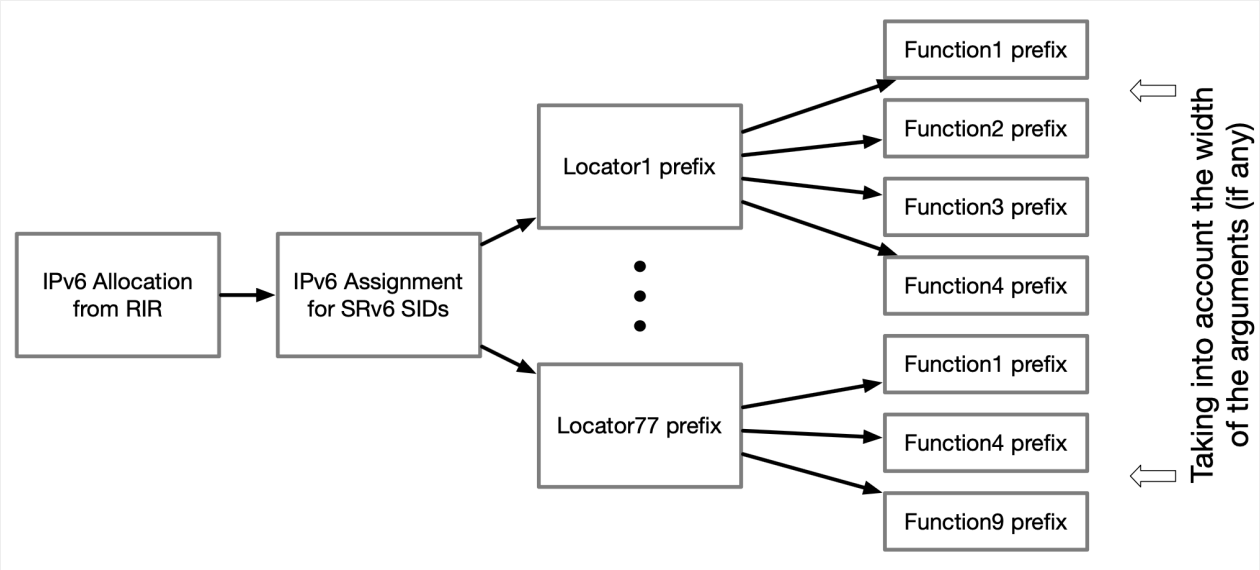
Example



SIDs in SRv6

Address plan

- We have no special global range for SIDs, they are **valid IPv6 addresses**
  - And they also can be **aggregated**
- But we have to allocate a special prefix in our allocation



SRv6 routing decisions

Who and how produces SRH?

- Headend – the point where SRHs are generated
  - Entry to the SRv6 domain
- Headend creates an SRH with the sequence of SID based on SR Policy
  - **Explicit/static** — an exact segment list is specified
  - **Dynamic** — the segment list is calculated based on constraints
  - **Composite** — several policies are combined
- SR Policy can produce several path candidates
  - SID list corresponds to the best one
    - (define “the best”, of course)
- A trivial policy is a usual Shortest Path Tree (SPT)

Adding SRH into the packet

How do we insert it?

- **Usually we don't**; instead, **we encapsulate** the original packet into the new one, adding both SRH and a basic IPv6 header (T.Encap / H.Encap)
  - Insertion is prohibited on the transit nodes (T.Insert)
  - Insertion is theoretically possible if the headend is the end host (H.Insert)
    - But in practice it is not supported by operating systems
- And that's why the support of O-bit in SRH is important
  - The good new is that all major manufacturers support it properly

Static SR Policy

Linux example

```
ip -6 route add <dst> encap seg6 mode encap segs <SID-list> dev <iface>
```

Dynamic SR Policy

Simple architecture: no SRv6 controller

- IGP distributes locators and SRv6 topological information
  - The network knows how to find the places where SIDs live
  - The IGP is primarily responsible for underlay reachability and topology
- It is BGP that typically carries service SIDs and may carry policies
  - Service SIDs live in the BGP world, particularly when it comes to VPNs and such
- The headend calculates candidate paths

IGP Support

OSPFv3 extensions for SRv6

- OSPFv3 simply takes its standard topology packets and embeds additional fields within them
- Router spreads the SRv6 Locator TLV across the area
  - Basic Availability
- Node-specific functions in SRv6 End SID sub-TLV
  - An additional sub-TLV is embedded directly within the locator announcement
- Adjacent interface functions in SRv6 End.X SID sub-TLV
  - The router adds the proper SID to the description of its link

Dynamic SR Policy

SRv6 controller

- In the SRv6 world the controller does not enable SRv6, but makes orchestration and path computation [more] scalable
- So, the controller needs to be able to see the network topology and SRv6 attributes
  - Typically, via BGP-LS
- The controller then **calculates candidate paths** that meet the required constraints and **forwards them** to the headends
  - For example, via PCEP
    - (Now it is exactly where it needs to be)
- The controller does not replace underlay routing, but operates **above** it
- The controller is responsible for determining **which policy** to select and **for which traffic**, and sometimes for **recalculating** it when the network changes

Fast Re-Route

TI-LFA

- SRv6-capable nodes can track the outages
- Also it can calculate the alternative paths to destinations
- Such SRv6-node is called Point of Local Repair (**PLR**)
- If a failure occurs, the PLR immediately begins to encapsulate transit traffic into a pre-configured Segment List (repair path)
    - So, it is forcing it to bypass the failed section without waiting for a response from the rest of the network.

- This way of doing fast re-routing is called Topology Independent Loop-Free Alternate (**TI-LFA**)

Security

SRH is too similar to RH0, we are doomed!

- It is a policy that makes a technology from the set of protocols and header formats
- And it is more than true for SRv6

Security considerations:

- SRv6 operates within the **trust boundary**
- The SID address space **must not be accessible from outside**
  - By internal SRv6-capable nodes as well
- **The control plane** is part of the security perimeter
  - Theoretically, also HMAC signature for SRH were defined
  - Currently is supported only by Linux

Operational aspects

Which constrains do we have? Which bonuses?

- SR-capable router must ignore traffic policy even on non-SRv6 interfaces
  - Easy to create a security breach!
- MTU
  - SRH is quite large
- MSD, Maximum SID Depth(s)
  - ASICs cannot scan SRH too deep
    - At least, at the moment
- OAM
  - Advanced telemetry with O-bit

Questions?
Comments?
Applauses?

asemenyaka@ripe.net