SPRING Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 26, 2020

Z. Li
C. Li
Huawei Technologies
C. Xie
China Telecom
D. Voyer
Bell Canada
K. LEE
LG U+
H. Tian
F. Zhao
CAICT
J. Guichard
Futurewei Technologies
C. Li
China Telecom
S. Peng
Huawei Technologies
January 23, 2020

Compressed SRv6 Network Programming
draft-li-spring-compressed-srv6-np-01

Abstract

Segment Routing can be applied to the IPv6 data plane by leveraging a new type of Routing Extension Header, called Segment Routing Header (SRH). However, the overhead introduced by SRH may be a challenge for existing hardware, which may influence on the forwarding performance and the payload efficiency.

This document defines a compressed SRv6 network programming mechanism in order to reduce the overhead of SRv6 by introducing the Compressed Segment Identifier (C-SID) and the Compressed SRH (C-SRH). The C-SRH can be a new Routing Header or an enhancement of SRH.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Segment routing (SR) [RFC8402] is a source routing paradigm that explicitly indicates the forwarding path for packets at the ingress node by inserting an ordered list of instructions, called segments.

When segment routing is deployed on the IPv6 data plane, it is called SRv6 [I-D.ietf-6man-segment-routing-header].An SRv6 Segment ID (SID) is a 128-bit value, and it can be represented as LOC:FUNCT where LOC is the L most significant bits and FUNCT is the 128-L least significant bits [I-D.ietf-spring-srv6-network-programming]. L is called the locator length and is flexible. Each network operator is free to use the locator length it chooses. The LOC part of the SID is routable and leads to the node which instantiates that SID.

For support of SR, a new routing header called Segment Routing Header (SRH), which contains a list of SIDs and other information, has been defined in [I-D.ietf-6man-segment-routing-header]. In use cases like Traffic Engineering, an ordered SID List with multiple SIDs is inserted into the SRH to steer packets along an explicit path.

However, the overhead of SIDs (16 bytes per SID) may be a challenge for existing hardware processing, as the size of the SRH may affect the forwarding performance. When the packet is small, the payload efficiency is not ideal due to the large overhead of the SRH. When the packet is large, the overhead of the SRH may cause the packet to be dropped due to PMTU [RFC8200].

This document defines a compressed SRv6 network programming mechanism in order to reduce the overhead of SRv6 by introducing the Compressed Segment Identifier (C-SID) and the Compressed SRH (C-SRH). The C-SRH can be a new Routing Header or an enhancement of SRH, in either case, compatibility with the existing SRH is maintained.

2. Terminology

This document makes use of the terms defined in [I-D.ietf-6man-segment-routing-header], [RFC8402] and [RFC8200], and the reader is assumed to be familiar with that terminology. This document introduces the following new terms:

C-SRH: Compressed Segment Routing Header

C-SID: Compressed Segment Identifier

C-Tag: Compressed Tag

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Compressed SID (C-SID)

This document defines a Compressed SID (C-SID) to carry the last 16 - N bytes of the original SRv6 SID, where N is the length of the common prefix among SIDs in the SID list. N is calculated by comparing the difference of each SID with other SIDs on the SID list.

The common prefix part contains the common part of all Locators in the SID list, while the C-SID contains the different part, if any, of all Locators and the Function ID of an SRv6 SID. Generally, in an SRv6 domain, the common prefix can be the SRv6 SID block as per [I-D.ietf-spring-srv6-network-programming], and the C-SID consists of the Node ID and Function ID.

The IPv6 DA contains a 128-bits (16 Bytes) SRv6 SID, and it can be separated into two parts: the common prefix among all SIDs, and the current C-SID on the SID list.

```
  0                                        N                16 bytes
  +---------------------------------------------------------------+
  |                Common Prefix           |        C-SID         |
  +---------------------------------------------------------------+
  |<------------------ Locator -------------------->|<-FunctionID->|
                                           |<--->|
                                              |
                                 Different part of Locator
```

Figure 1. C-SID in IPv6 DA

In this way, the common prefix is carried by the IPv6 DA only, and the SIDs in the SID list will not carry the common prefix, but only the last 16 - N bytes of the original SRv6 SID.

Therefore, this document does not define any new SRv6 segment types.

Editor's Note: C-SID can be a fixed length value, such as 32 bits, if the WG can reach a consensus on it, and actually authors suggest this solution.

The C-SID is not needed to be the last N bits/Bytes of the SRv6 SID, it can be at any location in the SRv6 SID. In the best case, it follows the Common Prefix.

If the the length of Common prefix and C-SID is less than 128 bits, than padding is required. With the padding, C-SID does not need to be at the last part of SID, which will low down the compablities requirement of hardware.

In this case, operators can allocate a appropriate length common prefix for fitting their networks, and the fixed length of C-SID will be good for the hardware to process.

For example, the Common Prefix is A::/48, C-SID is a 32 bits value, and padding can be 48 bits zero, then only the 80 bits(Common prefix + C-SID) are used for routing, and only the C-SIDs should be carried in SRH and updated to DA, which is good for ASIC hardware.

In the same time, this format of SID can support the explicit match(Exact Match), which has better performance than LPM(Longes Prefix Match). But vendors can implemente LPM for C-SID, it is up to the vendor.

```
 0         Variable Length            32 bits               128 bits
 +---------------------------------------------------------------+
 |             Common Prefix   |  C-SID         | Padding        |
 +---------------------------------------------------------------+
 |<-------- Locator ---------------->|<-FuctID->|<---Padding--->|
                              |<--->|
                                 |
                      Different part of Locator
```

Figure 1. 32 bits C-SID in IPv6 DA

The authors would like to have the comments from the working group, to see which option is the best, so welcome to send your comments.

4. Compressed Segment Routing Header (C-SRH)

In order to carry the C-SID, this document defines the Compressed Segment Routing Header (C-SRH).

The C-SRH can be a new Routing Header (with new Routing Type (TBD)) or an enhancement of the SRH (Note: the latter is preferred in this document).

The C-SRH provides a more efficient encoding mechanism for SRv6, and is compatible with the existing SRv6 architecture.

```
 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   |E|      Flag   | C-Tag |           Tag          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Segment List[0](16 or 16 - C-Tag bytes)              |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Segment List[1](16 - C-Tag bytes)                |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Segment List[n](16 - C-Tag bytes)                |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Optional Padding to align with 64 bits Boundary  ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                                                             //
//         Optional Type Length Value objects (variable)       //
//                                                             //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2. Compressed Segment Routing Header

where:

o Next Header: Defined in [RFC8200]

o Hdr Ext Len: Defined in [RFC8200]

o Routing Type: 4 when C-SRH is an enhancement of SRH, or new type (TBD) when C-SRH is a new Routing Header.

o Segments Left: Defined in [RFC8200]

o Last Entry: contains the index (zero based), in the Segment List, of the last element of the Segment List.

o Flags:

```
0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|E|U U U U U U U|
+-+-+-+-+-+-+-+-+
```

U: Unused and for future use. MUST be 0 on transmission and ignored on receipt.

E: Exclude flag, set when the last SID is excluded in compression.

1: the last SID is excluded in compression, and it is a 16 bytes (128 bits) value

0: the last SID is included in compression, and it is a 16 - C-Tag bytes value

- o C-Tag: 4-bit unsigned integer to indicate the length of the common prefix. Therefore, the length of the C-SID in C-SRH is 16 - C-Tag bytes except the last segment if and only if the E-flag is set. When the C-Tag is 0, the length of C-SIDs in C-SRH is 16 bytes, which is compatible with the existing SRH [I-D.ietf-6man-segment-routing-header].

- o Tag: 12 bits value to tag a packet as part of a class or group of packets, e.g., packets sharing the same set of properties as per [I-D.ietf-6man-segment-routing-header].

- o Segment List[0]: 16 bytes (128 bits) IPv6 address when E-flag is set, and last 16 - C-Tag bytes of SID when E-flag is unset.

- o Segment List[n]: a compressed SRv6 SID, which is the last 16 - C-Tag bytes value of the original nth SRv6 SID. The Segment List is encoded starting from the last segment of the SR Policy, i.e., 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.

- o Type Length Value (TLV) are described in [I-D.ietf-6man-segment-routing-header].

In some use cases, the last SID may be a normal SID, which has a different prefix against all other SIDs, so it can be excluded in C-SID generation for better compression.

The E-flag indicates whether the last SID is excluded in compression. When E-flag is set, Segment List[0] will carry the original SID,

otherwise, it carries the compressed SID, i.e. the last 16 - C-Tag bytes of the original Segment List[0].

Padding is needed after the SID List[Last entry] to align with 64 bits.

Editor's Note: The authors had consided that there are some mechanisms to indicate compression, authors would like to have the comments from the working group, to see which option is the best, so welcome to send your comments.

1. Option 1: Compression Tag: C-tag in SRH

   * Compression tag indicates the length of Common prefix explicitly.

   * Indicate the length of C-SID as well, if the length of C-SID is a well-known length value.

   * No need to modify the control plane to distribute new type of SIDs.

   * SRH is modified, some bits are needed.

   * A SID can be used for normal SRv6 routing or Compression SRv6 routing.

   * New SID space for compression is optional.

2. Option 2: Compression Flag: C-flag in SRH

   * C-flag indicates compression, and the length of common prefix is learned from the control plane or configuration.

   * Indicate the length of C-SID as well, if the length of C-SID is a well-known length value.

   * Need to distribute the length of Common prefix or configure it in the SRv6 domain.

   * SRH is modified, a bit in flag for indicating compression is needed.

   * A SID can be used for normal SRv6 routing or Compression SRv6 routing.

   * New SID space for compression is optional.

3. Option 3: Compression Flavor/SID/Locator/SID Space: C Flavor/C Type in Control plane

   * New compression type of SIDs or SIDs with C flavor, they should be distributed via control plane, such as IGP.

   * The format of SID should be distributed via control plane, such as the length of common prefix.

   * SRH is not modified.

   * C-SID is used for compression SRv6 routing only.

   * New SID space for compression is required.

The format of C-SRH of Option 2 and 3 are shown below.

```
 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   |C|    Flag     |             Tag               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Segment List[0](N bytes)                    |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Segment List[1]   (N bytes)                 |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Segment List[n](N bytes)                    |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Optional Padding to align with 64 bits Boundary  ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                                                             //
//         Optional Type Length Value objects (variable)       //
//                                                             //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3. Compressed Segment Routing Header with C-flag

```
 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   |        Flag   |             Tag               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Segment List[0](N bytes)                    |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Segment List[1](N bytes)                    |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Segment List[N](N bytes)                    |
.                              ...                              .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Optional Padding to align with 64 bits Boundary  ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                                                             //
//         Optional Type Length Value objects (variable)       //
//                                                             //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2. C-SRH without changing Common header of SRH

5. C-SRH Processing

The compressed SID List can be generated by the ingress node by comparing the SIDs to get the C-Tag value according to the length of the common prefix. The compressed SID List can also be generated by a Controller and be sent to the ingress node, and the necessary protocol extensions for this are outside the scope of this document. (Note: The former is preferred in this document)

When the ingress node applies SRv6 policy to packets, a C-SRH can be encapsulated in a new IPv6 header (Encapsulation Mode). The first SID is carried along with the common prefix in the DA, and the remaining C-SIDs are carried in the SID List of the C-SRH. The last SID is inserted according to the E-flag.

When an SRv6 endpoint node receives the packet, the node will follow the same processing procedure as with an SRH, that is, to inspect whether the DA is a local SID or not, if yes, then process the SID according to its function. Otherwise, it will perform regular IPv6 forwarding.

When the DA is a local SID, then the node will process the C-SRH and the C-SIDs, and the current C-SID on the segment-list will replace the last 16 - C-Tag bytes of the IPv6 DA.

Regarding the last SID, if the E-flag is set, the entire 128 bits of Segment List[0] is updated to IPv6 DA. Otherwise, the C-SID will be updated to replace the last 16 - C-Tag bytes of IPv6 DA. After updating the IPv6 DA, the packet will be forwarded accordingly.

The pseudo code of C-SRH processing is shown below.

Editor's Note: The pseudo code will be updated when the options of Compression SRv6 NP are converged.

```
01. When a C-SRH is processed {
02.   If Segments Left is equal to zero {
03.     Proceed to process the next header in the packet,
        whose type is identified by the Next Header field in
        the Routing header.
04.   }
05.   Else {
06.     If local configuration requires TLV processing {
07.       Perform TLV processing
08        //If E-flag is unset:
09.       //  TLV begins at SID length + Padding Length
10.       //  SID Length = Last Entry * (16 - C-Tag)
11.       //  Padding Length = 8 - Last Entry * (16 - C-Tag)%8
12.       //Else:
          //  TLV begins at SID length + Padding Length
13.       //  SID Length = Last Entry * (16 - C-Tag) + C-Tag
14.     If  (Segments Left is greater than (Last Entry+1)) {
15.       Send an ICMP Parameter Problem, Code 0, message to
          the Source Address, pointing to the Segments Left
          field, and discard the packet.
16.     }
17.     Else {
18.       Decrement Segments Left by 1.
19.       if Segments Left > 0 or Segments Left = 0 and E-flag = 0:
20.         // Update the C-SID to the DA
21.         Copy Segment List[Segments Left] from the SRH to
            replace the last 16 - C-Tag bytes of
            destination address of the IPv6 header.
22.       else:
23.         // Segment Left = 0 and E-flag = 1
            // Segment List[0] is a 16 bytes value.
24.         Copy Segment List[Segments Left] from the SRH to
            destination address of the IPv6 header.
25.       If the IPv6 Hop Limit is less than or equal to 1 {
26.         Send an ICMP Time Exceeded -- Hop Limit Exceeded in
            Transit message to the Source Address and discard
            the packet.
27.       }
28.       Else {
29.         Decrement the Hop Limit by 1
30.         Resubmit the packet to the IPv6 module for
            transmission to the new destination.
31.       }
32.     }
33.   }
34. }
```

6. Control Plane Consideration

Editor's note: Control Plane consideration will be described in separate drafts in the future. Note that, some extensions may be not needed in some Compression options.

For indicating compression, the node should advertise the capabilities of SRv6 compression via control plane. A C-flag should be added in:

- o SRv6 Capabilities sub-TLV in IS-IS [I-D.ietf-lsr-isis-srv6-extensions]

- o SRv6-Capabilities TLV in OSPF [I-D.li-ospf-ospfv3-srv6-extensions],

- o OPEN Object/PATH-SETUP-TYPE-CAPABILITIES TLV/SRv6 Capabilities sub-TLV in PCEP [I-D.ietf-pce-segment-routing-ipv6]

For distributing the C-SID in control plane, the C-flag should be added to the following TLV or sub-TLV in IGP/BGP/BGP-LS and PCEP.

- o IS-IS [I-D.ietf-lsr-isis-srv6-extensions]:

  - * SRv6 Locator TLV, when C-flag is set, the Locator Block length is less than 96 if the C-SID is a 32 bits value.

  - * SRv6 END.X/ LAN END.X sub-TLV

- o OSPF [I-D.li-ospf-ospfv3-srv6-extensions]

  - * SRv6 Locator TLV, when C-flag is set, the Locator Block length is less than 96 if the C-SID is a 32 bits value.

  - * SRv6 END.X/ LAN END.X sub-TLV , when C-flag is set, the Locator Block length is less than 96 if the C-SID is a 32 bits value.

- o BGP [I-D.ietf-bess-srv6-services]

  - * SRv6 SID Information sub-TLV, when C-flag is set, the Locator Block length is less than 96 if the C-SID is a 32 bits value.

- o BGP-LS [I-D.ietf-idr-bgpls-srv6-ext]

  - * SRv6 Link Attributes: SRv6 END.X SID TLV/LAN END.X SID TLV, when C-flag is set, the Locator Block length is less than 96 if the C-SID is a 32 bits value.

* SRv6 Prefix Attributes: SRv6 Locator TLV, when C-flag is set, the Locator Block length is less than 96 if the C-SID is a 32 bits value.

* SRv6 SID Attributes: SRv6 Endpoint Function TLV, when C-flag is set, the Locator Block length is less than 96 if the C-SID is a 32 bits value.

* SRv6 SID Attributes: SRv6 BGP Peer Node SID TLV, when C-flag is set, the Locator Block length is less than 96 if the C-SID is a 32 bits value.

For distributing SRv6 Policy with compression SIDs, a C-flag should be added in BGP and PCEP.

o BGP SR Policy [I-D.ietf-idr-segment-routing-te-policy]

  * SRv6 Segment List sub-TLV, when C-flag is set, the Locator Block length is less than 96 if the C-SID is a 32 bits value. Candidate path and SR Policy level's extensions will be described in the future revision.

  * Segment sub-TLV, when C-flag is set, the Locator Block length is less than 96 if the C-SID is a 32 bits value. The entire SID is still carried in the SR Policy, while the C-flag will indicate this is a compression SID with C-SID.

    + Type 2: SRv6 SID

    + Type 9: IPv6 Node addr with opt SRv6 SID

    + Type 10: IPv6 addr+intf ID for Local and remote pair for SRv6 with opt SID

o PCEP

  * PATH-SETUP-TYPE TLV [RFC8408]: When PST is 2, the the C-flag indicates the SID list of the Path should be Compression SID.

  * SRv6 ERO Subobject [I-D.ietf-pce-segment-routing-ipv6]

  * SRv6 RRO Subobject [I-D.ietf-pce-segment-routing-ipv6]

7. Illustration

This section describes a simple example to illustrate the usage of C-SID. Similar to [I-D.filsfils-spring-srv6-net-pgm-illustration],

in order to ease the reading of the example, we introduce a simple reference diagram and simplified SID allocations.

Editor's note: the following part will be updated accordingly when the compression option is converged in WG.

7.1. Reference Diagram

Nodes 1 - 8 are SRv6 enabled nodes within the network domain.

Nodes CE1, CE2, and CE3 are outside the domain.

CE1 and CE2 are tenants of VPN 10.

Nodes 1 and 8 act as PE respectively to nodes CE1 and CE3.

All the links within the domain have the same IGP metric.

The IGP metric shortest-path from 1 to 8 is 1-2-7-8, while the latency-metric shortest-path from 1 to 8 is 1-2-3-4-5-6-7-8.

```
                      CE2
                         \
                          4------5
                          |      |
                    +-----3------6
                    |     |    / |
                    |     |  /   |
                    |     |/     |
       Tenant10  CE1---1-----2------7---8---CE3  Tenant10 with
                                                     IPv4 20/8
```

Figure 3: Reference topology

7.2. Compressed SRv6 Forwarding Example

This section describes a simple example to show how efficiently C-SRH can reduce the overhead of SRv6.

In order to ease the reading of the example, it is better to introduce a simplified SID allocation schema. We assume:

- o B::/112 is dedicated to the internal SRv6 SID space, which is the common prefix. Therefore, the C-SID is a 16-bit value.

- o A locator expressed in 120 bits and a function expressed in 8 bits.

- o Node k has B::0k/120 for its local SID space. Its SIDs will be explicitly allocated from that block.

- o Node k advertises B::0k/120 in its IGP.

- o Function 01 represents the End function with PSP support.

- o B::0k01 represents the End function with PSP support allocated by node K, such as B::0601 represents the End function with PSP support allocated by node 6.

- o B::0810 is an END.DT4 SID initiated by node 8, which is associated with the VRF10.

In SRH based SRv6, the PE 1 encapsulates the packets from CE1 to CE3 in an outer IPv6 header with DA = B::0201 and SRH (B::0810, B::0701, B::0601, B::0501, B::0401, B::0301, B::0201; SL=6; NH=4).

<B::0201, B::0301, B::0401, B::0501, B::0601, B::0701, B::0810> follows the latency-metric shortest-path. The total length of SRH is 8+16*7=120 bytes.

In Compressed SRv6, PE 1 encapsulates the packets from CE1 to CE3 in an outer IPv6 header with DA = B::0201 and C-SRH (0810, 0701, 0601, 0501, 0401, 0301, 0201, SL=6; NH=4) with E-flag unset. The C-Tag is 14, since the length of the common prefix is 112 bits. Therefore, the total length of C-SRH is 8 + (16-14)*7 = 22 bytes, reducing the encapsulation overhead by 98 bytes (81.7% less overhead than SRH) or 87.5% reduction in SIDs overhead.

The packet leaves node 1 to node 2 according to the FIB associated with the IPv6 DA B::0201. The packet can be presented as:

```
(A::1, B::0201)
(0810, 0701, 0601, 0501, 0401, 0301, 0201, SL=6; NH=4)
(CE1, CE3)
```

When 2 receives the packet, 2 matches B::0201 in its "My SID Table" and executes the END function behavior to update the IPv6 DA. Since the updated SL is greater than 0, and the C-Tag is 14, then it copies the C-SID that is a 2 bytes value to replace the last 2 bytes of the IPv6 DA, and then forwards the packet according to the new IPv6 DA B::0301. The packet can be presented as:

```
(A::1, B::0301)
(0810, 0701, 0601, 0501, 0401, 0301, 0201, SL=5; NH=4)
(CE1, CE3)
```

Like node 2, the nodes 3, 4, 5, and 6 perform the END function behavior to update the IPv6 DA with the corresponding C-SID and then forward the packet by looking up the IPv6 DA in their FIB accordingly. Therefore, the packet leaving node 6 can be presented as:

```
(A::1, B::0701)
(0810, 0701, 0601, 0501, 0401, 0301, 0201, SL=1; NH=4)
(CE1, CE3)
```

When 7 receives the packet, 7 matches B::0701 in its "My SID Table" and executes the END function behavior to update the IPv6 DA. Since the updated SL is 0 and E-flag is unset, then it copies the C-SID that is a 2 bytes value to replace the last 2 bytes of the IPv6 DA. Also, the C-SRH is popped since the B::0701 is an END SID with PSP flavor. Node 7 then performs a lookup on the updated IPv6 DA B::0810 to forward the packet along the shortest path to node 8. The packet can be presented as:

```
(A::1, B::0810)
(CE1, CE3)
```

When 8 receives the packet, 8 matches B::0810 in its "My SID Table" and executes the END.DT4 function behavior to sends the IP packet (CE1, CE3) to its VPN destination.

This example illustrates the procedure of C-SRH based SRv6 forwarding, and shows that the longer the common prefix, the more the SRv6 overhead can be reduced. More benefits are described in section 7.

8. Inter-domain Routing Using C-SRH

Considering privacy and security of SRv6 domain, when SRv6 is used for inter-domain routing, the detailed SIDs will not be leaked between domains, and the Binding SID [RFC8402] SHOULD be used. The typical inter-domain using SRv6 is illustrated in Figure 4.

When receiving the packet from CE1 to CE2, the Ingress node of SRv6 domain A can generate an SRv6 packet with SID List <BSID1, BSID2, VPN1>.

BSID1 is bound to an SR Policy, which contains a list of SID list in SRv6 Domain B, for example [B1::1, B2::1, B3::1, B4::1, B5::1].

Similarly, BSID2 is bound to an SR Policy in SRv6 Domian C, for example [C1::1, C2::1, C3::1, C4::1, C5::1].

VPN1 SID can be an END.DT4 SID associated with CE2.

In this way, the SIDs should be inserted at the ingress node are reduced from 11 to 3.

```
                            BSID1             BSID2      VPN1
          +---------+          +---------+       +---------+
          |         |          |         |       |         |
    CE1---*---------*----------*-*-*-*-*-*-*--------*-*-*-*-*-*-*---CE2
          |         |          |         |       |         |
          +---------+          +---------+       +---------+
         SRv6 Domain A        SRv6 Domain B      SRv6 Domain C

         (A1::1,BSID1)
         (VPN1,BSID2,BSID1)
         (CE1, CE2)
```

Figure 4.SRv6 Inter-domain Routing using BSID

Normally, when a BSID is processed , a new IPv6 and SRH will be added to the packet, and the SRH carries the SID list representing the sub-path of this domain. C-SRH can be used to carry Compressed SID list within the SRv6 domain for reducing the overhead of SRv6.

In this way, a Binding SID can be associated with an SR Policy, which contains a C-SID list to be carried by a C-SRH.

Therefore, in inter-domain SRv6 routing, C-SRH can be used in each domain, while the SRH is used for inter-domian. In addition, if the common prefixs of SIDs in SRH can be compressed, C-SRH can be used for carring these SIDs as well.

9. Benefits

1. Seamless integration with SRv6 Network Programming

    o No new type (Functions, such as END) of SRv6 SIDs is defined. A C-SID is a sub-set of an SRv6 SID.

    o Does not redefine the IPv6 address space nor requires any specific IPv6 space.

2. Support for full set of SRv6 functionalities

   o Full set of SRv6 functionality (BE, Loose TE and Strict TE, etc.) are supported without any extra route advertisements.

   o Function ID information is maintained.

3. Control-Plane friendly

   o No need to make any extensions in Control-Plane to advertise new type of SIDs or binding information.

   o No indexed mapping table is required

   o No routing extension is required.

   o No new route advertisement is required if without new Locators

4. Hardware-friendly

   o Hardware has the mature capability to overwrite the IPv6 DA.

   o Avoids any extra lookup in indexed mapping table

5. Efficient MTU overhead

   o C-SRH has the smallest MTU overhead among alternative solutions (VxLAN with SR-MPLS, CRH, uSID), when all the Segment endpoint nodes information is maintained in the packet.

6. Scalable SR TE

   o 8 C-SIDs can be carried in 128 bits when C-SID is 16 bits value

   o 16 Segment endpoint nodes (1 in DA and 16 in C-SRH including the one in DA) in 40 bytes of overhead.

      o T.Encaps with a C-SRH of 40 bytes (8 fixed + 2 * 16 bytes)

      o ALL C-SIDs are maintained in C-SRH, which can be used for recording the explicit routing path.

7. Saving IPv6 address

   o Very limited IPv6 address are needed for SID space. Longer Common Prefix means smaller IPv6 address burning and smaller overhead of SRv6.

   o Very easy to meet the requirement of C-SRH since any operator

or person can offer a 112/, 80/ or even 64/ prefix.

10. IANA Considerations

TBD

11. Security Considerations

TBD

12. Contributors

Zhibo Hu
Huawei Technologies
huzhibo@huawei.com

Zhongzheng Wang
Huawei Technologies
wangzhongzhen@huawei.com

Bing Liu
Huawei Technologies
remy.liubing@huawei.com

Yang Xia
Huawei Technologies
yolanda.xia@huawei.com

Jianwei Mao
Huawei Technologies
MaoJianwei@huawei.com

13. Acknowledgements

14. References

14.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[I-D.ietf-6man-segment-routing-header] Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-26 (work in progress), October 2019.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

[I-D.ietf-lsr-isis-srv6-extensions] Psenak, P., Filsfils, C., Bashandy, A., Decraene, B., and Z. Hu, "IS-IS Extension to Support Segment Routing over IPv6 Dataplane", draft-ietf-lsr-isis-srv6-extensions-04 (work in progress), January 2020.

[I-D.li-ospf-ospfv3-srv6-extensions] Li, Z., Hu, Z., Cheng, D., Talaulikar, K., and P. Psenak, "OSPFv3 Extensions for SRv6", draft-li-ospf-ospfv3-srv6-extensions-07 (work in progress), November 2019.

[I-D.ietf-pce-segment-routing-ipv6] Negi, M., Li, C., Sivabalan, S., Kaladharan, P., and Y. Zhu, "PCEP Extensions for Segment Routing leveraging the IPv6 data plane", draft-ietf-pce-segment-routing-ipv6-03 (work in progress), October 2019.

[I-D.ietf-idr-bgpls-srv6-ext] Dawra, G., Filsfils, C., Talaulikar, K., Chen, M., daniel.bernier@bell.ca, d., and B. Decraene, "BGP Link State Extensions for SRv6", draft-ietf-idr-bgpls-srv6-ext-02 (work in progress), January 2020.

[I-D.ietf-bess-srv6-services] Dawra, G., Filsfils, C., Raszuk, R., Decraene, B., Zhuang, S., and J. Rabadan, "SRv6 BGP based Overlay services", draft-ietf-bess-srv6-services-01 (work in progress), November 2019.

[I-D.ietf-idr-segment-routing-te-policy]
Previdi, S., Filsfils, C., Talaulikar, K., Mattes, P., Rosen, E., Jain, D., and S. Lin, "Advertising Segment Routing Policies in BGP", draft-ietf-idr-segment-routing-te-policy-08 (work in progress), November 2019.

[RFC8408] Sivabalan, S., Tantsura, J., Minei, I., Varga, R., and J. Hardwick, "Conveying Path Setup Type in PCE Communication Protocol (PCEP) Messages", RFC 8408, DOI 10.17487/RFC8408, July 2018, <https://www.rfc-editor.org/info/rfc8408>.

14.2. Informative References

[I-D.ietf-spring-srv6-network-programming]
Filsfils, C., Camarillo, P., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-ietf-spring-srv6-network-programming-08 (work in progress), January 2020.

[I-D.filsfils-spring-srv6-net-pgm-illustration]
Filsfils, C., Camarillo, P., Li, Z., Matsushima, S., Decraene, B., Steinberg, D., Lebrun, D., Raszuk, R., and J. Leddy, "Illustrations for SRv6 Network Programming", draft-filsfils-spring-srv6-net-pgm-illustration-01 (work in progress), August 2019.

Authors' Addresses

Zhenbin Li
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing 100095
China

Email: lizhenbin@huawei.com

Cheng Li
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing 100095
China

Email: chengli13@huawei.com

Chongfeng Xie
China Telecom
China Telecom Information Science&Technology Innovation park, Beiqijia Town,Changping District
Beijing
China

Email: xiechf.bri@chinatelecom.cn

Daniel Voyer
Bell Canada
China

Email: daniel.voyer@bell.ca

Kihoon LEE
LG U+
71, Magokjungang 8-ro, Gangseo-gu
Seoul
Republic of Korea

Email: soho8416@lguplus.co.kr

Hui Tian
CAICT
Beijing
China

Email: tianhui@caict.ac.cn

Feng Zhao
CAICT
Beijing
China

Email: zhaofeng@caict.ac.cn

James N Guichard
Futurewei Technologies
2330 Central Express Way
Santa Clara
USA

Email: james.n.guichard@huawei.com

Cong Li
China Telecom
China Telecom Information Science&Technology Innovation park, Beiqijia
Town,Changping District
Beijing
China

Email: licong.bri@chinatelecom.cn

Shuping Peng
Huawei Technologies
Huawei Campus, No. 156 Beiqing Rd.
Beijing 100095
China

Email: pengshuping@huawei.com