IETF 110 – Online
March 2021

draft-ali-spring-network-slicing-building-blocks-04.txt

Building blocks for Slicing in SR Network

Zafar Ali, Cisco Systems (zali@cisco.com) - Presenter
Clarence Filsfils, Cisco Systems (cfilsfil@cisco.com)
Pablo Camarillo, Cisco Systems (pcamaril@cisco.com)
Francois Clad, Cisco Systems (fclad@cisco.com)
Daniel Voyer, Bell Canada (daniel.voyer@bell.ca)
Satoru Matsushima, Softbank, (satoru.matsushima@g.softbank.co.jp)

Scope of the Draft

- Informational Draft
- Lists essential building blocks needed for network slicing
- How these building blocks needs to work together seamlessly
- Goals
  - Scaling
  - Incremental deployments

History of the Draft

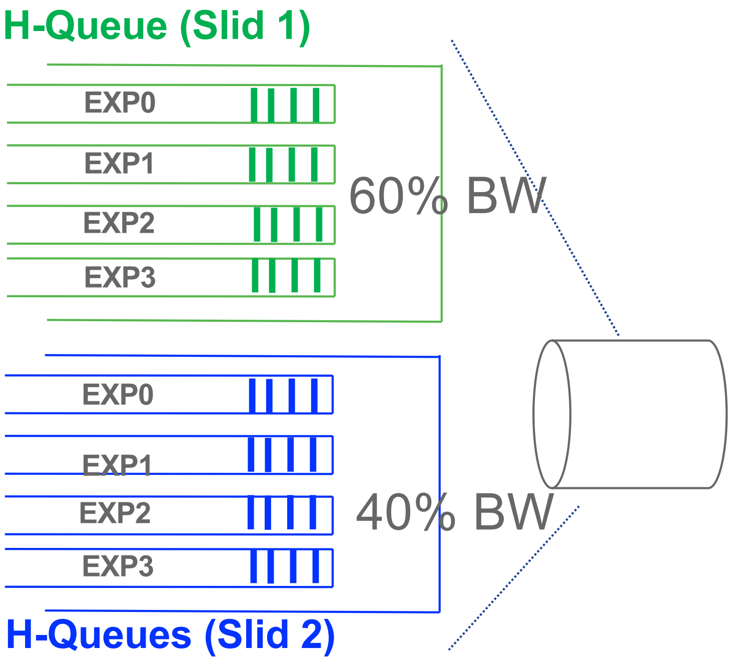
- History
  - Rev 0 was published in July 2018
  - Rev 2 was presented at IETF106

History of the Draft (Cont'ed)

- Contents Presented at IETF106
  - SR Policy - with or without Flexible Algorithm
  - Flexible Algorithm
  - TI-LFA with O(50 msec) protection
  - SR VPN
  - SR Service Programming (NFV, SFC)
  - OAM and Performance Management (PM)
  - QoS
  - Orchestration at the Controller
- Diffs
  - Stateless Slice ID (SLID) references
  - SLID works seamlessly with the rest of the slicing building blocks

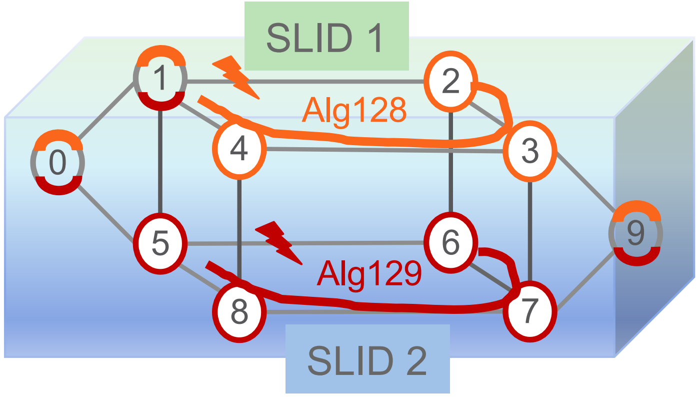
SLID Attributes

- SLID enables the differentiate treatment
  - QoS/ DiffServ policy on a per SLID
- SLID construct is like QoS
  - Independent of Routing and Topology
- Stateless
- Backward compatible
  - Incremental deployments



SLID Independence from Routing and Topology Example

- Flex-Algo and TI-LFA
  - The backup path is optimized per Flex-Algo
- Flex-Algo, TI-LFA and SLID
  - SLID does not create a new instance of Flex-Algo
    - >Scalable
  - TI-LFA works seamlessly for each SLID
    - >The SLID is stateless
    - >Backup paths provide differentiated treatment



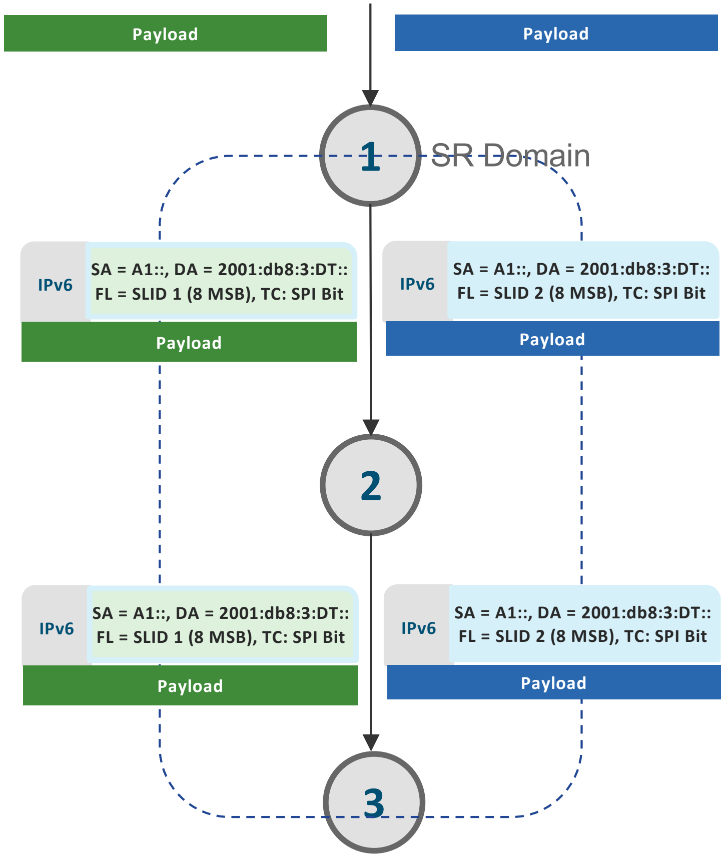
Slicing building blocks
shared among SLIDs example

Seamless Building Blocks

- SLID work seamlessly with other building blocks for scaling
  - Flex Algo
    - >Like in previous slide
      - Orange & red Flex Algo and Green & Blue SLID works seamlessly
  - VPN
  - SR Policy (with or without flex algo)
  - QoS/ DiffServ policy, etc.
- SLID is a differentiated behavior at a node
  - Not too many SLIDs are needed
    - >Scaling

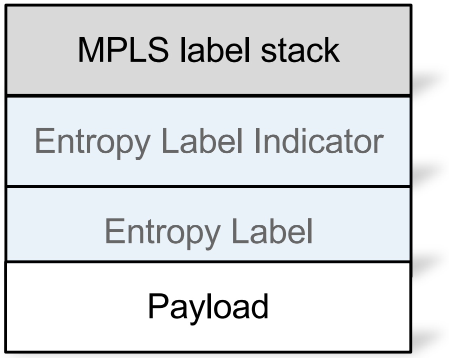
SLID for SRv6

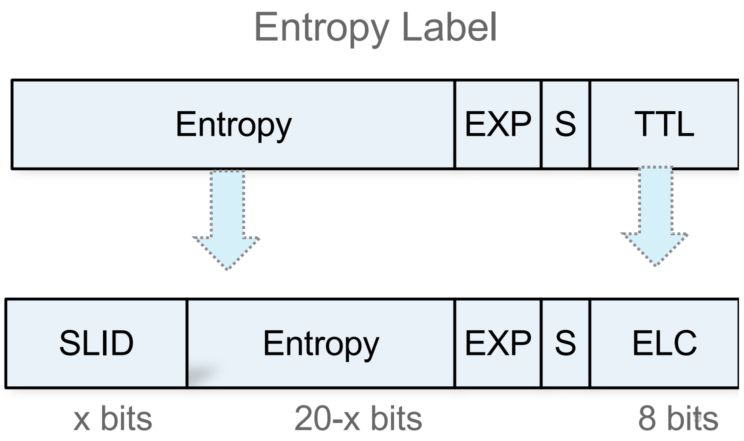
- Reference
  - draft-filsfils-spring-srv6-stateless-slice-id
- Ingress PE
  - Encapsulates an outer IPv6 header and optional SRH
  - MAY classify the traffic to a slice and sets the following in the outer IPv6 header
    - SPI bit (SLID Presence Indicator) in the TC
    - SLID in the 8 MSB of the Flow Label
- Per Slice Differential Treatment
  - The SLID is used to apply per-slice policies
- Backward Compatible
  - Node not supporting SLID provides slicing using non-SLID building blocks (default SLID)


Payload
Payload
1
SR Domain
IPv6
SA = A1::, DA = 2001:db8:3:DT::
FL = SLID 1 (8 MSB), TC: SPI Bit
Payload
IPv6
SA = A1::, DA = 2001:db8:3:DT::
FL = SLID 2 (8 MSB), TC: SPI Bit
Payload
2
IPv6
SA = A1::, DA = 2001:db8:3:DT::
FL = SLID 1 (8 MSB), TC: SPI Bit
Payload
IPv6
SA = A1::, DA = 2001:db8:3:DT::
FL = SLID 2 (8 MSB), TC: SPI Bit
Payload
3

SLID for MPLS

- Reference
  - draft-decraene-mpls-slid-encoded-entropy-label-id
- Similar to draft-filsfils-spring-srv6-stateless-slice-id for MPLS networks
- Ingress PE
  - MAY classify the traffic to a slice and sets the following in the entropy label of MPLS label stack:
    - > SLID in the x MSB of the entropy label
    - > SPI bit (SLID Presence Indicator) in the one bit of TTL field (ELC)
- Per Slice Differential Treatment
  - The SLID is used to apply per-slice policies
- Backward Compatible
  - Node not supporting SLID provides slicing using non-SLID building blocks (default SLID)





Next Steps

- The authors would like the WG provide comments
- The authors would like the WG to adopt the document