SPRING
Internet-Draft
Intended status: Experimental
Expires: 31 December 2020

K. Fang
Cisco Systems, Inc.
Y. Li
Google, Inc.
F. Cai
Cisco Systems, Inc.
29 June 2020

# Segment Routing over UDP(SRoU)
# draft-zartbot-sr-udp-00

Abstract

This document defines the Segment Routing Header[RFC8754] extension in UDP transport protocol with Network Address Translation Traversal.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 31 December 2020.

Table of Contents



## 1. Introduction

Many UDP based transport protocol(eg, IPSec/DTLS/QUIC) could provide a secure transportation layer to handle overlay traffic.

This document defines a new Segment Routing Header in UDP payload to enable segment routing over UDP(SRoU).

Segment Routing over UDP(SRoU) interworking with QUIC could provide a generic programmable and secure transport layer for next generation applications.

Discussion of this work is encouraged to happen on GitHub repository which contains the draft: https://github.com/zartbot/draft-quic-sr (https://github.com/zartbot/draft-quic-sr)

### 1.1. Specification of Requirements

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.

### 1.2. Motivation

Segment Routing provides source-based path enforcement and transportation level programmability but lacks of IPv4 support for transport over internet.

MPLS-over-UDP[RFC7510] and MPLS Segment Routing over IP[RFC8663] defined SR-MPLS over IPv4 network, but it lacks of NAT traversal capabilities.

Many SDWAN vendors defined their private protocols for routing control over multiple public cloud and internet, it's hard for interop with multi-vendors.

Many applications may require intelligence traffic steering(CDN/LB case), SRoU with QUIC could be used in these cases.

## 2. SR over UDP(SRoU) Packet encapsulation

The SRoU defined a generic segment routing enabled transport layer,the SR Header insert in UDP payload.

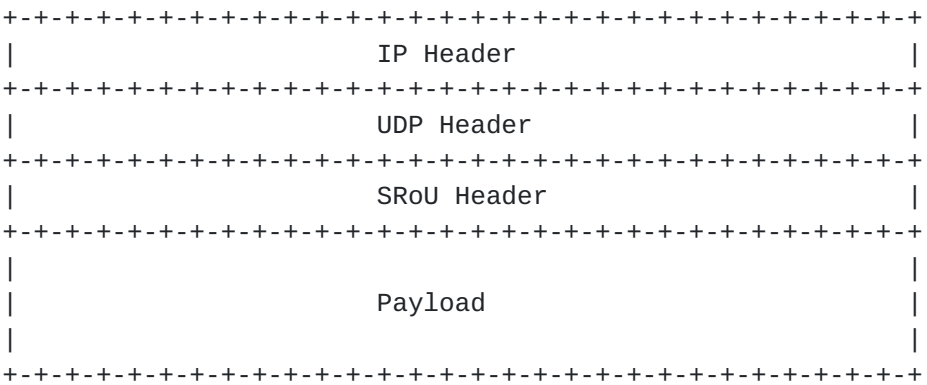



Figure 1: SRoU encapsulation

## 2.1. SR over UDP(SRoU) Header

SR over UDP must be present at the head of UDP payload.

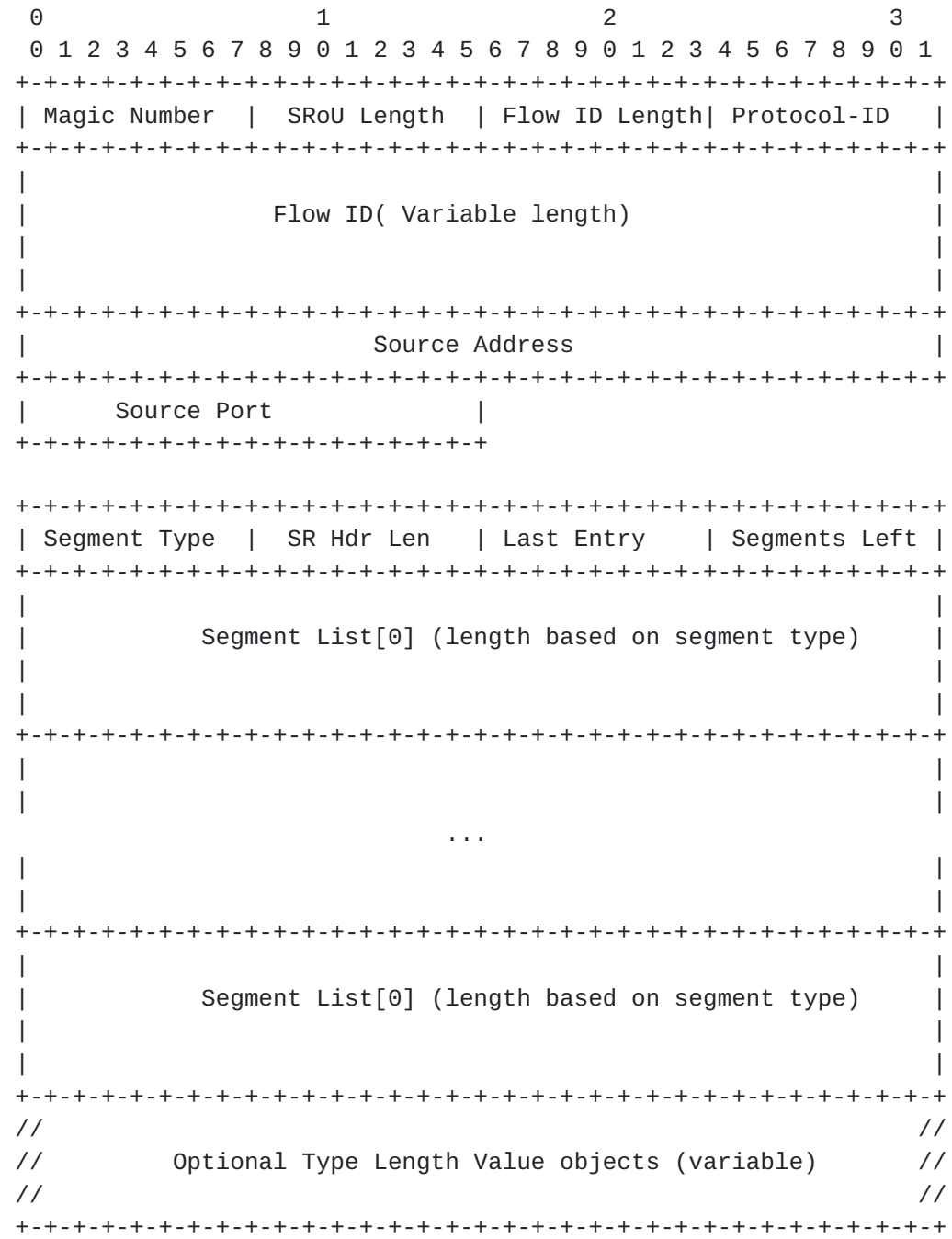



Figure 2: SRoU Header

Magic Number: 1 Byte field with ALL ZERO.

SRoU Length: 1 Byte, The byte length of a SRoU header.

FlowID Length: 1 Byte, The byte length of FlowID field.

Protocol-ID:

| Type | Name | Section |
|---|---|---|
| 0x0 | OAM | for Link state probe and other OAM |
| 0x1 | IPv4 | Indicate inner payload is IPv4 pkt |
| 0x2 | IPv6 | Indicate inner payload is IPv6 pkt |

Table 1: Protocol ID field

Source Address: Protocol-ID = 1, this field is 4-Bytes IPv4 address
Protocol-ID = 2, this field is 16-Bytes IPv6 address

Source Port: Source UDP Port Number

Segment Type:

| Type | Name | Len | Section |
|---|---|---|---|
| 0x0 | Reserved | | |
| 0x1 | IPv4 Address+ Port | 48b | Section 3.1 |
| 0x2 | SRv6 | 128b | Section 3.2 |
| 0x3 | Compressed Segment List | 128b | Section 3.3 |

Table 2: Segment Types

SR Hdr Len: SR Header length, include the SR Header flags Segment-List and Optional TLV.

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

Segments Left: 8-bit unsigned integer. Number of route segemnts remaining, i.e., number of explicitly listed intermediate nodes still to be visited before reaching the final destination.

Segment List[0..n]: 128-bit/48-bit/144-bit addresses to represent the SR Policy. Detailed forwarding behavior will be defined in Section 3

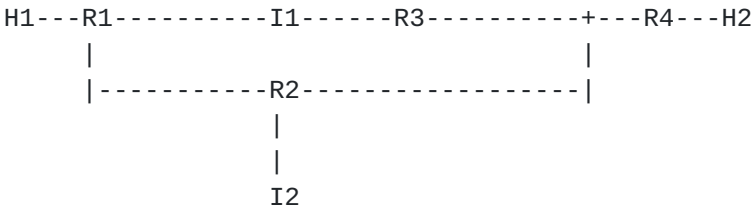
TLV: Opptional TLV used for future extension.currently only defined the following TLV.

| Type | Value | Len | Section |
|---|---|---|---|
| 0x0 | SR Integrity | 32b | Section 3.4.1 |
| 0x1 | Micro Segment Policy | variable | Section 3.4.2 |
| 0x2 | End.PacketInfo | variable | Section 3.4.3 |

Table 3: Optional TLV

## 3. Packet Processing

This section describe the packet proccessing procedure. The following topology will be used in this section.




I1,I2: Interim Node that support SRoU
R1~R4: Traditional Router
H1,H2: Host

Figure 3: Topology for packet proccesing

| Host | Address | SRoU Port | Post NAT |
|---|---|---|---|
| H1 | 192.168.1.2 | 5111 | 10.1.1.1:23456 |
| R1 | 192.168.1.1/10.1.1.1 | | |
| R2 | 10.1.2.2 | | |
| R3 | 10.1.3.3 | | |
| R4 | 10.1.4.4 | | |
| H2 | 10.99.2.2 | 443 | 10.1.4.4:443 |
| I1 | 10.99.1.1 | 8811 | |
| I2 | 192.168.99.2 | 8822 | 10.1.2.2:12345 |

Table 4: IP address table

## 3.1. Type:0x1, IPv4 Locator Mode

In this mode, the endpoint could directly insert the interim node IPv4 addresses and port into the segment-list.

For example, H1 intend to send packet to H2 via R1->I2--->H2, In this case SRoU packet will be NATed twice to show the NAT traversal workflow. I2's public address could use STUN[RFC5389] protocol detected and sync to all SRoU enabled devices.

H1 send packet with SRoU Header as below, H1 could use STUN detect it's source public address, but consider the simplicity, the 1st hop SRoU forwarder cloud update the source ip/port field in SRoU header.

```
IP/UDP Header {
  Source IP: 192.168.1.2,
  Destination IP: 10.1.2.2(SegmentList[1],I2 Pre-NAT public address),
  Source Port: 5111,
  Destination Port: 12345(SegmentList[1],I2 Pre-NAT public port),
}
SRoU Header {
  Magic Num = 0x0
  SRoU Length = 29
  FlowID Length = 0x3
  Protocol-ID = 0x1(IPv4),
  FlowID =  0x123,
  Source Address = 192.168.1.2,
  Source Port = 5111,
  Segment Left = 0x1,
  Last Entry   = 0x1,
  SegmenetList[0] = 10.1.4.4:443(H2),
  SegmenetList[1] = 10.1.2.2:12345(I2),
}
```

Figure 4: Type:0x1 H1-->I2 Packet Header

R1 is a NAT Device it will change the Source IP/Port to 10.1.1.1:23456. But this router may not have ALG function to modify SRoU Header.Then packet will send to 10.1.2.2:12345. It will be NAT again to I2.

After twice NAT, I2 Recieved packet as below:

```
IP/UDP Header {
  Source IP: 10.1.1.1(H1 post NAT addr),
  Destination IP: 192.168.99.2(I2 private addr),
  Source Port: 23456(H1 post NAT port),
  Destination Port: 8822(I2 private port),
}
SRoU Header {
  Magic Num = 0x0
  SRoU Length = 29
  FlowID Length = 0x3
  Protocol-ID = 0x1(IPv4),
  FlowID =  0x123,
  Source Address = 192.168.1.2,
  Source Port = 5111,
  Segment Left = 0x1,
  Last Entry   = 0x1,
  SegmenetList[0] = 10.1.4.4:443(H2),
  SegmenetList[1] = 10.1.2.2:12345(I2),
}
```

Figure 5: Type:0x1 H1-->I2, I2 Recieved Packet Header

if the (LastEntry == Segment Left) indicate I2 is the 1st hop SRoU forwarder, It MUST apply ALG to update the Source Address/Port field by the IP/UDP header. Then it will execute Segment Left - 1, and copy SegmentList[0] to DA/Dport. Consider some interim router like R2 has URPF checking, the SA/Sport will also updated to I2 SRoU socket address.

I2-->H2 packet:

```
IP/UDP Header {
  Source IP: 192.168.00.2(I2 Private),
  Destination IP: 10.1.4.4(SegmentList[0]),
  Source Port: 8822(I2 Private),
  Destination Port: 443(SegmentList[0]),
}
SRoU Header {
  Magic Num = 0x0
  SRoU Length = 29
  FlowID Length = 0x3
  Protocol-ID = 0x1(IPv4),
  FlowID =  0x123,
  Source Address = 10.1.1.1(update by I2 ALG),
  Source Port = 23456(update by I2 ALG),
  Segment Left = 0x0(SL--),
  Last Entry   = 0x1,
  SegmenetList[0] = 10.1.4.4:443(H2),
  SegmenetList[1] = 10.1.2.2:12345(I2),
}
```

Figure 6: Type:0x1 I2-->H2 Packet Header

H2 will recieve the packet, and if the segment left == 0, it MUST copy the Source Address and Port into IP/UDP Header and strip out the SRoU Header and send to udp socket. It may cache the reversed segmentlist for symmetric routing.

H2 send to UDP socket

```
IP/UDP Header {
  Source IP: 10.1.1.1(Copied from SRoU Src field),
  Destination IP: 10.99.2.2(Static NAT by R4),
  Source Port: 23456(Copied from SRoU Src field),
  Destination Port: 443(SegmentList[0]),
}
UDP Payload {
}
```

Figure 7: Type:0x1 H2 Send to UDP socket

## 3.2. Type:0x2, SRv6 format

IPv6 does not need to consider the NAT traversal case, In this mode almost forwarding action is same as SRv6. This is only used for application driven traffic steering(like CDN/LB usecase.). It has some benefit interworking with QUIC, the pure userspace implementation could provide additional flexibility.

For example some IOT sensor with legacy kernel stack does not support SRv6 could use SRoU insert SRH in UDP payload, the 1st hop SRoU forwarder could convert it to standard SRv6 packet.

## 3.3. Type:0x3, Compressed Segment List

### 3.3.1. Service Registration & Mapping

I1,I2 use SRoU port as source port to inital STUN[RFC5389] session to SR mapping server, the mapping server could detect the Post NAT address and assign SID for each host, and distribute IP/port-SID mapping database to all the SRoU enabled host.

| Host | Socket | SID |
|---|---|---|
| I1 | 10.99.1.1:8811 | 1111 |
| I2 | 10.1.2.2:12345 | 2222 |

Table 5: sid mapping

In this mode the socket information could combined with IPv4 and IPv6.

## 3.4. Optional TLV

### 3.4.1. SR Integrity TLV

SR Integrity Tag to validate the SRH. All fields in the SRH except Segments Left fields need to be checked.

### 3.4.2. Micro-segmentation(uSeg)

Option-TLV could defined Sub-TLV to support Micro-segmentation Security policy

```
  OptionTLV {
    0x1, uSeg{
        0x0, SRC_GROUP_ID,
        0x1, DST_GROUP_ID,
        0x2, APP_GROUP_ID,
        0x3, SRC_DEVICE_ID,
        0x4, DST_DEVICE_ID,
        0x5, APP_ID,
    }
  }
```

Customer also could encode this microsegment policy header in flowID field.

### 3.4.3. End.PacketInfo

This optional TLV defines extened packet info and Segment-end packet edit function. Sub-TLV defines as below:

#### 3.4.3.1. Type:0x0, VPN-ID

The SDWAN Router could use [I-D.ietf-quic-datagram] as VPN tunnel, This Sub-TLV defined the VPN-ID inside the tunnel.

If SRoU header has this sub-TLV, the device MUST decrypt inner payload and use the VPN-ID for inner packet destination lookup.

#### 3.4.3.2. Type:0x1, Orginal Destination Address/Port

In SR Type 0x3, The original destination address/port cloud not encode in 128bit field, it could be store in option TLV.

## 4. Usage

### 4.1. Traffic engineering over Internet

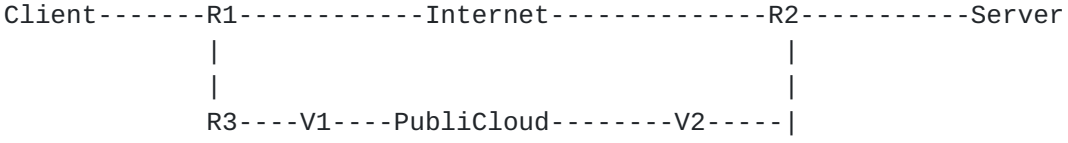



Figure 8: Traffic Engineering over internet

Many video/conferencing application requires traffic engineering over IPv4 Internet, Webex/Zoom/Teams may setup V1,V2 in public cloud, The client and server could encode the V1/V2 information in SRoU header for traffic engineering

### 4.2. Multipath forwarding

Same as previously topoloy Figure 8, customer cloud ask server transmit packet over different path, two path have same Flow-ID, QUIC could be used in this case to provide multistream/multihoming support.

### 4.3. Micro Segmentation

Same as previously topoloy Figure 8, the interim Router: R1/R2/R3, V1,V2 could insert uSeg Sub-TLV based on client and server uSeg identity, and other interim network equipment could based on this sub-TLV implement security policy or QoS policy.

### 4.4. Container Network

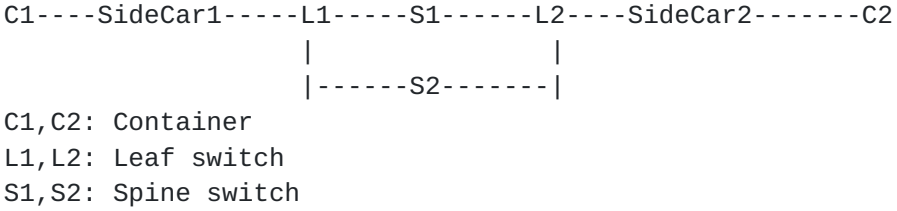



Figure 9: Service-Mesh & Container Network

SRoU with QUIC also could be used for container network interface, especially for service-mesh sidecar. The sidecar could aware the Datacenter underlay topology by BGP-LinkState, and use SRH select best path to avoid congestion. At the same time, all traffic are encrypted by [I-D.ietf-quic-tls].

### 4.5. MPLS-SR with SDWAN

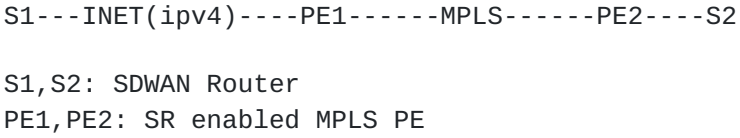



Figure 10: MPLS-SR with SDWAN

S1 will setup IPSec SA with S2 for end-to-end encryption, And it will use BSID between PE1-PE2 for traffic engineering.

MPLS based BSID and IPv4 based locator could be encoded in uSID.A distributed mapping table could be used to translate uSID to packet action.

```
IP/UDP Header {
  Source IP: H1,
  Destination IP: PE1,
  Source Port: srcport,
  Destination Port: IPSec,
}
SRoU Header {
  SegmentType = 0x1,
  SR_HDR_Len = 2,
  Last Entry = 0x0,
  Segment Left = 0,
  SegmenetList[0] = uSID: FC0:2222:3333:4444::
}
```

Figure 11: Type:0x1 S1-->PE1 Packet Header

## 4.6. Cloud Native Network platform

Each of the SRoU forwarder only rely on a UDP socket, it could be implement by a container. Customer could deploy such SRoU enable container in multiple cloud to provide a cloud-angonostic solution. All containers could be managed by K8S.

A distributed K-V store could be used for SRoU forwarder service registration, routing(announce prefix), all the SRoU forwarder could measue peer's reachability/jitter/loss and update link-state to the K-V store. forwarding policy also could be sync by the K-V store. Detailed information will be provided in another I.D(ETCD based disaggregated SDN control plane).

SRoU forwarder also could be implement by BPF for container communication. It will provide host level traffic engineering for massive scale datacenter to reduce the West-East traffic congestion.

The best practice for SRoU is working with QUIC. SRoU with QUIC transport protocol provides the following benefit for SDWAN :

* Stream multiplexing

* Stream and connection-level flow control

* Low-latency connection establishment

* Connection migration and resilience to NAT rebinding

* Authenticated and encrypted header and payload

SRoU add traffic-engineering and VPN capabilites for SDWAN. Many existing SDWAN features could gain the benefits like:

* TCP optimization

* Packet duplication

## 5. Security Considerations

The SRoU forwarder must validate the packet, FlowID could be used for source validation. It could be a token based solution, this token could be assigned by controller with a dedicated expire time. Source/Dest device ID and group cloud encode in flowid and signed by controller, just like JWT.

A blacklist on controller k-v store could be implemented to block device when the token does not expire.

## 6. IANA Considerations

### 6.1. SRoU with QUIC

The magic number in SRoU must be ZERO to distiguish with QUIC Long/ Short packet format.

Acknowledgements

The following people provided substantial contributions to this document:

* Bin Shi, Cisco Systems, Inc.

* Yijen Wang, Cisco Systems, Inc.

* Pix Xu, Cisco Systems, Inc.

* Xing James Jiang, Cisco Systems, Inc.

References

Normative References

[I-D.ietf-quic-datagram]
Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable Datagram Extension to QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-datagram-00, 26 February 2020, <http://www.ietf.org/internet-drafts/draft-ietf-quic-datagram-00.txt>.

[I-D.ietf-quic-tls]
Thomson, M. and S. Turner, "Using TLS to Secure QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-tls-29, 9 June 2020, <http://www.ietf.org/internet-drafts/draft-ietf-quic-tls-29.txt>.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI 10.17487/RFC5389, October 2008, <https://www.rfc-editor.org/info/rfc5389>.

[RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <https://www.rfc-editor.org/info/rfc7510>.

[RFC8663] Xu, X., Bryant, S., Farrel, A., Hassan, S., Henderickx, W., and Z. Li, "MPLS Segment Routing over IP", RFC 8663, DOI 10.17487/RFC8663, December 2019, <https://www.rfc-editor.org/info/rfc8663>.

[RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <https://www.rfc-editor.org/info/rfc8754>.

Informative 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>.

Authors' Addresses

Kevin Fang
Cisco Systems, Inc.

Email: zartbot.ietf@gmail.com

Yinghao Li
Google, Inc.

Email: liyinghao@gmail.com

Feng Cai
Cisco Systems, Inc.

Email: fecai@cisco.com