Segment Routing for End-to-End IETF Network Slicing

draft-li-spring-sr-e2e-ietf-network-slicing-01

Zhenbin Li, Jie Dong @Huawei

Ran Pang @China Unicom

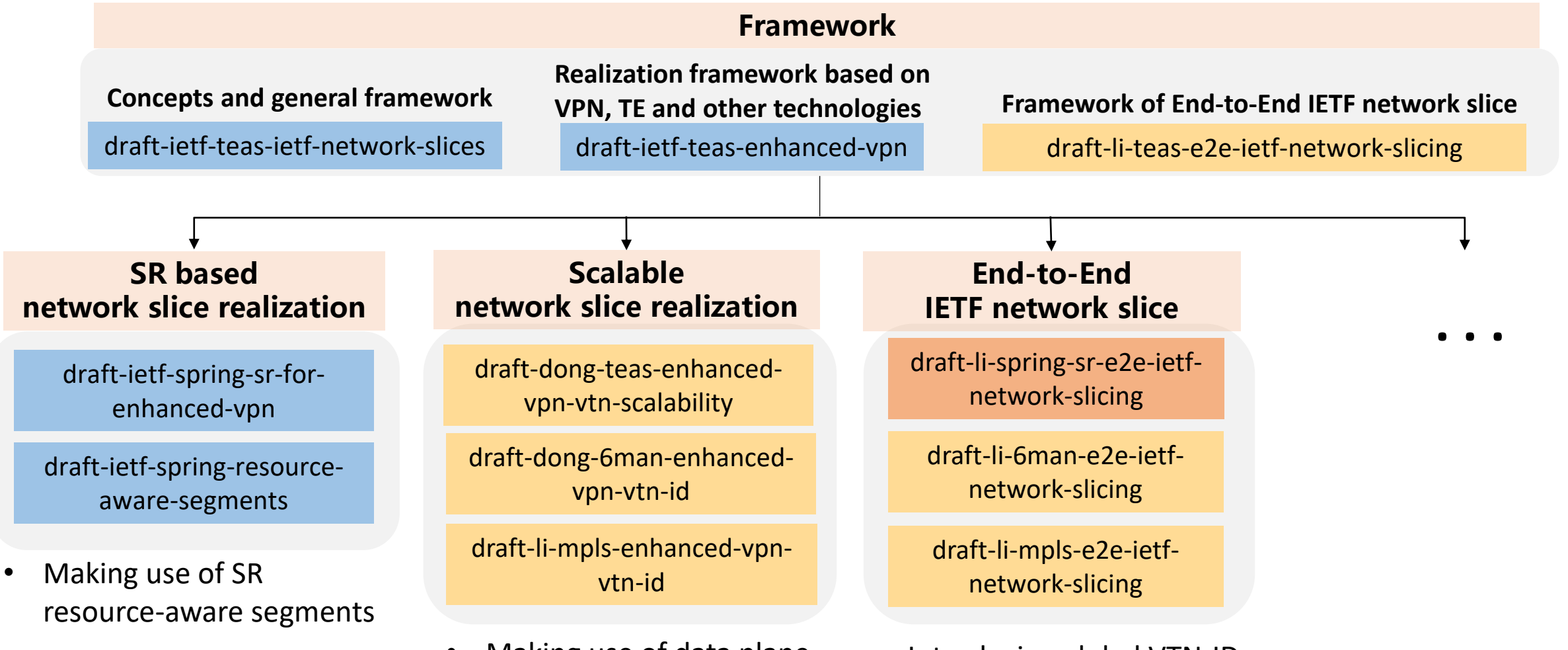
Yongqing Zhu @China Telecom

SPRING WG IETF 112 Online Meeting Nov. 2021

Background

- Network slicing can be used to meet the connectivity and performance requirement of different services or customers in a shared network
  - draft-ietf-teas-ietf-network-slices defines the concepts and general framework of IETF network slice
- IETF network slices can be realized by mapping one or a group of overlay VPNs to a VTN as the underlay
  - As described in draft-ietf-teas-enhanced-vpn
  - The SR based VPN+ mechanism is defined in draft-ietf-spring-sr-for-enhanced-vpn
- An end-to-end IETF network slice may span multiple network domains
  - In each domain, IETF network slice traffic needs to be mapped to a local VTN
- This document describes the SR extensions to support end-to-end IETF network slice
  - By introducing VTN Binding Segments

IETF Network Slice Framework and Realization



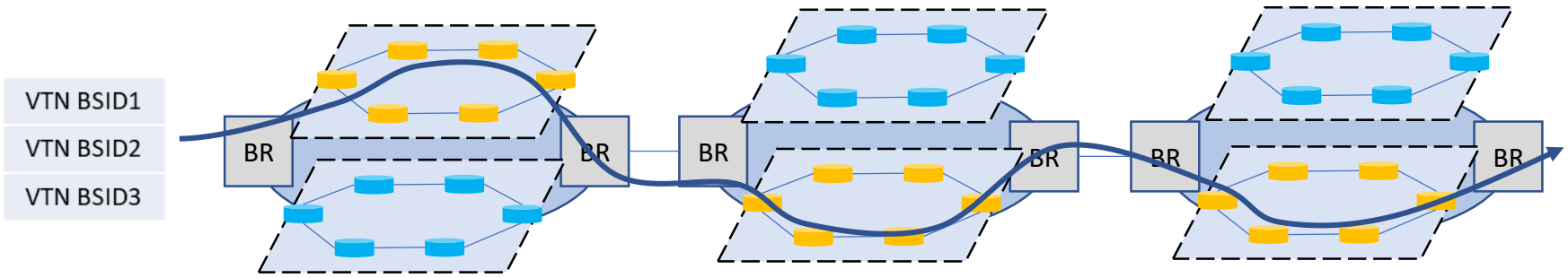
- Making use of SR resource-aware segments
- Making use of data plane VTN resource ID
- Introducing global VTN-ID and mapping mechanisms

Individual document

WG document

VTN Binding Segments

- VTN Binding Segment (BSID) is a special BSID used by the domain edge nodes to steer traffic into a local VTN
- A VTN BSID can be used to provide one of the following functions:
  1. Map the packet to a list of resource-aware segments which are associated with a local VTN
  2. Determine the local VTN-ID based on local mapping information, and instruct the encapsulation of the local VTN-ID to the data packet
  3. Obtain the local VTN-ID from the packet, and instruct the encapsulation of the local VTN-ID to the data packet
  4. Identify an SR policy which is bound to a local VTN



- The VTN BSID can be instantiated with SRv6 or SR-MPLS data plane

SRv6 Functions for VTN BSID

- The SRv6 End.B6.Encaps function defined in RFC 8986 can be used to realize the option 1 of the VTN BSID function
  - Map the packet to an SRv6 Segment List built with the resource-aware segments of the VTN
- New SRv6 functions are introduced for the other 3 VTN BSID functions
  - End.VTN.Encaps
  - End.BVTN.Encaps
  - End.B6VTN.Encaps

End.VTN.Encaps

```
Any SID instance of this behavior is associated with one VTN-ID V and
a source address A.

When node N receives a packet whose IPv6 DA is S, and S is a local
End.VTN.Encaps SID, N does the following:

S01. When an SRH is processed {
S02.   If (Segments Left == 0) {
S03.      Stop processing the SRH, and proceed to process the next
             header in the packet, whose type is identified by
             the Next Header field in the routing header.
S04.   }
S05.   If (IPv6 Hop Limit <= 1) {
S06.      Send an ICMP Time Exceeded message to the Source Address
             with Code 0 (Hop limit exceeded in transit),
             interrupt packet processing, and discard the packet.
S07.   }
S08.   max_LE = (Hdr Ext Len / 2) - 1
S09.   If ((Last Entry > max_LE) or (Segments Left > Last Entry+1)) {
S10.      Send an ICMP Parameter Problem to the Source Address
             with Code 0 (Erroneous header field encountered)
             and Pointer set to the Segments Left field,
             interrupt packet processing, and discard the packet.
S11.   }
S12.   Decrement IPv6 Hop Limit by 1
S13.   Decrement Segments Left by 1
S14.   Update IPv6 DA with Segment List [Segments Left]
S15.   Set the VTN-ID option to V in the HBH Ext header
S16.   Submit the packet to the egress IPv6 FIB lookup for
          transmission to the new destination
S17. }
```

End.BVTN.Encaps

```
When node N receives a packet whose IPv6 DA is S, and S is a local
End.BVTN.Encaps SID, N does the following:

S01. When an SRH is processed {
S02.   If (Segments Left == 0) {
S03.      Stop processing the SRH, and proceed to process the next
             header in the packet, whose type is identified by
             the Next Header field in the routing header.
S04.   }
S05.   If (IPv6 Hop Limit <= 1) {
S06.      Send an ICMP Time Exceeded message to the Source Address
             with Code 0 (Hop limit exceeded in transit),
             interrupt packet processing, and discard the packet.
S07.   }
S08.   max_LE = (Hdr Ext Len / 2) - 1
S09.   If ((Last Entry > max_LE) or (Segments Left > Last Entry+1)) {
S10.      Send an ICMP Parameter Problem to the Source Address
             with Code 0 (Erroneous header field encountered)
             and Pointer set to the Segments Left field,
             interrupt packet processing, and discard the packet.
S11.   }
S12.   Obtain the VTN-ID V from the argument part of the IPv6 DA
S13.   Decrement IPv6 Hop Limit by 1
S14.   Decrement Segments Left by 1
S15.   Update IPv6 DA with Segment List [Segments Left]
S16.   Set the VTN-ID option to V in the HBH Ext header
S17.   Submit the packet to the egress IPv6 FIB lookup for
          transmission to the new destination
S18. }
```

End.B6VTN.Encaps

```
S01. When an SRH is processed {
S02.   If (Segments Left == 0) {
S03.      Stop processing the SRH, and proceed to process the next
             header in the packet, whose type is identified by
             the Next Header field in the routing header.
S04.   }
S05.   If (IPv6 Hop Limit <= 1) {
S06.      Send an ICMP Time Exceeded message to the Source Address
             with Code 0 (Hop limit exceeded in transit),
             interrupt packet processing, and discard the packet.
S07.   }
S08.   max_LE = (Hdr Ext Len / 2) - 1
S09.   If ((Last Entry > max_LE) or (Segments Left > Last Entry+1)) {
S10.      Send an ICMP Parameter Problem to the Source Address
             with Code 0 (Erroneous header field encountered)
             and Pointer set to the Segments Left field,
             interrupt packet processing, and discard the packet.
S11.   }
S12.   Decrement IPv6 Hop Limit by 1
S13.   Decrement Segments Left by 1
S14.   Update IPv6 DA with Segment List [Segments Left]
S15.   Push a new IPv6 header with its own SRH containing B, and
        the VTN-ID option set to V in the HBH Ext header
S16.   Set the outer IPv6 SA to A
S17.   Set the outer IPv6 DA to the first SID of B
S18.   Set the outer Payload Length, Traffic Class, Flow Label,
          Hop Limit, and Next Header fields
S19.   Submit the packet to the egress IPv6 FIB lookup for
          transmission to the new destination
S20. }
```

VTN BSID in SR-MPLS

- Similarly, VTN BSID can be instantiated using SR-MPLS Binding SIDs with different semantics
  - Please refer to the draft for the details

Next Steps

- Comments and feedback are welcome
- Refine this draft accordingly

Thank You