White Paper

XFLOW RESEARCH

Segment Routing over IPv6 (SRv6)

Author

Ghulam Bahoo

Author

Hafiz Mati Ur Rahman

Contents

1 Introduction

Source Routing: it is a control mechanism that allows an originating router to direct a packet through a predetermined sequence of nodes and links in a network. Unlike traditional routing methods, intermediate nodes do not need to determine the path that the packet should take. In this context, the term ”source” refers to the location where the specified route is initiated [1].

Segment Routing (SR): A network protocol has been developed to forward data packets based on source routes, known as Segment Routing. This protocol involves dividing a network path into multiple segments and assigning each segment and forwarding node a segment ID (SID). The segments and nodes are then arranged in a segment list in sequential order to create a forwarding path.

Segment Routing is categorized into two types based on the forwarding plane utilized. The first type is known as Segment Routing MPLS and utilizes the MPLS forwarding plane. The second type is known as Segment Routing IPV6, which is also referred to as SRV6 and utilizes the IPV6 forwarding plane [2].

IPv6: The IPv6 forwarding plane is utilized by a networking protocol that is designed to facilitate communication over a network. IPv6 is a version of the Internet Protocol that operates at the network layer and is responsible for identifying and locating endpoint systems on a computer network. It enables the routing of online traffic and aims to solve the problem of IPv4 address exhaustion that has arisen due to prolonged internet use worldwide. The development of IPv6 was overseen by the Internet Engineering Task Force (IETF).

Segment Routing IPv6 (SRv6) is a next-generation IP bearer protocol that combines Segment Routing (SR) and IPv6. Utilizing existing IPv6 forwarding technology, SRv6 implements network programming through flexible IPv6 extension headers.

SRv6 reduces the number of required protocol types, offers great extensibility and programmability, and meets the diversified requirements of more new services. It also provides high reliability and offers exciting cloud service application potential.

2 Background

2.1 Traditional Networks Challenges

The rapid progress of global digitization is driving the expansion of Internet-based technologies. As networks continue to expand in size and we enter the era of cloud computing, we encounter a wider range of network services and their associated demands on the networks. Consequently, conventional IP/MPLS networks are confronted with a series of challenges:

- **Isolated IP bearer Network Islands:** Despite the unifying capabilities of MPLS as a bearer network technology, there exist isolated IP bearer network islands, including the IP backbone, metro, and mobile bearer networks. These domains operate as separate MPLS entities and require complex technologies such as inter-AS VPN for interconnectivity, leading to added complexity in deploying end-to-end services. Furthermore, coexistence of L2VPN and L3VPN services on a device may involve a multitude of protocols (e.g., LDP, RSVP, IGP, and BGP), which can complicate management and pose difficulties for large-scale service deployment.

- **Limited programming space in IPv4 and MPLS:** The evolving needs of modern services demand the inclusion of additional forwarding information in packets. Unfortunately, the IETF has announced that it will cease developing new standards for IPv4. Moreover, the MPLS label space, which is limited to 20 bits and lacks extensibility, can no longer adequately accommodate the requirements of network programming for new services.

- **Decoupling of applications and bearer networks:** The practice of decoupling presents a challenge to optimizing networks and enhancing their value, often leaving carriers limited to providing basic connectivity without benefiting from value-added applications. Additionally, the absence of application-specific information restricts carriers to performing coarse-grained network adjustments and optimizations, resulting in wastage of resources. Over time, various efforts have been made to integrate MPLS more closely with user hosts and applications, but such endeavors have been unsuccessful, partly due to the complexity of network borders and management.

- **Tight coupling of the data and control planes:** These planes are bound together for sale and evolution, prolonging service provisioning and making it difficult to cope with the rapid development of new services [2].

2.2 From Simplicity to Complexity

In the early days of networking, communication protocols were simple and limited in functionality. The first networking protocol, ARPANET, was a simple packet-switching protocol that allowed computers to communicate over a network. However, as the internet began to grow in popularity in the 1980s and 1990s, more complex protocols were needed to support new applications and services. The Transmission Control Protocol/Internet Protocol (TCP/IP) became the standard protocol for the internet, providing reliable data transmission and routing.As networks continued to grow larger and more complex, new protocols were developed to manage and optimize network performance. The Border Gateway Protocol (BGP) was introduced to route traffic between autonomous systems, while the Simple Network Management Protocol (SNMP) was developed to monitor and manage network devices. These protocols were instrumental in enabling the internet to support a growing number of users and applications. In recent years, protocols have become even more complex and specialized to support emerging technologies such as virtualization and cloud computing as shown in Figure 1. Multiprotocol Label Switching (MPLS) was developed to provide faster routing of network traffic, while Virtual Extensible LAN (VXLAN) was introduced to support the creation of virtual networks in cloud computing environments. Other protocols, such as the Internet Group Management Protocol (IGMP) and the Session Initiation Protocol (SIP), have been developed to support multimedia streaming and voice over IP (VoIP) services.

Overall, the evolution of networking protocols has been driven by the need for greater functionality and efficiency in a growing and increasingly complex digital landscape. As the internet continues to evolve, new protocols will continue to be developed to support emerging technologies and applications [3].

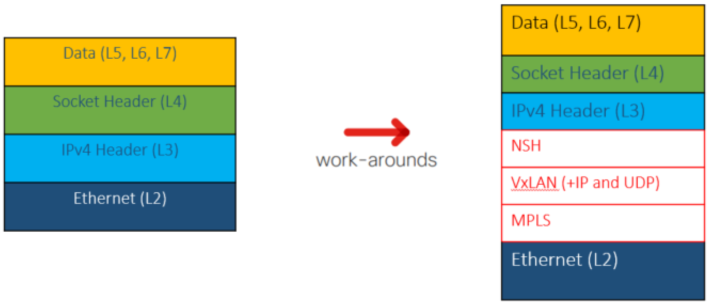


Figure 1: From Simplicity to Complexity

2.3 From complexity to simplicity

Segment Routing over IPv6 (SRv6) is a network architecture that simplifies network operations by reducing the complexity of network infrastructure. Traditional network architectures rely on complex routing protocols and overlays to manage network traffic, leading to a high level of complexity that can be difficult to manage and troubleshoot. SRv6 simplifies this process by using the IPv6 data plane to create a flexible and programmable network architecture. This simplifies network operations by reducing the number of protocols and overlays required to manage network traffic, while also providing greater flexibility and control over network routing as shown in Figure 2. With SRv6, network administrators can easily create and modify network paths without the need for complex protocols, reducing the time and effort required to manage network infrastructure [3].

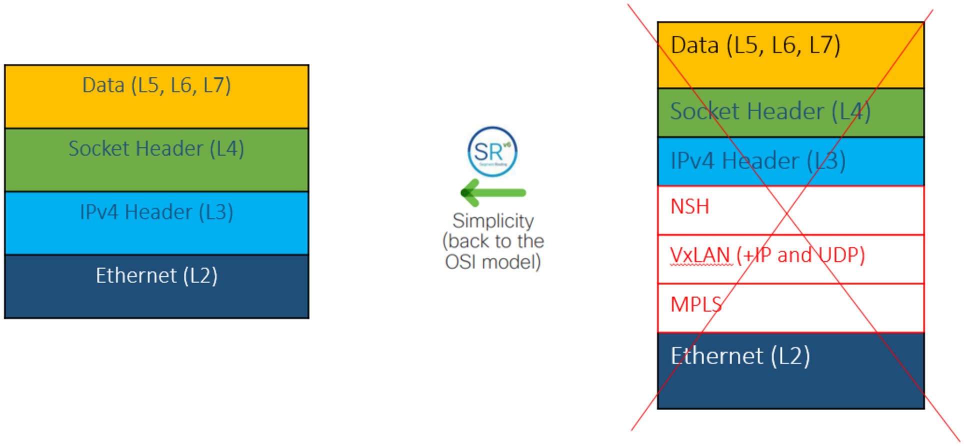


Figure 2: From Complexity to Simplicity

2.4 Advantages of SRV6

1. **Simplified Network Operation**: SRv6 simplifies network operation by allowing network operators to program the network paths that packets take through the network. This reduces the need for complex routing protocols and simplifies the management of the network.

2. **Better Network Scalability**: SRv6 improves network scalability by reducing the number of labels required to implement a given network topology. This reduces the burden on network devices and improves their ability to scale to larger networks.

3. **Enhanced Network Security**: SRv6 provides enhanced network security by allowing network operators to apply security policies at the network edge. This can include routing policies, access control lists, and other security features that help to secure the network.

4. **Improved Network Performance**: SRv6 improves network performance by allowing network operators to define explicit paths for packets to take through the network. This can improve the efficiency of the network and reduce packet loss and latency.

3 Segment Routing Architecture

Segment Routing (SR) is a routing paradigm that relies on source routing. In this paradigm, a node guides a packet by following a sequence of ordered instructions called ”segments”. Each segment can represent a topological or service-based instruction and may have a local or global meaning. By using SR, it is possible to restrict a flow to a specific path while keeping per-flow state only at the ingress node(s) to the SR domain.

With the MPLS architecture, where a segment is represented by an MPLS label and an ordered list of segments is a stack of labels. The topmost segment on the stack is the one to be processed, and after it's done, the label is popped from the stack.

With the IPv6 architecture, where a segment is represented by an IPv6 address and an ordered list of segments is an ordered list of IPv6 addresses in the routing header. The currently active segment is indicated by the packet's Destination Address (DA), and a pointer in the new routing header specifies the next active segment [4].

The SR architecture consists of two key components. The first component is the **data-plane**, which specifies how to encode the sequence of segments to be applied to a packet and how each device should process the packet based on a segment. It is important to note that this operation of SR does not depend on the protocol used to carry the SR header information.

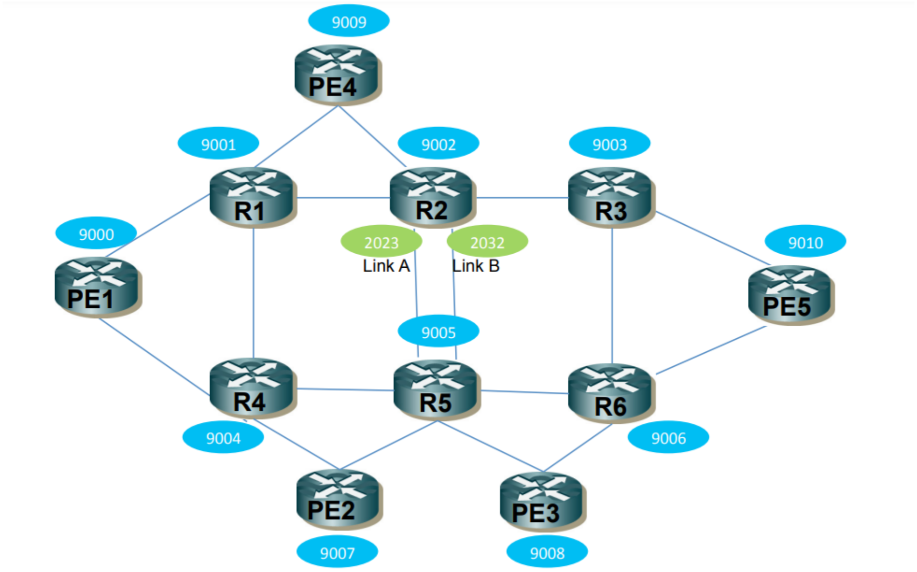


Figure 3: Segment Routing Architecture

The second component is the **control-plane**, which focuses on how segment identifiers are distributed among network devices and how these devices are instructed to apply a particular sequence of segments to a flow.

A. SR data-plane

The SR header of a packet contains a sequence of segments and a pointer to the currently active segment, which represents the instruction that the device processing the packet should execute. After executing the active segment, the device moves on to the next segment on the list, which then becomes the active one. Each segment is identified by a Segment ID (SID), which may have either domain-wide significance or be significant only locally to the router that is processing it as shown in Figure 3.

In terms of the different types of segments that exist, the following are the most common:

- **Node SID**: The Node SID has a forwarding semantic that directs the packet towards the Node associated with that specific Segment ID through the shortest path available. To achieve this, the network operator assigns a unique Node segment ID to each router within the network. This process can be carried out either manually or through a centralized controller.

- **Adjacency SID**: An Adjacency SID has a forwarding semantic associated with it that directs the packet to be forwarded over the corresponding adjacency. To accomplish this, each router in the network assigns a segment ID that is locally significant to each of its Interior Gateway Protocol (IGP) adjacencies.

- **Service SID**: A Service SID has a forwarding semantic that directs the packet to the specific service provided by the node processing the packet. Each node in the network will assign a locally significant segment ID for each service it offers.

An SR-enabled node supports the following data-plane operations:

- **CONTINUE** - Forwarding action performed based on active segment.

- **PUSH** - Add a segment ahead of the SR header of the packet and set that segment as the active segment.

- **NEXT** - Mark the next segment as the active segment.

The topology shown in Figure 4 has assigned a Node SID to each router. For example, Node SIDs 9002, 9003, and 9010 are assigned to routers R2, R3, and PE5, respectively. The Adjacency SIDs 2023 and 2032

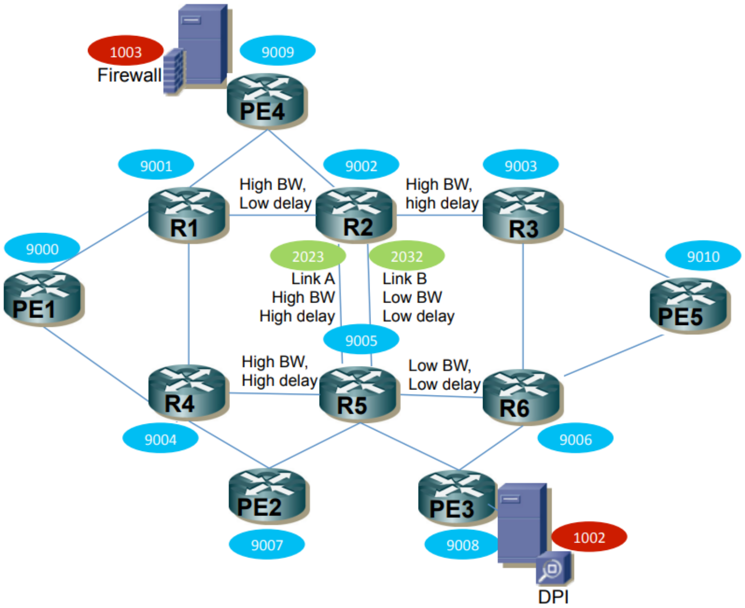


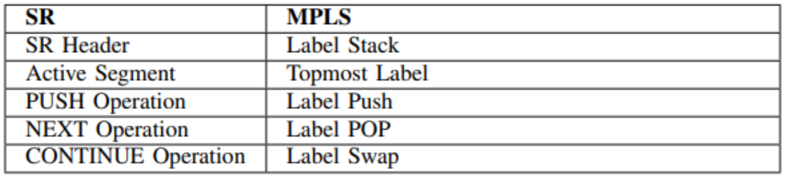
Figure 4: SRV6 Packet Flow

are assigned by R2 for its links to R5 via Link A and Link B, respectively. To simplify the diagram, the rest of the Adjacency SIDs are not shown, but each device may assign an Adjacency SID for each available adjacency.

When sending packets to PE5, PE1 only needs to use Node SID 9010 in the SR header. The packets are then load-balanced over the shortest paths to PE5, as determined by the IGP. To enforce a specific path via R2 in the reference topology, PE1 can use a PUSH operation to set the segment list to 9002, 9010. The packet is then sent to R3, following the meaning of Node SID 9002. R3 performs a CONTINUE operation on this SID, forwarding the packet to R2. Upon receiving the packet, R2 moves the pointer to the next segment (NEXT), which is 9010, and forwards the packet towards R3.

Alternatively, if the packet is required to flow over the path R2-R5-R6-PE5, moving between R2 and R5 using link A, PE1 would use the segment list 9002, 2023, 9010. Upon receiving the packet, R2 moves the pointer to the next segment, which is 2023, identifying the instruction to perform a NEXT operation and forward over link A. The packet then reaches R5 with active segment 9010, and shortest path forwarding proceeds from there to the destination. After reaching R5, the packet will be forwarded to its destination following the shortest path. It's worth noting that using SR, source routers can achieve a high level of path definition flexibility without requiring additional state to be kept in intermediate routers, as is the case with RSVP-TE. While this additional state may be irrelevant for this particular example, it can pose difficulties for ISPs managing networks with thousands of service chains.

Operators are free to choose the SR data-plane technology that best suits their network requirements as shown in Figure 5. Currently, MPLS and IPv6 are the two data-plane technologies considered for SR support, as they are typical data-planes for such networks:



SR	MPLS
SR Header	Label Stack
Active Segment	Topmost Label
PUSH Operation	Label Push
NEXT Operation	Label POP
CONTINUE Operation	Label Swap

Figure 5: SR Operations Mapping To MPLS Label Operations

B. SR control-plane

In a Segment Routing (SR) network, the communication of segment ID information among devices is facilitated by the control-plane. To achieve this, the link state Interior Gateway Protocol (IGP) is used to advertise Node and Adjacency SIDs. Figure 6 shows service chainning and traffic engineering use-cases. This has been extended in popular IGPs such as ISIS and OSPF to enable the distribution of segment IDs throughout the network. With these extensions, any router can maintain a database of all nodes and adjacency segments, and the database can be quickly updated after any topology change due to the sub-second convergence properties of both IGPs. The use of these extensions enables end-to-end encapsulation in the network without the need for enabling and managing another protocol like LDP. Another element of the control-plane of SR deals with how an ingress router is instructed to select the SR path that a packet should follow. The following methods can be used for this purpose:

1. **Distributed Constrained SPF (CSPF) calculation**. Under this approach, an ingress router computes the shortest path to a destination while ensuring that the path satisfies certain criteria. It then generates a sequence of node and adjacency segments that represent this path.

2. **SDN controller based approach**. The incorporation of Segment Routing (SR) into network designs offers a scalable and resilient data-plane, while still providing the flexibility of control that is typically associated with Software-Defined Networking (SDN) environments. This has resulted in the planned adoption of SR into the architectures of some SDN-oriented controllers. OpenDaylight is one such example, as it supports the control of SR through the use of the Path Computation Element Protocol (PCEP).

3. **Statically defined by the operator.** Although static configuration of tunnels can be utilized for certain purposes, such as testing or troubleshooting, it is generally not advisable for long-term network operation. This is because it presents several evident limitations in terms of scaling, resiliency, and management.

   Operators have the flexibility to select any of these methods, depending on the applications and scenarios they wish to support. It is important to note that all three strategies can exist simultaneously within the same network. For instance, static tunnels could be utilized for troubleshooting or specific, infrequent purposes. The Constrained Shortest Path First (CSPF) method strikes a balance between connectivity optimization and automation. On the other hand, centralized approaches offer greater flexibility, making them highly attractive for networks with Traffic Engineering (TE) objectives, especially in situations where conflicting decisions could arise when performed in a distributed manner (e.g., demand placement for capacity engineering purposes).

3.1 Segment Routing Domain

A Segment Routing (SR) domain refers to a group of nodes that participate in the source-based routing model. These nodes can be connected to the same physical infrastructure, such as a Service Provider's network, or remotely connected to each other, like an enterprise VPN or overlay. In cases where multiple protocol instances are deployed, the SR domain typically encompasses all of the protocol instances in the network. However, some deployments may choose to divide the network into multiple SR domains, each containing one or more protocol instances. It is essential to note that all nodes within an SR domain are typically managed by the same administrative entity.

3.2 Active Segment

This segment refers to the identifier utilized by the recipient router to handle the incoming packet. In the context of the MPLS data plane, it corresponds to the top label. In contrast, in the IPv6 data plane, the destination address is used as the segment identifier.

3.3 SR Global Block (SRGB)

The SRGB refers to the collection of global segments present within an SR domain. It is important to note that if a node participates in multiple SR domains, there is a distinct SRGB for each domain. In SR-MPLS, the SRGB is a local attribute of a node that identifies the set of reserved local labels for global segments. For ease of operation and troubleshooting, it is strongly recommended to use identical SRGBs across all nodes within an SR domain. This ensures that the same label represents the same global segment at each node. In contrast, in SRv6, the SRGB consists of the global SRv6 SIDs present within the SR domain.

3.4 SR Local Block (SRLB)

The SRLB is a characteristic specific to each SR node. If a node takes part in multiple SR domains, there is a unique SRLB for each domain. In the context of SR-MPLS, the SRLB is a set of local labels reserved for local segments. Similarly, in SRv6, the SRLB is a set of local IPv6 addresses reserved for local SRv6 SIDs. In controller-driven networks, certain controllers or applications may utilize the control plane to identify the set of available local segments.

3.5 Global Segment

A global segment is a segment that belongs to the SRGB of the SR domain. Its associated instruction is determined at the SR domain level. An instance of a global segment is a topological shortest-path segment to a specific destination within the SR domain.

3.6 Local Segment

This refers to a segment that is not included in the SRGB in SR-MPLS or can be any IPv6 address in SRv6, even if it is part of the SRGB. In SR-MPLS, this label is a local label that is reserved for local segments and can be part of the SRLB. In SRv6, this can be any IPv6 address that has local significance, and the instruction associated with the segment is defined at the node level.

3.7 Segment Routing Operatins

1. **PUSH**: This term refers to the act of adding a segment at the beginning of a segment list. In SR-MPLS, the top of the segment list refers to the highest (outermost) label of the label stack. Meanwhile, in SRv6, the top of the segment list is denoted by the first segment in the Segment Routing Header, as defined in IPv6 Segment Routing Header (SRv6-SRH).

2. **NEXT**: When the active segment is completed, NEXT is the operation

   consisting of the inspection of the next segment. The next segment becomes active. In SR-MPLS, NEXT is implemented as a POP of the top label. In SRv6, NEXT is implemented as the copy of the next segment from the SRH to the destination address of the IPv6 header.

3. **Continue**: the active segment is not completed; hence, it remains active. In SR-MPLS, the CONTINUE operation is implemented as a SWAP of the top label. In SRv6, this is the plain IPv6 forwarding action of a regular IPv6 packet according to its destination address.

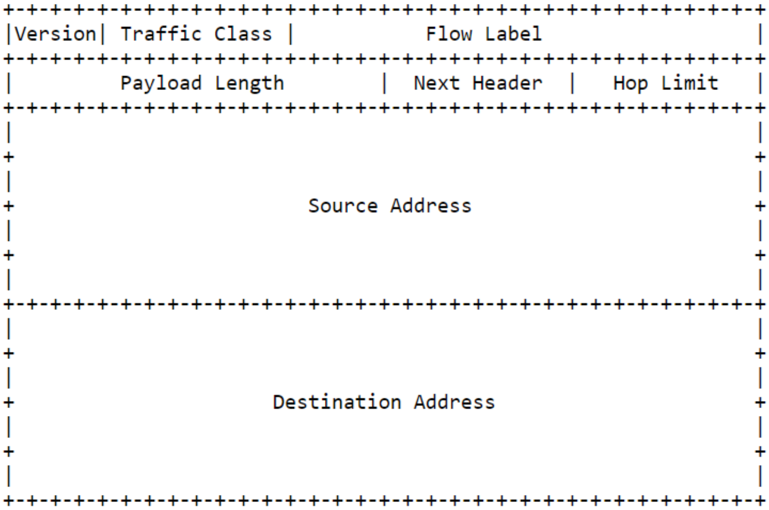
4 IPV6

An IPv6 address is a unique alphanumeric value consisting of 128 bits that identifies a device connected to an Internet Protocol Version 6 (IPv6) network. IPv6 was developed as the successor to IPv4, which had limitations that IPv6 aimed to overcome. One major difference is that IPv6 offers significantly larger address space than IPv4.

IP is a method for transmitting data between computers over the internet. Each computer or network interface on the internet is assigned at least one IP address to identify it uniquely. Since every device connected to the internet needs an IP address, there was a concern about the limited number of IP addresses available in IPv4. To address this issue, the Internet Engineering Task Force (IETF) defined the new IPv6 standard [5].

4.1 IPv6 Header

IPv6 has a 40 bytes header as shown in Figure 6. Different fields of IPv6 header are described below.



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Traffic Class |           Flow Label                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Payload Length        |  Next Header  |   Hop Limit   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                         Source Address                        +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                      Destination Address                      +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: IPv6 Header

- **Version** 4-bit Internet Protocol version number = 6.
- **Traffic Class** 8-bit Traffic Class field.
- **Flow Label** 20-bit flow label.
- **Payload Length** 16-bit unsigned integer. Length of the IPv6 payload, i.e., the rest of the packet following this IPv6 header, in octets. Present are considered part of the payload, i.e., included in the length count.)
- **Next Header** 8-bit selector. Identifies the type of header immediately following the IPv6 header. Uses the same values as the IPv4 Protocol field.
- **Hop Limit** 8-bit unsigned integer. The Hop Limit is a field in the IPv6 header that is decreased by 1 at each node that forwards the packet as shown in Figure 6. If the Hop Limit value becomes zero during forwarding, the packet is dropped. However, when the packet reaches its destination, even if the Hop Limit value is zero, the destination node should process the packet normally instead of discarding it.
- **Source Address** 128-bit address of the originator of the packet.
- **Destination Address** This refers to the 128-bit destination address of the packet, which specifies the intended recipient of the packet. However, if there is a Routing header present in the packet, the ultimate recipient may be different from the intended recipient.

5 Fundamentals of SRV6

The following introduces fundamentals of SRv6 including types of SRv6 node, segment routing header, the composition of segment, functions of SRv6 and distribution of SID.

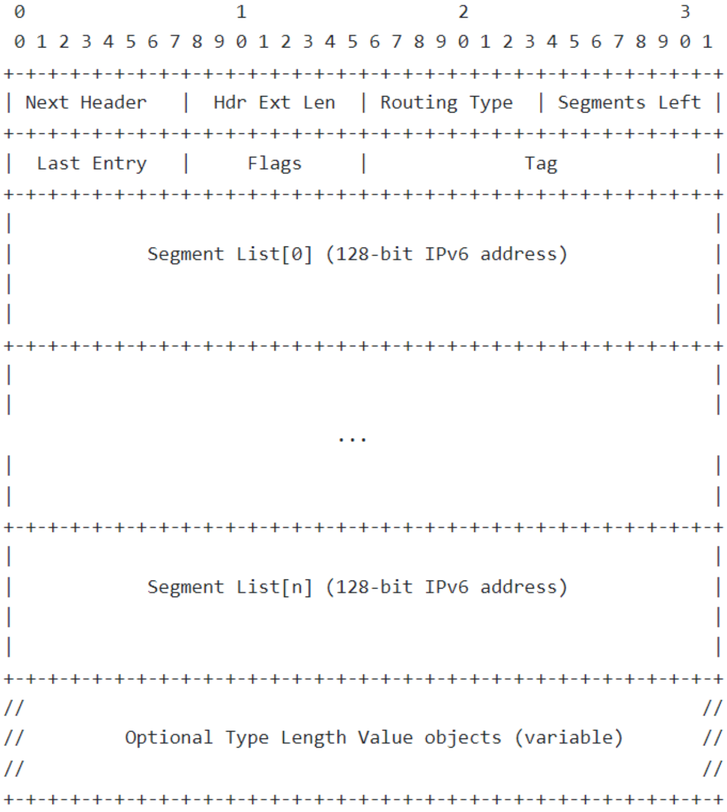
5.1 Types of SRV6 Node

There are three types of nodes in the SRv6 network [6]:

- SRv6 Source Node: any node that originates SRv6 packets.
- Transit Node: any node that forwards SRv6 packets but does not perform SRv6 processing.
- SRv6 Segment Endpoint Node: any node that receives and processes an SRv6 packet, where the destination address of the packet must be the locally configured SID or local interface address.

5.2 Segment Routing Header

In order to utilize SR using the IPv6 forwarding plane, a specific type of IPv6 RH is introduced shown in Figure 7, known as the Segment Routing Header (SRH) [6]. This RH is added to every IPv6 packet by the ingress and holds IPv6 path constraint details in the form of segment lists. These segment lists specify an explicit path for IPv6. As the packets traverse through the network, transit nodes use the path information included in the SRH to forward the packets.



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Header   |  Hdr Ext Len  | Routing Type  | Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Last Entry   |     Flags     |              Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Segment List[0] (128-bit IPv6 address)             |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                                                               |
                              ...
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Segment List[n] (128-bit IPv6 address)             |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                                                             //
//         Optional Type Length Value objects (variable)       //
//                                                             //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: SRH Header

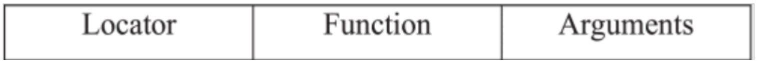
- **Next Header**: 8-bit selector. Identifies the type of headerimmediately following the IPv6 header. Usesthe same values as the IPv4 Protocol field.
- **Hdr Ext Len**: 8-bit unsigned integer. Length of the Routing header in 8-octet units, not including the first 8 octets.
- **Routing Type**: 8-bit identifier of a particular Routing header variant.
- **Segments Left**: 8-bit unsigned integer. Number of route segments remaining, i.e., number of explicitly listed intermediate nodes still to be visited before reaching the final destination.

- **Last Entry**: contains the index (zero based), in the Segment List, of the last element of the Segment List.
- **Flags**: 8 bits of flags.
- **Tag**: Tag a packet as part of a class or group of packets – e.g., packets sharing the same set of properties. When Tag is not used at the source, it MUST be set to zero on transmission. When Tag is not used during SRH processing, it SHOULD be ignored. Tag is not used when processing the SID defined in Section 4.3.1. It may be used when processing other SIDs that are not defined in this document. The allocation and use of tag is outside the scope of this document.
- **Segment List[0..n]:** 128-bit IPv6 addresses representing the nth segment in the Segment List. The Segment List is encoded starting from the last segment of the SR Policy. That is, the first element of the Segment List (Segment List[0]) contains the last segment of the SR Policy, the second element contains the penultimate segment of the SR Policy, and so on.
- **TLV: Type Length Value:** A TLV provides metadata for segment processing.

5.3 SRV6 Segment

SRv6 segment (also called SRv6 SID commonly) is a SRv6 network programming instruction with the format of 128-bit address [7]. It is consisted by of three elements as follows and shown in Figure 8:

- Locator: to route the packet to the node
- Function: to represent a behavior and conduct the behavior defined in the node
- Arguments: optional parameters for local function.



Locator	Function	Arguments

Figure 8: SRV6 Segment

5.4 SRH TLVs

TLV is a metadata unit used for segment processing. The support for TLV and HMAC is not mandatory for any implementation, but if an implementation adds or parses TLVs, it must support PAD TLVs. Other documents may define additional TLVs and processing rules for them. TLVs are included when the Hdr Ext Len is greater than **(Last Entry+1)*2.**

When processing TLVs at a segment endpoint, it is mandatory to ensure that TLVs are entirely within the SRH boundary as defined by the Hdr Ext Len. If TLVs exceed this boundary, it should trigger an ICMP Parameter Problem, Code 0, message to the Source Address, indicating the Hdr Ext Len field of the SRH, and result in packet discard.

An implementation is allowed to set a limit on the number and/or length of TLVs it processes, based on local configuration. It MAY limit:

- The number of consecutive Pad1 options to 1. If padding of more than one byte is required, then PadN should be used.
- The length in PadN to 5.
- The maximum number of non-Pad TLVs to be processed.
- The maximum length of all TLVs to be processed.

TLVs have their own length, format, and meaning. The assigned codepoint for each TLV type, allocated by IANA, specifies both the format and the meaning of the information contained in the TLV. Several TLVs may be encoded in the same SRH.

Bit 0 of the TLV type specifies whether the TLV data of that type can change while en route to the packet's final destination or not. An implementation can set limits on the number and/or length of TLVs it processes based on local configuration. If configured limits are exceeded, the implementation may stop processing additional TLVs in the SRH:

0: TLV data does not change en route

1: TLV data does change en route

TLVs in Segment Routing Header (SRH) have alignment requirements that are specified in the xn+y format, as defined in [8]. SR source nodes use these alignment requirements when constructing an SRH that includes TLVs and Padding TLVs. The Length field of a TLV is used to skip the TLV if a node does not support or recognize the TLV Type. The Length field indicates the length of the TLV in octets, not including the Type and Length fields.

5.5 Sid Format

SRv6 SID consist of **LOC:FUNCT:ARG**, The SID in SRv6 is composed of three parts: the locator (LOC), function (FUNCT), and arguments (ARG) as shown in Figure 9. The L most significant bits of the SID represent the LOC, followed by F bits of FUNCT and A bits of ARG. The length of the LOC, denoted as L, is flexible and can be chosen by the operator. The total length of L, F, and A must not exceed 128 bits. If L+F+A is less than 128 bits, the remaining bits of the SID must be set to zero.

- The Locator part of the SID contains the location information and is typically unique within an SRv6 domain, although it may be configured for multiple devices in certain cases such as anycast protection. Once a locator value is configured for a node, a locator route is generated and propagated throughout the SRv6 domain via an IGP. This allows other nodes on the network to locate the node based on the received route, and all SRv6 SIDs advertised by that node can be reached through the route.

- The ”Function” component of the SRv6 SID refers to an instruction that is pre-defined on the node that generates the SID. It is essentially an operation code (opcode) that instructs the node to perform a particular operation.

- The Arguments field, which is an optional part of the SRv6 SID, can be separated from the Function part. When expressed in the Locator:Function:Arguments format, this field is used to specify relevant information related to packet flow and services, occupying the least significant bits of the IPv6 address. Currently, a significant application of the Arguments field is to implement split horizon during forwarding of Broadcast, Unknown-unicast, and Multicast (BUM) traffic in Ethernet VPN (EVPN) Virtual Private LAN Service (VPLS) Customer Edge (CE) multi-homing scenarios.

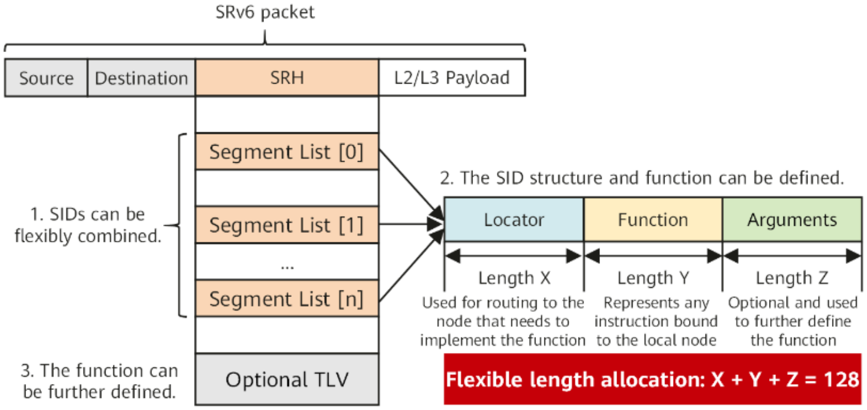
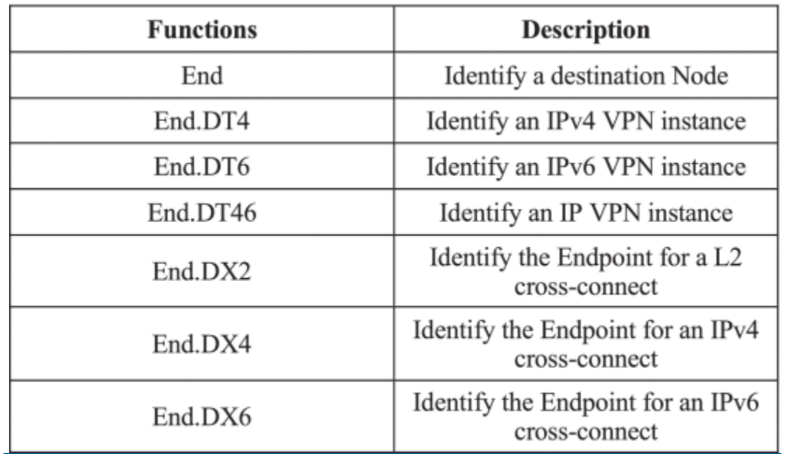

SRv6 packet
Source
Destination
SRH
L2/L3 Payload
Segment List [0]
Segment List [1]
...
Segment List [n]
Optional TLV
1. SIDs can be flexibly combined.
2. The SID structure and function can be defined.
3. The function can be further defined.
Locator
Function
Arguments
Length X
Length Y
Length Z
Used for routing to the node that needs to implement the function
Represents any instruction bound to the local node
Optional and used to further define the function
Flexible length allocation: X + Y + Z = 128

Figure 9: SID Structure

5.6 SRv6 Functions

There are several types of SRv6 operations. Different types of SRv6 functions represent different behaviors. Common SRv6 functions are shown in Figure 10. A detailed list of SRv6 functions is given in [7]



Functions	Description
End	Identify a destination Node
End.DT4	Identify an IPv4 VPN instance
End.DT6	Identify an IPv6 VPN instance
End.DT46	Identify an IP VPN instance
End.DX2	Identify the Endpoint for a L2 cross-connect
End.DX4	Identify the Endpoint for an IPv4 cross-connect
End.DX6	Identify the Endpoint for an IPv6 cross-connect

Figure 10: Functions of SRV6 SID

5.7 Distribution of SRV6 SID

SRv6 SID can be distributed by routing protocol or southbound API of SDN controller. For routing protocol, Boarder Gateway Protocol (BGP) can be extended to carry the SID between sender and receiver, please refer to Section 7 for more details. For southbound API of SDN controller, the SR path can be sent by gRPC, Restful API, NETCONF and OpenFlow [9].

6 SRv6 Network Programming

SRv6 offers powerful network programming capabilities as described in [7]. But how does it work on a network? The following section SRv6 implementation on a network from two perspectives: SRv6 packet forwarding process and SRv6 working mode.

6.1 Packet Forwarding Process

Figure 11 shows the scenario of forwarding a packet from host 1 to host 2 through several nodes. The nodes in the path, namely A, B, D, and E, support SRv6, while node C only supports IPv6. To achieve this forwarding, network programming is needed on node A. Specifically, the packet will first be sent to node A, and then it will be forwarded through links B-C and C-D before finally reaching node E and then host 2.

The packet forwarding process is as follows:

1. To forward the packet from node A to host 2 through nodes B, C, D, and E using SRv6, node A needs to encapsulate the SRv6 path information into an SRH. This information includes the SIDs of the B-C and C-D links, as well as SID A5::100 advertised by node E, and is encapsulated in reverse order. The packet's encapsulated header also includes an SL value of 2, indicating that there are three SIDs in total, and the segment list to be processed is "Segment List[2]". Node A then copies the value of Segment List to the DA field in the outer IPv6 header, searches the corresponding IPv6 routing table based on the longest match rule, and forwards the packet to node B.

2. Upon receiving the packet from node A, node B looks up its local SID table, which stores the SRv6 SID information generated by itself, to find a matching End.X SID based on the destination address in the outer IPv6 header. Following the instructions specified by the End.X SID, node B decrements the

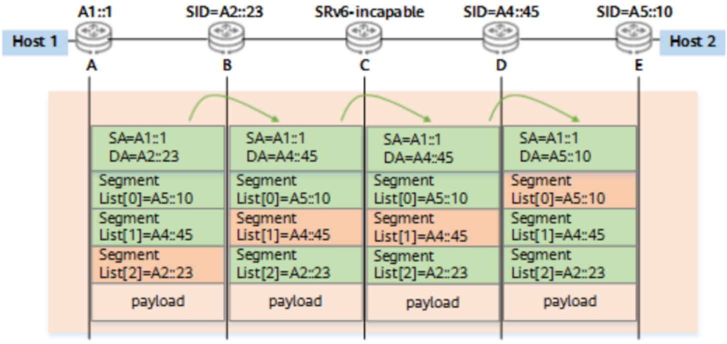


Figure 11: Packet forwarding process

SL value by 1, updates the DA field in the outer IPv6 header with the ”Segment List [1]” value, and then forwards the packet over the link (B-C) that is bound to the SID.

3. When the packet arrives at node C, it cannot recognize the SRH because it lacks any SRv6 capabilities. Therefore, it treats the packet as a regular IPv6 packet. Node C searches the corresponding IPv6 routing table based on the longest match rule and forwards the packet to node D, which is represented by the current destination address in the outer IPv6 header.

4. Upon receiving the packet forwarded by node C, node D searches its local SID table using the destination address A4::45 and finds a matching End.X SID. Node D follows the same procedure as node B: it decrements the SL value by 1, updates the DA field in the outer IPv6 header with A5::100, and forwards the packet over the link bound to the End.X SID.

5. When the packet arrives at node E, it searches its local SID table based on A5::100 and finds a matching End.DT4 SID. Following the instructions specified by the SID, node E decapsulates the packet by removing the IPv6 header. Node E then searches the IPv4 routing table of the VPN instance bound to the End.DT4 SID and forwards the inner IPv4 packet to host 2, ending the process.

6.2 SRv6 Working Mode

SRv6 can work in either SRv6 Traffic Engineering (TE) Policy or SRv6 Best Effort (BE) mode [10]. Both modes can be used to carry traditional services, such as L3VPN, EVPN L3VPN, EVPN VPLS, EVPN VPWS, and public IP services.

6.2.1 SRv6 TE Policy

Figure 12 shows SRv6 TE Policy that leverages SR’s source routing mechanism to instruct packet forwarding across a network based on an ordered list of segments (path information) encapsulated by the source node. As a result, SRv6 TE Policy can be used to implement traffic engineering, which improves network quality and meets E2E service requirements. When combined with SDN, SRv6 TE Policy is ideal for service-driven networks and is the recommended SRv6 working mode.

1. Through the use of BGP-LS, the forwarder (PE3) communicates network topology information to the controller. This information encompasses details about nodes and links, as well as traffic engineering attributes like link cost, bandwidth, and delay.

2. After gathering the topology information, the controller examines it and calculates routes that meet the service level agreement (SLA) requirements based on the service’s specifications.

3. Once the path information has been determined, the controller transmits it to the ingress node (PE1) of the network. The ingress node then creates SRv6 TE Policies, which consist of headend and destination addresses, as well as colors (extended community attributes).

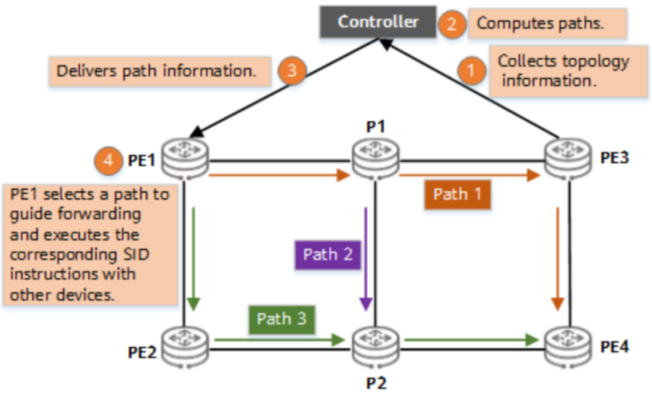

Controller
2 Computes paths.
3 Delivers path information.
1 Collects topology information.
4 PE1
PE1 selects a path to guide forwarding and executes the corresponding SID instructions with other devices.
P1
PE3
Path 1
Path 2
Path 3
PE2
P2
PE4

Figure 12: SRv6 TE Policy working process

4. The ingress node (PE1) chooses a suitable SRv6 TE Policy to direct service forwarding. As the service is forwarded, each forwarder in the network performs the instructions contained within their advertised Segment IDs (SIDs) based on the data carried by the SRv6 packets.

6.2.2 SRv6 BE Policy

SRv6 BE functions similarly to LDP in an MPLS network, whereby it utilizes the IGP's SPF algorithm to calculate the most efficient SRv6 path, employing only one Service SID for directing packet forwarding over links. As a best-effort operating mode, SRv6 BE does not possess traffic engineering capabilities and is mainly utilized for rapidly provisioning common VPN services.

To illustrate how SRv6 BE services are implemented, let's take the example of L3VPNv4 over SRv6 BE show in Figure 13. In this network setup, VPN instances are distributed across the network, with SRv6 deployed on both PE1 and PE2, while IPv6 is implemented on the P node. **Route advertisement phase:**

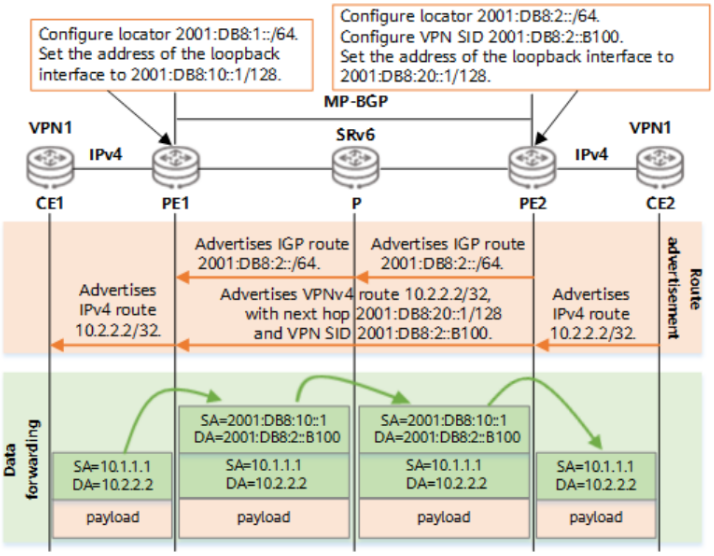

Configure locator 2001:DB8:1::/64.
Set the address of the loopback interface to 2001:DB8:10::1/128.
Configure locator 2001:DB8:2::/64.
Configure VPN SID 2001:DB8:2::B100.
Set the address of the loopback interface to 2001:DB8:20::1/128.
MP-BGP
VPN1
IPv4
SRv6
VPN1
IPv4
CE1
PE1
P
PE2
CE2
Advertises IGP route 2001:DB8:2::/64.
Advertises IGP route 2001:DB8:2::/64.
Advertises IPv4 route 10.2.2.2/32.
Advertises VPNv4 route 10.2.2.2/32, with next hop 2001:DB8:20::1/128 and VPN SID 2001:DB8:2::B100.
Advertises IPv4 route 10.2.2.2/32.
Route advertisement
Data forwarding
SA=10.1.1.1
DA=10.2.2.2
payload
SA=2001:DB8:10::1
DA=2001:DB8:2::B100
SA=10.1.1.1
DA=10.2.2.2
payload
SA=2001:DB8:10::1
DA=2001:DB8:2::B100
SA=10.1.1.1
DA=10.2.2.2
payload
SA=10.1.1.1
DA=10.2.2.2
payload

Figure 13: SRv6 BE working process

1. A locator is configured on PE2.
2. To inform PE1 about the SRv6 SID, PE2 employs an IGP to advertise the locator route 2001:DB8:2::/64. This route is then added to the IPv6 routing table of PE1.
3. When a VPN SID (2001:DB8:2::B100) is set up within the locator range on PE2, the router generates a local SID entry.
4. Once CE2 advertises an IPv4 route, PE2 transforms the route into a BGP VPNv4 route and propagates it to its MP-BGP peer, PE1. This route includes the SRv6 VPN SID, specifically the SID 2001:DB8:2::B100 assigned to the VPN instance.
5. After receiving the VPNv4 route, PE1 leaks the route to the routing table of the corresponding VPN instance, converts it into a common IPv4 route, and advertises it to CE1.

Data forwarding phase:

1. CE1 sends a common IPv4 packet to PE1.
2. Upon receiving the packet via the interface associated with the VPN instance, PE1 examines the VPN instance's routing table for a prefix entry that corresponds to the packet's destination IPv4 address. Once the relevant SRv6 VPN SID and next hop data are identified, PE1 encapsulates the packet into an IPv6 packet using the SRv6 VPN SID 2001:DB8:2::B100 as the destination address.
3. Using the longest match rule, PE1 locates the route 2001:DB8:2::/64 and directs the packet over the shortest available path to the P device.
4. Similarly, the P device applies the longest match rule to locate the route 2001:DB8:2::/64, and forwards the packet over the shortest available path to PE2.
5. PE2 queries its local SID table using 2001:DB8:2::B100. Once it locates the matching SID, PE2 performs the action associated with the SID, which entails removing the IPv6 packet header and examining the VPN instance's routing table for packet forwarding. At this stage, the packet is transformed back into a standard IPv4 packet.

7 SRv6 SID Distribution

7.1 SRv6 Implementation Through Protocol Extensions

To support SRv6, network nodes need to advertise the following two types of SRv6 information [10]:

1. Locator information is used by other nodes in the network to locate the node that advertises a particular SID, allowing them to execute the instruction associated with that SID. Typically, intra-area locator information is disseminated through IGP extensions.
2. SID information: A complete description of SIDs includes the functions and behaviors associated with them, such as their instructions. SIDs are classified into path SIDs and service SIDs, both of which are globally visible but locally effective. Path SIDs are primarily used to describe nodes or links and require propagation through IGP extensions, while service SIDs are closely related to routing information and are usually advertised through BGP extensions in BGP Update messages.

In conclusion, implementing basic SRv6 functions requires at least IGP and BGP extensions

7.2 IGP Extensions

7.2.1 IS-IS Extension

A link-state routing protocol works by using Dijkstra's Shortest Path First (SPF) algorithm to compute the shortest path to a specified address. This is done through a process where adjacent nodes establish neighbor relationships by exchanging Hello packets and flooding their local Link-State PDUs (LSPs) throughout the network to form an identical Link-State Database (LSDB). Using the LSDB, each node then runs the SPF algorithm to compute the shortest path to the desired address. Figure 14 shows an LSP carrying SRv6 information.

1. SRv6 Locator TLV:The Locator TLV is an important component in SRv6 that contains the locator's prefix and mask. It is used for advertising the locator information and allows other SRv6 nodes in the network to learn the locator route. Along with routing information, the TLV also carries SRv6 SIDs that do not require association with IS-IS neighbors, such as End SIDs.

2. Multi Topology Reachable IPv6 Prefixes TLV: The Multi Topology Reachable IPv6 Prefixes TLV and SRv6 Locator TLV both carry the same IPv6 prefix and mask as the locator information. While the SRv6 Locator TLV is specific to SRv6 nodes, the Multi Topology Reachable IPv6 Prefixes TLV can be processed by common IPv6 nodes as well. This allows common IPv6 nodes and SRv6 nodes to be deployed together on the same network. When a device receives both TLVs, the Multi Topology Reachable IPv6 Prefixes TLV takes precedence according to [11]. Both TLVs enable nodes on the network to generate a locator route to the node that advertises the corresponding locator and guide packet forwarding to that node.

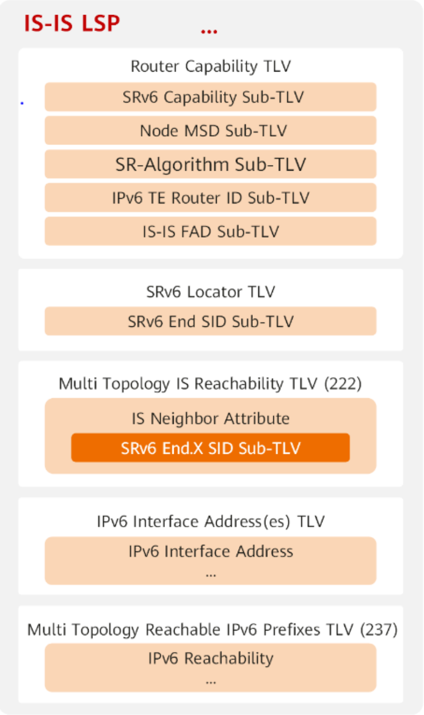


Figure 14: IS-IS LSP carrying SRv6 information

7.2.2 OSPF Extensions

OSPF extensions for MPLS data plane is already standardized in [12] While OSPF extensions for IPv6 data plane is still in working group document [13]. The extensions include advertisement of an OSPFv3 router's SRv6 capabilities, SRv6 Locators, and required SRv6 SIDs along with their supported endpoint behaviors.

7.3 BGP Extensions

The Segment Routing (SR) architecture is based on the source-routing paradigm, where a segment represents either a topological instruction, such as ”go to prefix P following shortest path”, or a service instruction. Each segment is identified by a Segment Identifier (SID), which is assigned in a single administrative domain called an ”SR domain”. An SR domain may consist of a single Autonomous System (AS) or multiple ASes under consolidated global SID administration. Typically, the ingress node of the SR domain adds an SR header containing SIDs to an incoming packet. The BGP extensions for SRv6 are defined in [14].

A BGP Prefix Segment is a BGP prefix with a Prefix-SID attached to it. The Prefix-SID is always a global SID within the SR domain and identifies an instruction to forward the packet over the Equal-Cost Multipath (ECMP) best path computed by BGP to the related prefix. The BGP Prefix-SID is the identifier of the BGP Prefix Segment.

A BGP attribute known as the **”BGP Prefix-SID attribute”** and specifies the rules to originate, receive, and handle error conditions for the attribute is defined here.The BGP Prefix-SID attribute defined here can be attached to prefixes from Multiprotocol BGP IPv4/IPv6 Labeled Unicast as shown in Figure 15.

It should be noted that: multiple Autonomous Systems (ASes) are interconnected and part of the same SR domain, the BGP Prefix-SID will be global across these ASes. However, if the ASes are not part of the same SR domain, the Autonomous System Border Routers (ASBRs) of each domain will need to handle the advertisement of unique SIDs. The mechanisms for interconnecting such ASes and handling the unique SIDs are not within the scope of the protocol extensions defined in the document. A BGP Prefix-SID MAY be attached to a BGP prefix. This implies that each prefix is advertised individually, reducing the ability to pack BGP advertisements (when sharing common attributes).

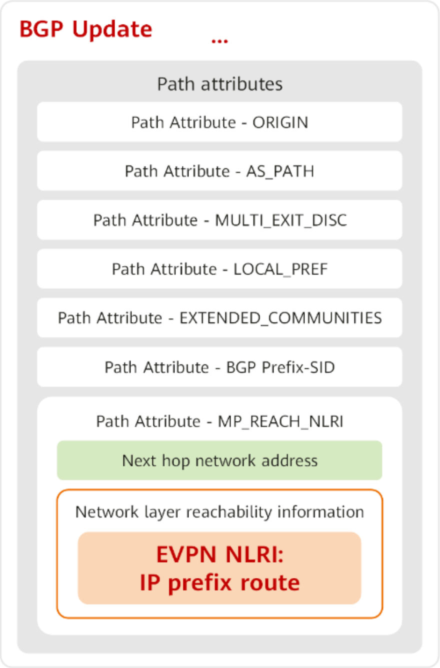

BGP Update
...
Path attributes
Path Attribute - ORIGIN
Path Attribute - AS_PATH
Path Attribute - MULTI_EXIT_DISC
Path Attribute - LOCAL_PREF
Path Attribute - EXTENDED_COMMUNITIES
Path Attribute - BGP Prefix-SID
Path Attribute - MP_REACH_NLRI
Next hop network address
Network layer reachability information
EVPN NLRI:
IP prefix route

Figure 15: BGP EVPN Update message carrying SRv6 information

8 SRV6 Use-Cases

8.1 NetworkAPI : An In-band Signalling Application-aware Traffic Engineering using SRv6 and IP anycast

Application-aware Traffic Engineering (TE) plays a crucial role in ensuring quality of services (QoS) for recently emerging applications such as AR, VR, cloud gaming, and connected vehicles. While a deterministic application-aware TE is required for these missioncritical applications, a negotiation procedure between applications and network operators needs to undergo major simplification to fulfill the scalability of the application based on emerging microservices and container-based architecture. In [reference to this paper] a NetworkAPI framework is presented which allows an application to indicate a desired TE behavior inside IP packets by leveraging Segment Routing over IPv6 (SRv6). In the NetworkAPI framework, the TE behavior provided by the network operator is expressed as an SRv6 Segment Identifier (SID) in the form of a 128-bit IPv6 address. Because the IPv6 address of an SRv6 SID is distributed using IP anycast, the application can utilize the unchanged SRv6 SID regardless of the application's location, as if the application controls an API on the transport network [15].

8.2 IPv6 Segment Routing in Enterprise Networks

Many entreprises are inspired by Software Defined Networks (SDN) which promise to simplify the management of their networks. Software Resolved Networks (SRN) [16] instantiate this SDN vision by using SRv6 in enterprise networks. Like SDNs, SRNs use a controller that manages the network resources. As in SDNs, the presence of the controller simplifies the management of the network and allows the operator to better control the available resources. However, there are several differences between SRNs and SDNs. First, SRNs leverage SRv6 in the dataplane and the SRH to control the flow of packets through the network. This reduces the amount of state required on the routers in contrast with Openflow-based SDNs. Second, applications can interact explicitely with the controller to indicate the requirements for their flows. The controller responds to these requirements by returning an SRH for a path that meets them. In SRNs, the controller is co-located with the entreprise DNS resolver and hosts use the DNS protocol to interact with the controller/resolver. When an application initiates a conversation, it performs the following operations. Figure 16 shows process of path selection.

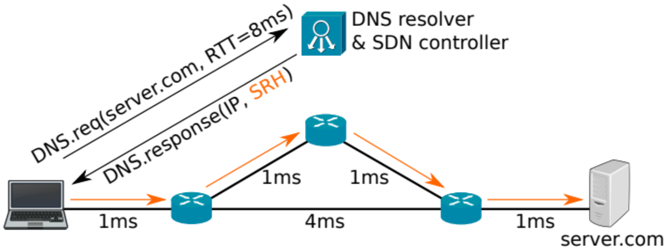


Figure 16: Illustration of path selection in SRN

First, it issues a DNS request to resolve the DNS server name and adds its requirements. Then, the controller chooses a network path that meets those requirements. The controller can use any optimization algorithm to select this path. Once the path is chosen, it is transformed into a list of SRv6 segments. Different path selection algorithms can be included in the SRN controller. The controller then sends back to the endhost a DNS response containing the server IPv6 address and the SRH corresponding to the selected path. Finally, the endhost attaches the SRH to each packet of the connection. Figure ? shows a Software Resolved Network. All its links have an IGP weight of 1. In a traditional IPv6 network, the application flows have to follow the shortest network path. In this example, the application wants an RTT of maximum 8ms and the shortest path has an RTT of 12ms. The controller selects the upper path and returns its SRH to the endhost.

8.3 Middle-boxes and IPv6 Segment Routing

IPv6 Segment Routing can also be used to support middleboxes. Middleboxes can perform two different types of network functions: per-packet (Network Address Translation, simple firewall) and per-bytestream (Intrusion Detection, Transocoding). Per-packet functions can be easily supported by using SRv6 [17] since they operate on the network or transport layer header. Per-bystream functions are more complex and often need to reconstruct the TCP bytestream and cope with reordering, losses, etc. Such functions need to reorder the received TCP packets which is similar to including a complete TCP implementation in each function. Instead of forcing each function to include a TCP stack, recently proposed SRv6 Pipes enable in-network bytestream functions with two main components: a transparent TCP proxy and a scheme to encode functions and parameters inside the SRH. Each middlebox uses a transparent TCP proxy to terminate the TCP connections before passing its bytestream to the network function. A middlebox can support different functions (e.g. transparent compression/decompression, IDS, etc.). Each of the supported functions is implemented as a dynamically loadable function that processes the bytestream exposed by the TCP proxy. The network uses the SRH to enforce the utilisation of specific network functions. This SRH can be inserted by the client , or by an access router. Since a given middlebox may support different functions, we need a way to specify the function that needs to process each packet on each middlebox. For this, we encode the functions and the parameters in the IPv6 address, as described in Figure 9. The SRH is thus used for two different purposes: (i) enforcing a specific path through selected middleboxes and (ii) indicating the network functions to be applied to each packet/flow.

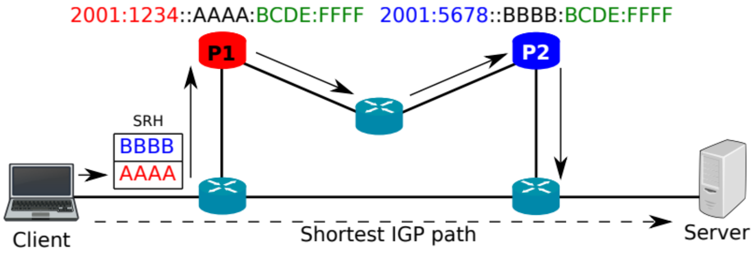


Figure 17: Illustration of path selection in SRN

Consider the network described in Figure 16 where the client requires to encrypt the traffic between P1 and P2. The client will use the function bits of the address of P1 to specify the identifier of the encrypt function (AAAA), and the parameters bits to specify the identifier (BCDE:FFFF) of an encryption key. The same will be done in P2's address with the decrypt function. While receiving the packets, the proxies will look at the function and parameters bits to pass them to the selected function.

We have mentioned only three use-cases here for the sake of brevity but there are variety of other use-cases worth mentioning e.g. Source Packet Routing in Networking (SPRING) [18], Network Slice Realization in Segment Routing Network [19], Segment Routing Traffic Engineering Leveraging Existing IPv6 [20], Topology Independent Fast Reroute using Segment Routing [21], Loop avoidance using Segment Routing [22], A Two-Way Active Measurement Protocol (TWAMP) [23], Path Tracing in SRv6 networks [24] and Operations, Administration, and Maintenance (OAM) in Segment Routing over IPv6 (SRv6) [25] etc.

9 SRv6 State of the Art

This section provides an overview of IPv6 Segment Routing (SRv6) deployment status. It lists various SRv6 features that have been deployed in the production networks.

9.1 SRv6 Deployments

SRv6 is currently deployed in more than ten different orgnizations. In this paper only few are listed for detailed list refer to [26]

9.1.1 Soft Bank

As part of the 5G rollout, Softbank deployed a nationwide SRv6 network. The following SRv6 features have been deployed:

- A Segment Routing Header [6] based data plane.
- END (PSP), END.X (PSP), END.DT4, H.Encaps.Red and H.Insert.Red functions as per [7], [I-D.filsfils-spring-srv6-net-pgm- insertion].
- ISIS SRv6 extensions [27].
- BGP VPN SRv6 extensions [28].
- SRH based Topology Independent (TI-LFA) Fast Reroute mechanisms using H.Insert.Red for the O(50msec) protection against node and link, as described in [29].
- BGP Prefix Independent Convergence (PIC) core and edge [30].

9.1.2 Iliad Italy

Iliad has implemented a nationwide SRv6 network in Italy as part of their 5G rollout. This network, based on Cisco NCS 5500, offers a comprehensive mobile IP solution. Iliad has developed their own SRv6 capable routers called Nodebox, which are deployed at each cell site. This deployment involves the interoperability of multiple SRv6 implementations, including NCS 5500 and Iliad's Nodebox.

- A Segment Routing Header [6] based data plane.
- End (PSP), End.X (PSP), End.DT4, END.DX2, H.Encaps.Red, H.Insert.Red, END.DT6 functions as per [7] , [31].
- BGP VPN SRv6 extensions [28].
- ISIS SRv6 extensions [27].
- SRH based Topology Independent (TI-LFA) Fast Reroute mechanisms using H.Insert.Red for the O(50msec) protection against node and link, as described in [29].
- Support for Ping and Traceroute as defined in [29].

9.1.3 China Telecom

A Segment Routing Header [6] based data plane.

- o END.DT4 function as per [7].
- BGP VPN SRv6 extensions [28]
- BGP Prefix Independent Convergence (PIC) core and edge [30].
- Support for Ping and Traceroute as defined in [25].

9.1.4 Bell Canada

Bell Canada has successfully deployed a nationwide SRv6 uSID network as part of their MEC (Multi-Access Edge Computing) rollout. They have achieved interoperability between Cisco, Arrcus, and Noviflow, incorporating these different vendors' solutions. In this deployment, SRv6 SIDs (Segment Identifiers) are allocated from the ULA (Unique Local Address) block [32].

- L3VPN services for IPv4 and IPv6 traffic with SRv6 uSIDs.

- SRv6 service programming [33] using SRv6 uSID with H.Encaps.Red and H.Insert.Red encapsulation as per [7], [31].
- SRv6 to MPLS interworking with End.DTM, End.DPM functions [34]
- ISIS SRv6 extensions [27].
- SRv6 BGP services extensions [28].
- SRv6 uSID TILFA for 50msec TILFA protection with uLoop avoidance [21]
- BGP Prefix Independent Convergence (PIC) core and edge [21]
- Support for Ping and Traceroute as defined in [28]

9.1.5 Alibaba

Alibaba has introduced their next-generation ”Predictable Network” that offers reliable network services to all applications. This is achieved through the utilization of full-stack SRv6 innovations across various endpoints such as containers, hosts, and P4 gateways, as well as network devices and the controller/network service center. With this approach, Alibaba ensures predictable and consistent network performance for every application.

9.2 Implementation Status

The following hardware and software platforms have either shipped or demonstrated support for SRv6, including [6] and [7]. This section also provides information about the supported SRv6 functions and transit behaviors in open-source software.

9.2.1 Open Source Implementation

- Linux kernel [35]
- Linux srext module
- FD.io VPP
- P4
- Zebra, an open-source implementation that succeeded the GNU Zebra and Quagga projects, offers support for SRv6. Zebra's SRv6 implementation includes all End functions, H.Insert, and H.Encaps. Additionally, the implementation provides support for FRR (Free Range Routing) for BGP Prefix-SID.

Apart from the mentioned routing platforms, several open-source applications have been enhanced to handle IPv6 packets with an SRH (Segment Routing Header). These extensions have been incorporated into the mainstream versions of Wireshark, tcpdump, iptables, and nftables, enabling them to support the processing of such packets.

- Wireshark [36]
- tcpdump [37]
- iptables [38] [39]
- nftables [40]
- Snort [41]
- SEgment Routing Aware firewall (SERA) [42]
- ExaBGP [43]

- Contiv-VPP [44]
- GoBGP [45]
- GoBMP [46]

9.2.2 Enterprise Implementations

As of now, SRv6 is supported by 25 publicly known hardware platforms from 10 different vendors. These hardware platforms, listed in alphabetical order, provide support for SRv6 according to [6] [7]

1. Arrcus: Arrcus offers support for SRv6, including BGP VPN extensions [28] and ISIS extensions [27], on the following hardware platforms:
    - Arrcus Quanta (IXAE, IXA) Broadcom Jericho2-based platforms with ArcOS EFT (early field trial) code.
    - Arrcus Edgecore (AS7926) Broadcom Jericho2-based platform with ArcOS EFT (early field trial) code.
2. Barefoot Networks:
    - Hardware implementation in the Tofino NPU is present since May 2017.
3. Broadcom:
    - Platforms have been shipping hardware implementations on the Jericho, Jericho+, Qumran AX, and Qumran MX NPUs since December 2018. Furthermore, Arrcus platforms offer hardware implementations on the Jericho2 NPU, which are currently available for early field trials.
4. Cisco:
    - Cisco ASR 9000 platform with IOS XR.
    - Cisco NCS 5500 platform with IOS XR.
    - Cisco NCS 560 platform with IOS XR.
    - Cisco NCS 540 platform with IOS XR.
    - Cisco ASR 1000 platform with IOS XE.
    - Cisco Nexus 9316D-GX platform with NX-OS.
    - Cisco 93600CD-GX platform with NX-OS.
    - Cisco 9364C-GX platform with NX-OS.
5. Huawei:
    - Huawei ATN with VRPV8.
    - Huawei CX600 with VRPV8.
    - Huawei NE40E with VRPV8.
    - Huawei ME60 with VRPV8.
    - Huawei NE5000E with VRPV8.
    - Huawei NE9000 with VRPV8.
    - Huawei NE8000 with VRPV8.
    - Huawei NG-OLT MA5800 with VRPV8.

10 Conclusion

In conclusion, SRv6 (Segment Routing over IPv6) emerges as a transformative networking paradigm that revolutionizes how we design, deploy, and manage networks. Throughout this white paper, we have explored the fundamental principles, benefits, and applications of SRv6, shedding light on its immense potential to address the evolving demands of modern network architectures.

By leveraging the inherent capabilities of IPv6 and segment routing, SRv6 offers unprecedented flexibility, scalability, and efficiency in routing and traffic engineering. Its ability to encapsulate and steer packets based on flexible and extensible segments empowers network operators to achieve fine-grained control over traffic paths, enabling dynamic service chaining, traffic optimization, and network slicing. One of the key advantages of SRv6 lies in its compatibility with existing network infrastructures. By leveraging the IPv6 data plane, SRv6 can be seamlessly integrated into legacy networks, allowing for a smooth transition without requiring costly upgrades or disruptive overhauls. This interoperability facilitates incremental adoption and fosters a path towards network evolution, enabling organizations to unlock the benefits of SRv6 while preserving their existing investments.

As we move into the future, SRv6 is poised to play a pivotal role in shaping the next generation of networking architectures. Its programmability, flexibility, and scalability open up new possibilities for innovation and service delivery, empowering organizations to meet the growing demands of digital transformation, IoT, edge computing, and emerging technologies.

In summary, SRv6 represents a significant step forward in network architecture, offering a powerful and versatile framework for building intelligent, scalable, and agile networks. By adopting SRv6, organizations can embrace the future of networking, unlock new opportunities, and ensure their networks are primed for the challenges and opportunities that lie ahead.

References

[1] "Juniper source routing document." https://www.juniper.net/documentation/us/en/software/junos/is-is/ospf/topics/concept/source-packet-routing.html.

[2] "Huawei article what is srv6." https://support.huawei.com/enterprise/en/doc/EDOC1100196195.

[3] "Segment routing official website." https://www.segment-routing.net/tutorials/2017-12-05-srv6-introduction/.

[4] C. Filsfils, S. Previdi, L. Ginsberg, B. Decraene, S. Litkowski, and R. Shakir, "Segment Routing Architecture." RFC 8402, July 2018.

[5] B. Hinden and D. S. E. Deering, "Internet Protocol, Version 6 (IPv6) Specification." RFC 2460, Dec. 1998.

[6] C. Filsfils, D. Dukes, S. Previdi, J. Leddy, S. Matsushima, and D. Voyer, "IPv6 Segment Routing Header (SRH)." RFC 8754, Mar. 2020.

[7] C. Filsfils, P. Camarillo, J. Leddy, D. Voyer, S. Matsushima, and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming." RFC 8986, Feb. 2021.

[8] D. S. E. Deering and B. Hinden, "Internet Protocol, Version 6 (IPv6) Specification." RFC 8200, July 2017.

[9] P. Ventre, M. M. Tajiki, S. Salsano, and C. Filsfils, "Sdn architecture and southbound apis for ipv6 segment routing enabled wide area networks," *IEEE Transactions on Network and Service Management*, 2018.

[10] "(ipv6 series ebook) srv6." https://support.huawei.com/enterprise/en/doc/EDOC1100200080.

[11] S. Previdi, L. Ginsberg, C. Filsfils, A. Bashandy, H. Gredler, and B. Decraene, "IS-IS Extensions for Segment Routing." RFC 8667, Dec. 2019.

[12] P. Psenak and S. Previdi, “OSPFv3 Extensions for Segment Routing MPLS Data Plane.” RFC 8666, Dec. 2019.

[13] Z. Li, Z. Hu, K. Talaulikar, and P. Psenak, “OSPFv3 Extensions for SRv6 IPv6 Data Plane,” Internet-Draft draft-ietf-lsr-ospfv3-srv6-extensions-11, Internet Engineering Task Force, May 2023. Work in Progress.

[14] S. Previdi, C. Filsfils, A. Lindem, A. Sreekantiah, and H. Gredler, “Segment Routing Prefix Segment Identifier Extensions for BGP.” RFC 8669, Dec. 2019.

[15] T. Miyasaka, Y. Hei, and T. Kitahara, “Networkapi: An in-band signalling application-aware traffic engineering using srv6 and ip anycast,” in *Proceedings of the Workshop on Network Application Integration/CoDesign*, NAI ’20, (New York, NY, USA), p. 8–13, Association for Computing Machinery, 2020.

[16] D. Lebrun, M. Jadin, F. Clad, C. Filsfils, and O. Bonaventure, “Software resolved networks: Rethinking enterprise networks with ipv6 segment routing,” in *Proceedings of the Symposium on SDN Research*, SOSR ’18, (New York, NY, USA), Association for Computing Machinery, 2018.

[17] F. Duchene, D. Lebrun, and O. Bonaventure, “Srv6pipes: enabling in-network bytestream functions,” in *2018 IFIP Networking Conference (IFIP Networking) and Workshops*, 2018.

[18] S. Previdi, C. Filsfils, B. Decraene, S. Litkowski, M. Horneffer, and R. Shakir, “Source Packet Routing in Networking (SPRING) Problem Statement and Requirements.” RFC 7855, May 2016.

[19] Z. Ali, C. Filsfils, P. Camarillo, D. Voyer, S. Matsushima, R. Rokui, A. Dhamija, and P. Maheshwari, “Building blocks for Network Slice Realization in Segment Routing Network,” Internet-Draft draft-ali-teas-spring-ns-building-blocks-03, Internet Engineering Task Force, Sept. 2022. Work in Progress.

[20] D. Dukes and C. Filsfils, “Segment Routing Traffic Engineering Leveraging Existing IPv6 Interface Addresses,” Internet-Draft draft-dukes-6man-sr-te-intf-address-00, Internet Engineering Task Force, June 2020. Work in Progress.

[21] S. Litkowski, A. Bashandy, C. Filsfils, P. Francois, B. Decraene, and D. Voyer, “Topology Independent Fast Reroute using Segment Routing,” Internet-Draft draft-ietf-rtgwg-segment-routing-ti-lfa-10, Internet Engineering Task Force, Apr. 2023. Work in Progress.

[22] A. Bashandy, C. Filsfils, S. Litkowski, B. Decraene, P. Francois, and P. Psenak, “Loop avoidance using Segment Routing,” Internet-Draft draft-bashandy-rtgwg-segment-routing-uloop-14, Internet Engineering Task Force, Dec. 2022. Work in Progress.

[23] J. Babiarz, R. M. Krzanowski, K. Hedayat, K. Yum, and A. Morton, “A Two-Way Active Measurement Protocol (TWAMP).” RFC 5357, Oct. 2008.

[24] C. Filsfils, A. Abdelsalam, P. Camarillo, M. Yufit, T. Graf, Y. Su, S. Matsushima, M. Valentine, and A. Dhamija, “Path Tracing in SRv6 networks,” Internet-Draft draft-filsfils-spring-path-tracing-03, Internet Engineering Task Force, Feb. 2023. Work in Progress.

[25] Z. Ali, C. Filsfils, S. Matsushima, D. Voyer, and M. Chen, “Operations, Administration, and Maintenance (OAM) in Segment Routing over IPv6 (SRv6).” RFC 9259, June 2022.

[26] S. Matsushima, C. Filsfils, Z. Ali, Z. Li, K. Rajaraman, and A. Dhamija, “SRv6 Implementation and Deployment Status,” Internet-Draft draft-matsushima-spring-srv6-deployment-status-15, Internet Engineering Task Force, Apr. 2022. Work in Progress.

[27] P. Psenak, C. Filsfils, A. Bashandy, B. Decraene, and Z. Hu, “IS-IS Extensions to Support Segment Routing over the IPv6 Data Plane.” RFC 9352, Feb. 2023.

[28] G. Dawra, K. Talaulikar, R. Raszuk, B. Decraene, S. Zhuang, and J. Rabadan, “BGP Overlay Services Based on Segment Routing over IPv6 (SRv6).” RFC 9252, July 2022.

[29] D. Voyer, C. Filsfils, D. Dukes, S. Matsushima, J. Leddy, Z. Li, and J. Guichard, “Deployments With Insertion of IPv6 Segment Routing Headers,” Internet-Draft draft-voyer-6man-extension-header-insertion-10, Internet Engineering Task Force, Nov. 2020. Work in Progress.

[30] A. Bashandy, C. Filsfils, and P. Mohapatra, “BGP Prefix Independent Convergence,” Internet-Draft draft-ietf-rtgwg-bgp-pic-19, Internet Engineering Task Force, Apr. 2023. Work in Progress.

[31] C. Filsfils, P. Camarillo, J. Leddy, D. Voyer, S. Matsushima, and Z. Li, “SRv6 NET-PGM extension: Insertion,” Internet-Draft draft-filsfils-spring-srv6-net-pgm-insertion-08, Internet Engineering Task Force, Feb. 2023. Work in Progress.

[32] B. Haberman and B. Hinden, “Unique Local IPv6 Unicast Addresses.” RFC 4193, Oct. 2005.

[33] F. Clad, X. Xu, C. Filsfils, D. Bernier, C. Li, B. Decraene, S. Ma, C. Yadlapalli, W. Henderickx, and S. Salsano, “Service Programming with Segment Routing,” Internet-Draft draft-ietf-spring-sr-service-programming-07, Internet Engineering Task Force, Feb. 2023. Work in Progress.

[34] S. Agrawal, Z. Ali, C. Filsfils, D. Voyer, G. Dawra, and Z. Li, “SRv6 and MPLS interworking,” Internet-Draft draft-agrawal-spring-srv6-mpls-interworking-11, Internet Engineering Task Force, Mar. 2023. Work in Progress.

[35] D. Lebrun and O. Bonaventure, “Implementing ipv6 segment routing in the linux kernel,” in *Proceedings of the Applied Networking Research Workshop*, ANRW ’17, (New York, NY, USA), p. 35–41, Association for Computing Machinery, 2017.

[36] “Add support for segment routing (type 4) extension header.” https://code.wireshark.org/review/gitweb?p=wireshark.git;a=commit;h=d6e9665872989c5f343fce47484 abe415d77486c.

[37] “Add support for ipv6 routing header type 4.” https://github.com/the-tcpdump-group/tcpdump/ commit/9c33608cb2fb6a64e1b76745efa530a63de08100.

[38] “add segment routing header ’srh’ match.” https://patchwork.ozlabs.org/patch/856578/.

[39] “add support for ’srh’.” https://patchwork.ozlabs.org/patch/859206/.

[40] “Adding support for segment routing header ’srh’.” http://patchwork.ozlabs.org/patch/879061/.

[41] “Ipv6 segment routing (srv6) aware snort.” https://github.com/SRouting/sr-snort.

[42] “Segment routing aware firewall (sera).” https://github.com/SRouting/SERA.

[43] “Exabgp to support bgp-prefix-sid for srv6-vpn.” https://github.com/Exa-Networks/exabgp/releases/tag/4.2.0.

[44] “”srv6 (segment routing on ipv6) implementation of k8s services.” https://github.com/contiv/vpp/blob/master/docs/setup/SRV6.md.

[45] “Srv6 extensions in gobgp (bgp implementation in go).” https://github.com/osrg/gobgp.

[46] “Srv6 extensions in bgp monitoring protocol (bmp)”.” https://github.com/sbezverk/gobmp.