SRv6
Network Programming
Introduction
Segment Routing

• Source Routing
  - the topological and service (NFV) path is encoded in packet header

• Scalability
  - the network fabric does not hold any per-flow state for TE or NFV

• Simplicity
  - automation: TILFA sub-50msec FRR
  - protocol elimination: LDP, RSVP-TE, VxLAN, NSH, GTP, ...

• End-to-End
  - DC, Metro, WAN
Two dataplane instantiations

**MPLS**
- leverage the mature MPLS HW with only SW upgrade
- 1 segment = 1 label
- a segment list = a label stack

**IPv6**
- leverages RFC8200 provision for source routing extension header
- 1 segment = 1 address
- a segment list = an address list in the SRH
IPv6 provides reachability
IPv4 limitations & work-arounds

- Limited address space
- No engineered Load Balancing
- No VPN
- No Traffic Engineering
- No Service Chaining
IPv4 limitations & work-arounds

- Limited address space → NAT
- No engineered Load Balancing → MPLS Entropy Label, VxLAN UDP
- No VPN → MPLS VPN’s, VxLAN
- No Traffic Engineering → RSVP-TE, SR-TE MPLS
- No Service Chaining → NSH

![Diagram showing layers of network protocols and work-arounds](image-url)
IPv4 limitations & work-arounds

- Limited address space → NAT
- No engineered Load Balancing → MPLS Entropy Label, VxLAN UDP
- No VPN → MPLS VPN’s, VxLAN
- No Traffic Engineering → RSVP-TE, SR-TE MPLS
- No Service Chaining → NSH

<table>
<thead>
<tr>
<th>Data (L5,L6 &amp; L7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Socket header (L4)</td>
</tr>
<tr>
<td>IPv4 header (L3)</td>
</tr>
<tr>
<td>Ethernet (L2)</td>
</tr>
</tbody>
</table>

Simplicity (back to the OSI model)
SRv6 unleashes IPv6 potential
SR for anything:
Network as a Computer
Network instruction

• 128-bit SRv6 SID
  - Locator: routed to the node performing the function
  - Function: any possible function
    either local to NPU or app in VM/Container
  - Flexible bit-length selection
Network Program

Next Segment

Locator 1 Function 1
Locator 2 Function 2
Locator 3 Function 3
Network Program

Next Segment

Locator 1  Function 1
Locator 2  Function 2
Locator 3  Function 3
Network Program in the Packet Header

IPv6 header

Segment Routing Header

IPv6 payload

Source Address

Active Segment

Locator 1
Function 1

Locator 1
Function 1

Locator 2
Function 2

Locator 3
Function 3

TCP, UDP, QUIC
Network Program in the Packet Header

IPv6 header

Source Address

Locator 2 Function 2

Segment Routing Header

Active Segment

Locator 1 Function 1

Locator 2 Function 2

Locator 3 Function 3

IPv6 payload

TCP, UDP, QUIC

© 2021 Cisco and/or its affiliates. All rights reserved.
Network Program in the Packet Header

IPv6 header

- Source Address
- Locator 3 Function 3

Segment Routing Header

- Active Segment
- Locator 1 Function 1
- Locator 2 Function 2
- Locator 3 Function 3

IPv6 payload

TCP, UDP, QUIC
### Group-Based Policy

![Diagram showing Group-Based Policy]

The diagram illustrates the components of Group-Based Policy, including the Metadata TLV and TAG segments.

<table>
<thead>
<tr>
<th>TAG</th>
<th>Segments Left</th>
<th>Locator 1</th>
<th>Function 1</th>
<th>Locator 2</th>
<th>Function 2</th>
<th>Locator 3</th>
<th>Function 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata TLV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Argument shared between functions

“Global” Argument

Metadata TLV

TAG
- Segments Left
- Locator 1: Function 1
- Locator 2: Function 2
- Locator 3: Function 3
## SRv6 Header

### IPv6 header

<table>
<thead>
<tr>
<th>TAG</th>
<th>Segments Left</th>
<th>Locator 1</th>
<th>Function 1</th>
<th>Locator 2</th>
<th>Function 2</th>
<th>Locator 3</th>
<th>Function 3</th>
</tr>
</thead>
</table>

### Metadata TLV

```
// Segment List[0] (128 bits IPv6 address)
...
// Segment List[n] (128 bits IPv6 address)
```

// Optional Type Length Value objects (variable) //
SRv6 for anything

<table>
<thead>
<tr>
<th>TAG</th>
<th>Segments Left</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Locator 1</td>
</tr>
<tr>
<td></td>
<td>Locator 2</td>
</tr>
<tr>
<td></td>
<td>Locator 3</td>
</tr>
</tbody>
</table>

Metadata TLV

Turing
SRv6 for anything

<table>
<thead>
<tr>
<th>TAG</th>
<th>Segments Left</th>
</tr>
</thead>
<tbody>
<tr>
<td>Locator 1</td>
<td>Function 1</td>
</tr>
<tr>
<td>Locator 2</td>
<td>Function 2</td>
</tr>
<tr>
<td>Locator 3</td>
<td>Function 3</td>
</tr>
</tbody>
</table>

**Optimized for HW processing**
e.g. Underlay & Tenant use-cases

**Optimized for SW processing**
e.g. NFV, Container, Micro-Service
SRv6 Domain

IPv6 enabled provider infrastructure
SR Domain
Encapsulation at the Domain ingress

- IPv4, IPv6 or L2 frame is encapsulated within the SR Domain
- Outer IPv6 header includes an SRH with the list of segments
SRH of the outer IPv6 encapsulation

• Domain acts as a giant computer
• The network program in the outer SRH is executed
Decapsulation at Domain Egress

- Egress PE removes the outer IPv6 header as the packet leaves the SR domain
End-to-End Integrity

- End-to-end integrity principle is strictly guaranteed
  - Inner packet is unmodified
  - Same as SR-MPLS (MPLS stack is replaced by IPv6 outer header and SRH)
End and End.X SID behaviors

• End – Default endpoint behavior
  - shortest-path to the SID’s endpoint
  - endpoint updates DA with next SID
  - endpoint forwards according to updated DA

• End.X – Endpoint with cross-connect
  - shortest-path to SID’s endpoint
  - endpoint updates DA with next SID
  - endpoint forwards to interface associated with SID
Endpoint behaviors illustration

SR: \( \langle B:0:0:4:1::, B:0:0:5:C6::, A:8:\rangle \)

- \( B:0:0:4:1:: \): shortest path to node 4
- \( B:0:0:5:C6:: \): shortest path to node 5, then cross-connect towards 6
- \( A:8:: \): regular IPv6 address of node 8