Segment Routing for Redundancy Protection

draft-geng-spring-sr-redundancy-protection-02
draft-geng-6man-redundancy-protection-srh-00

Presenter: Fan Yang

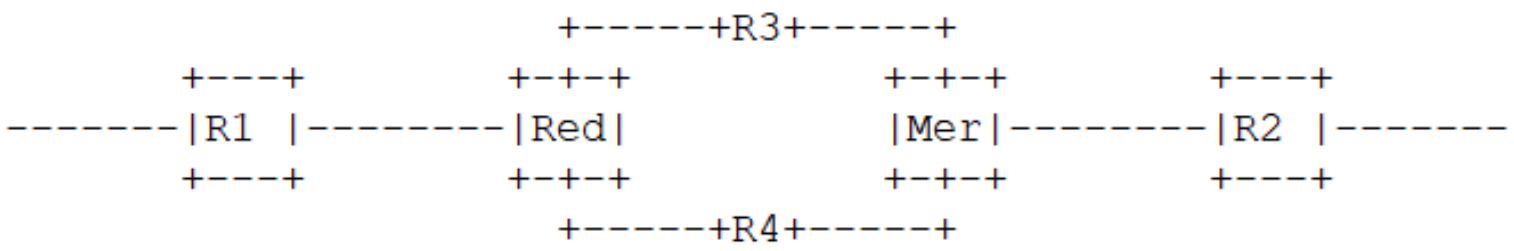
Xuesong Geng (gengxuesong@huawei.com)

Mach Chen (mach.chen@huawei.com)

Fan Yang (shirley.yangfan@huawei.com)

What is Redundancy Protection?

- Service Protection comes from Deterministic Networking (DetNet)
- New requirement for providing strict E2E reliability SLA guarantee to services, e.g. cloud VR, cloud game, HDV applications
- Definition
  - is one of the mechanisms to achieve service protection
  - follows the principle of PREOF (Packet Replication/ Elimination/Ordering Function)
- Example scenario:



Segment to support Redundancy Protection

- **Redundancy Segment:**
  - to perform the packet replication function on Redundancy Node
  - associated with a Redundancy policy (a variant of SR Policy) to steer the flow
  - in case of SRv6, new behavior End.R is defined
- **Merging Segment:**
  - to perform the packet elimination and ordering (optional) function on Merging Node
  - in case of SRv6, new behavior End.M is defined
- **Flow ID and sequence number:**
  - Flow Identification: to identify a unique flow
  - Sequence Number: to identify the packet sequence within one flow
- **Redundancy Policy:**
  - includes more than one ordered lists of segments between Redundancy Node and Merging Node
  - all the ordered lists of segments are used at the same time

Updates since IETF 109

1) Update Redundancy Segment with BSID-like behavior
2) Specify process of determining packet redundancy at Merging Segment
3) Redesign the process
4) Flow identification/sequence number encapsulation in another draft of 6man
5) Split the segment description in SR-MPLS and SRv6
6) Add structure description of redundancy policy in section 6

Redundancy Segment

- Updated as a variant of BSID
- Decouple replication behavior and marking of flow identification and sequence number
  - Packet replication is done by redundancy segment
  - Two deployments options to assign Flow ID:
    1. FI is unique in global domain, assigned to headend via SR-policy
    2. FI is unique between R and M nodes, assigned to R node via redundancy policy
  - Sequence number is added according to the configuration based on 5-tuple/color/sr-policy
  - Two deployment options to add seq num:
    1. SN is added at headend
    2. SN is added at R node

S01. When an SRH is processed {
S02. If (Segments Left>0) {
S03. Decrement IPv6 Hop Limit by 1
S04. Decrement Segments Left by 1
S05. Update IPv6 DA with Segment List[Segments Left]
S06. Create two new IPv6 headers with SRH-1 and SRH-2 respectively
S07. Insert different policy-instructed segment lists into SRH-1 and SRH-2
~~Add Flow Identification and Sequence Number to SRH-1 and SRH-2~~
~~Remove the incoming outer IPv6+SRH header~~
S08. Create a duplication of the incoming packet
S09. Encapsulate the original packet with the new IPv6+SRH-1 header
S10. Encapsulate the duplicate packet with the new IPv6+SRH-2 header
S11. Set IPv6 SA as the local address of this node
S12. Set IPv6 DA of IPv6+SRH-1 to the first segment of SRH-1 SL
S13. Set IPv6 DA of IPv6+SRH-2 to the first segment of SRH-2 SL
S14. Add flow identification and sequence number to SRH-1
S15. Add flow identification and sequence number to SRH-2
S16. Set the outer Payload Length, Traffic Class, Flow Label,Hop Limit and Next-Header fields
S17. Submit the packet to the egress IPv6 FIB lookup and transmit
S18. }
S19. ELSE {
S20. Drop the packet
S21. }
S22. }

Merging Segment

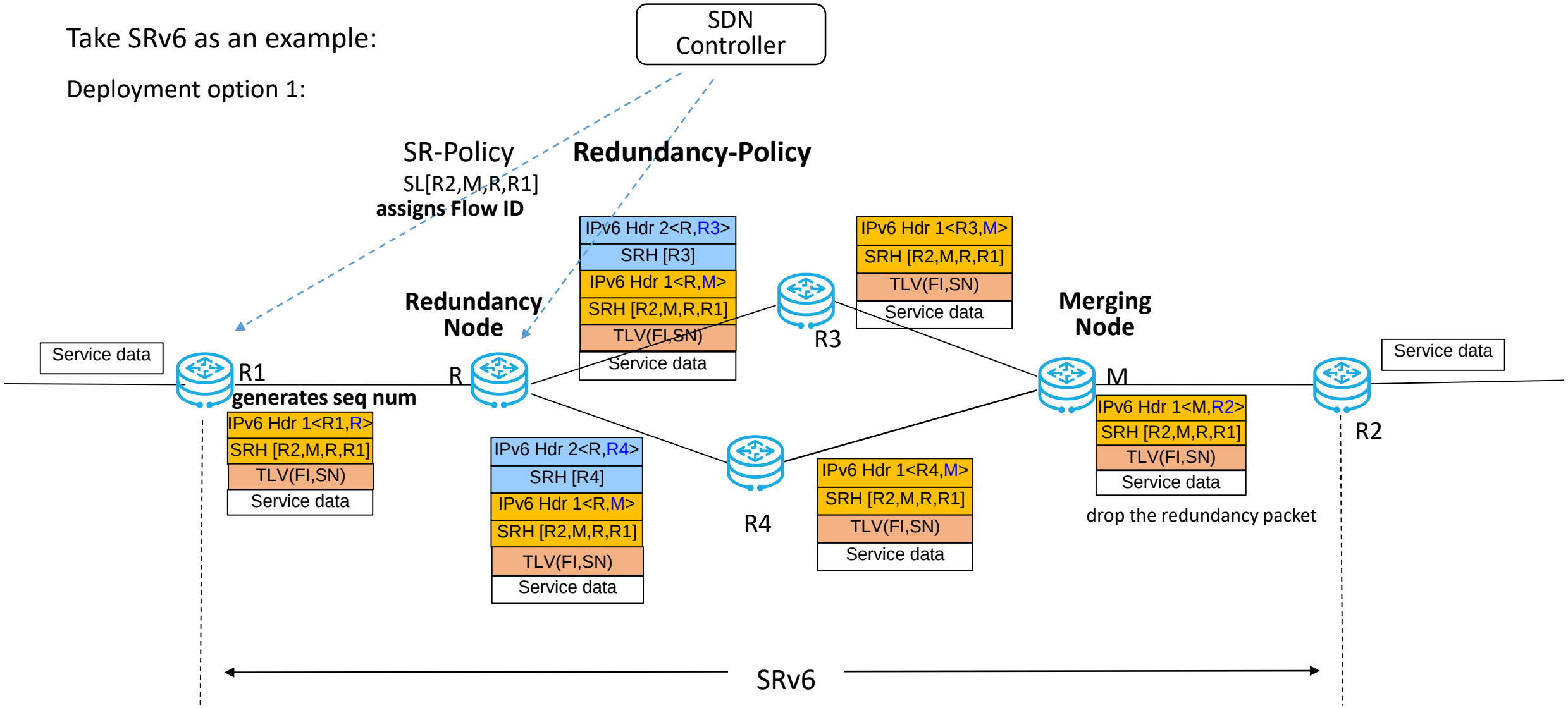
- Specify the determination of packet redundancy is based on sequence number
- Changes on the process of decap and encap of IPv6 header

```
S01.  When an SRH is processed {
        ~~If (Segments Left>0) & "the packet is not a redundant packet" {~~
S02.    If (Segments Left==0)   {
S03.        Acquire the sequence number of received packet and lookup it in a local table
S04.        If (the sequence number is not existed in table ) {
S05.            Store the packet and record the sequence number in table
S06.            Remove the outer IPv6+SRH header
S07.            Decrement IPv6 Hop Limit by 1 in inner SRH
S08.            Decrement Segments Left by 1 in inner SRH
S09.            Update IPv6 DA with Segment List[Segments Left] in inner SRH
S10.            Submit the packet to the egress IPv6 FIB lookup and transmit
S11.        }
S12.        ELSE {
S13.            Drop the packet
S14.        }
S15.    }
S16.  }
```

Redundancy Protection Process

Take SRv6 as an example:

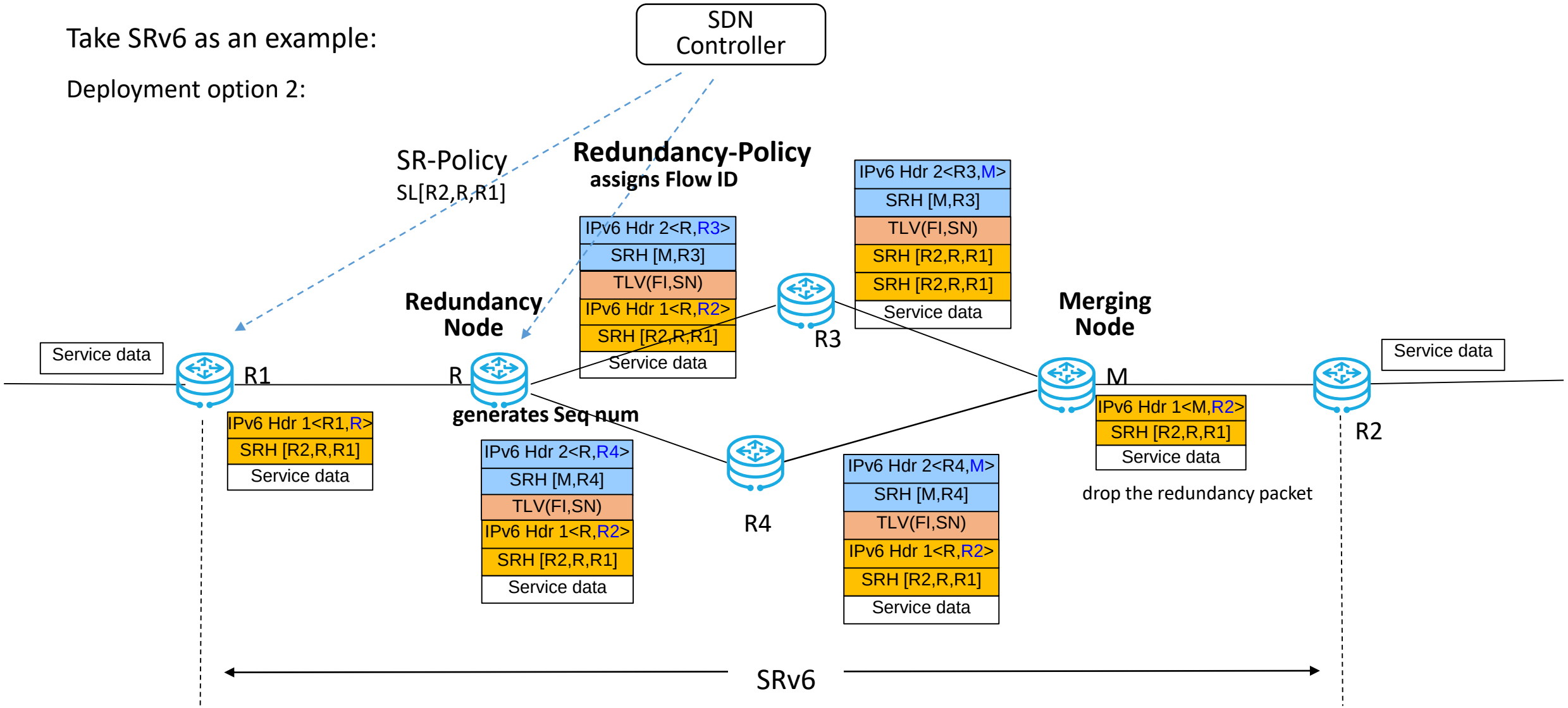
Deployment option 1:



Redundancy Protection Process

Take SRv6 as an example:

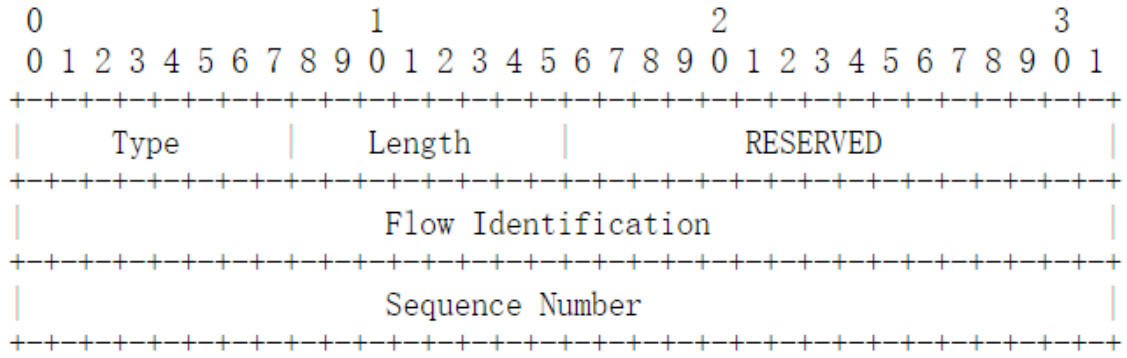
Deployment option 2:



Flow ID and Sequence Number Encapsulation

SRH Encapsulation in *draft-geng-6man-redundancy-protection-srh-00*

A TLV is defined to carry flow ID and sequence number



- Flow Identification: 32 bits, to identify a unique flow
- Sequence Number: 32 bits, to identify the packet sequence within one flow

Redundancy Policy

- Redundancy Policy is a variant of SR policy
- is identified through the tuple <redundancy node, redundancy ID, merging node>
  - ✓ Redundancy node is specified as IPv4/IPv6 address of redundancy node
  - ✓ Merging node is specified as IPv4/IPv6 address of merging node
  - ✓ Redundancy ID could be a specified value of "color", which indicates the SR policy as a redundancy policy
  - ✓ Redundancy ID could also be used to distinguish different redundancy policies sharing the same redundancy node and merging node
- includes more than one ordered lists of segments between redundancy node and merging node
- all the ordered lists of segments are used at the same time

Next Step

- Refine the overall solution and SRH encapsulation
- Specify the redundancy policy in another draft
- Comments and discussions in mailing list
- Seek for collaborations
  - Segment specification in SR-MPLS data plane
  - Scalability discussion of flow ID and sequence number

Thanks

Scenarios of redundancy node and merging node

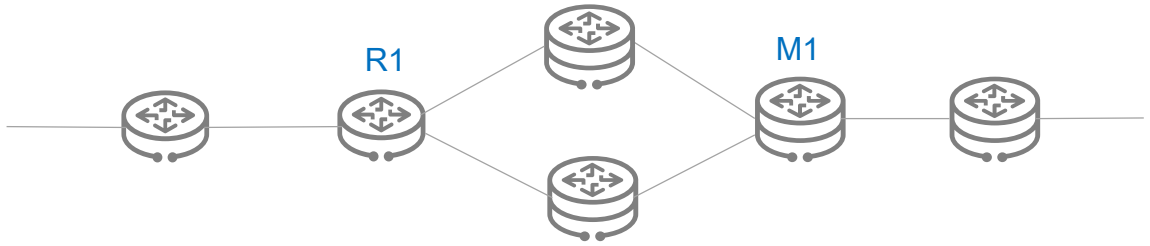


Figure 1 simplified 1:1

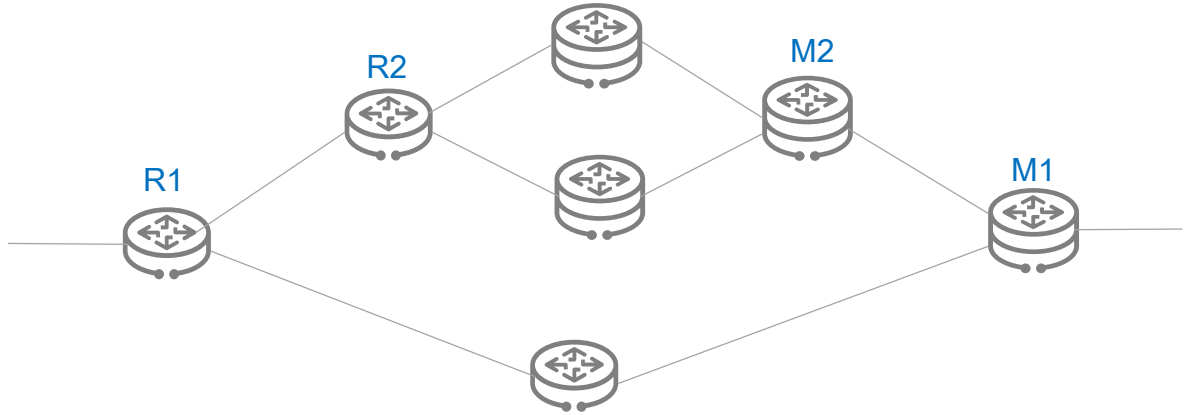


Figure 2 Russian doll

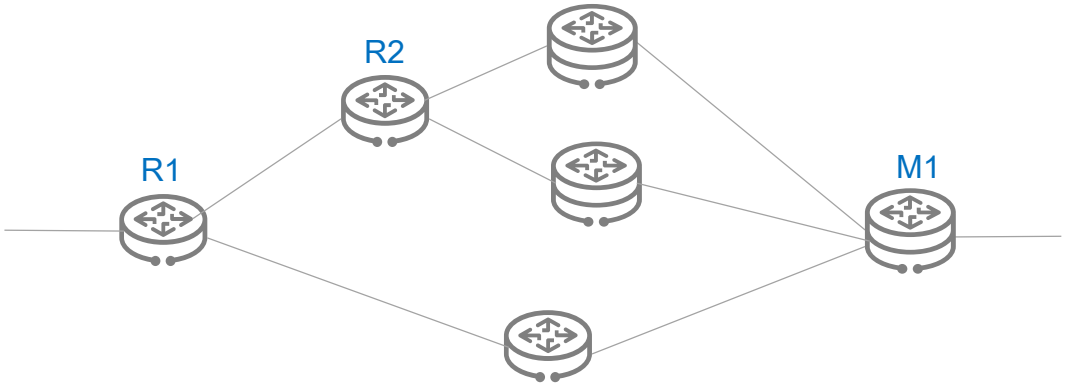


Figure 3 n:1

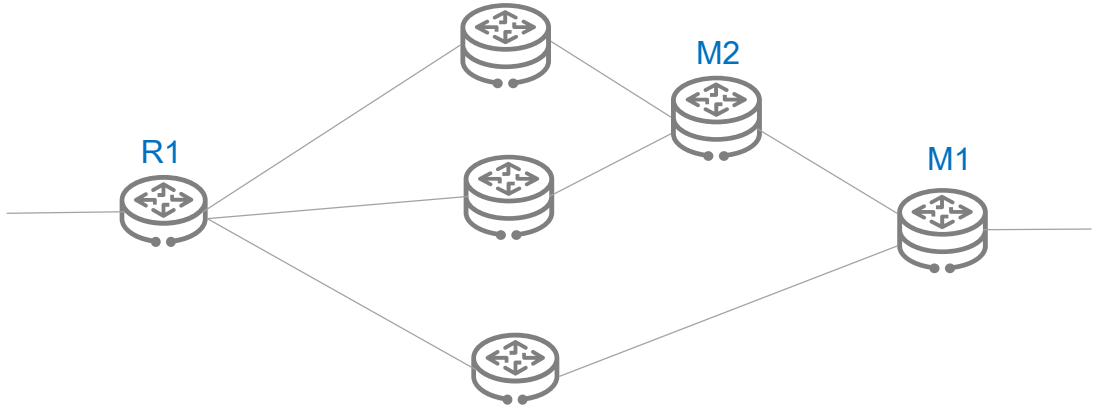


Figure 4 1:n