Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 21, 2018

S. Previdi
Individual
C. Filsfils, Ed.
Cisco Systems, Inc.
J. Leddy
Comcast
S. Matsushima
Softbank
D. Voyer, Ed.
Bell Canada

April 19, 2018

IPv6 Segment Routing Header (SRH)
draft-ietf-6man-segment-routing-header-12

Abstract

Segment Routing (SR) allows a node to steer a packet through a controlled set of instructions, called segments, by prepending an SR header to the packet. A segment can represent any instruction, topological or service-based. SR allows a node to steer a flow through any path (topological, or application/service based) while maintaining per-flow state only at the ingress node to the SR domain.

Segment Routing can be applied to the IPv6 data plane with the addition of a new type of Routing Extension Header. This document describes the Segment Routing Extension Header Type and how it is used by SR capable nodes.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

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 October 21, 2018.

Table of Contents

1. Segment Routing Documents

Segment Routing terminology is defined in [I-D.ietf-spring-segment-routing].

The network programming paradigm [I-D.filsfils-spring-srv6-network-programming] defines additional functions associated to a Segment Routing for IPv6 (SRv6) Segment ID (SID).

Segment Routing use cases are described in [RFC7855] and [RFC8354].

Segment Routing protocol extensions are defined in [I-D.ietf-isis-segment-routing-extensions], and [I-D.ietf-ospf-ospfv3-segment-routing-extensions].

2. Introduction

Segment Routing (SR), defined in [I-D.ietf-spring-segment-routing], allows a node to steer a packet through a controlled set of instructions, called segments, by prepending an SR header to the packet. A segment can represent any instruction, topological or service-based. SR allows a node to steer a flow through any path (topological or service/application based) while maintaining per-flow

state only at the ingress node to the SR domain. Segments can be derived from different components: IGP, BGP, Services, Contexts, Locators, etc. The list of segment forming the path is called the Segment List and is encoded in the packet header.

SR allows the use of strict and loose source based routing paradigms without requiring any additional signaling protocols in the infrastructure hence delivering an excellent scalability property.

The source based routing model described in [I-D.ietf-spring-segment-routing] inherits from the ones proposed by [RFC1940] and [RFC8200]. The source based routing model offers the support for explicit routing capability.

2.1. Data Planes supporting Segment Routing

Segment Routing (SR), can be instantiated over MPLS (SRMPLS)([I-D.ietf-spring-segment-routing-mpls]) and IPv6 (SRv6). This document defines its instantiation over the IPv6 data-plane based on the use-cases defined in [RFC8354].

This document defines a new type of Routing Header (originally defined in [RFC8200]) called the Segment Routing Header (SRH) in order to convey the Segment List in the packet header as defined in [I-D.ietf-spring-segment-routing]. Mechanisms through which segments are known and advertised are outside the scope of this document.

2.2. SRv6 Segment

An SRv6 Segment is a 128-bit value. "SRv6 SID" or simply "SID" are often used as a shorter reference for "SRv6 Segment".

An SRv6-capable node N maintains a "My Local SID Table". This table contains all the local SRv6 segments explicitly instantiated at node N. N is the parent node for these SIDs.

A local SID of N could be an IPv6 address of a local interface of N but it does not have to be. Most often, a local SID is not an address of a local interface of N. The My Local SID Table is thus not populated by default with all the addresses of a node.

Every SRv6 local SID instantiated has a specific instruction bounded to it.

This information is stored in the My Local SID Table. The My Local SID Table has three main purposes:

o Define which local SIDs are explicitly instantiated.

- Specify which instruction is bound to each of the instantiated SIDs.

- Store the parameters associated to such instruction (i.e. OIF, NextHop,...).

A Segment ID (SID) in the destination address of an IPv6 header, is routed through an IPv6 network as an IPv6 address. SIDs, or the prefix(es) covering SIDs in the My Local SID Table, and their reachability may be distributed by means outside the scope of this document. For example, [RFC5308] or [RFC5340] may be used to advertise a prefix covering the My Local SID Table.

A node may receive a packet with an SRv6 SID in the DA without an SRH. In such case the packet should still be processed by the Segment Routing engine.

2.3. Segment Routing (SR) Domain

The Segment Routing Domain (SR Domain) is defined as the set of nodes participating in the source based routing model. These nodes may be connected to the same physical infrastructure (e.g.: a Service Provider's network).

The SRH is added to the packet by its source, consistent with the source routing model defined in [RFC8200]. For example:

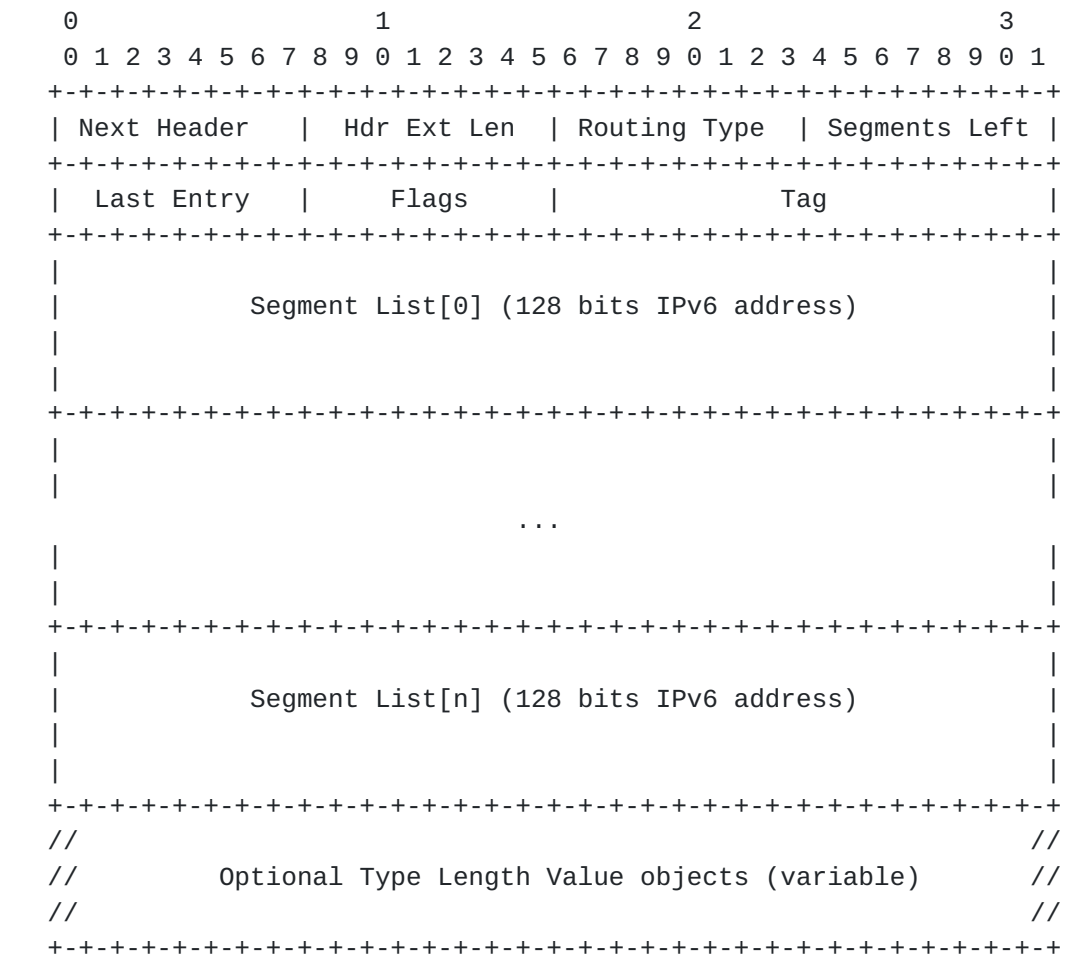
- At the node originating the packet (host, server).

- At the ingress node of an SR domain where the ingress node receives a packet and encapsulates it in an outer IPv6 header followed by a Segment Routing header.

3. Segment Routing Extension Header (SRH)

A new type of the Routing Header (originally defined in [RFC8200]) is defined: the Segment Routing Header (SRH) which has a new Routing Type, (suggested value 4) to be assigned by IANA.

The Segment Routing Header (SRH) is defined as follows:



where:

- Next Header: Defined in [RFC8200]

- Hdr Ext Len: Defined in [RFC8200]

- Routing Type: TBD, to be assigned by IANA (suggested value: 4).

- Segments Left: Defined in [RFC8200]. See Section 3.2 for detailed usage during SRH processing.

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

- Flags: 8 bits of flags. Following flags are defined:

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

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

H-flag: HMAC flag. If set, the HMAC TLV is present and is encoded as the last TLV of the SRH. In other words, the last 36 octets of the SRH represent the HMAC information. See Section 3.1.2 for details on the HMAC TLV.

- 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 source it SHOULD be set to zero on transmission. When tag is not used during SRH Processing it MUST be ignored. The allocation and use of tag is outside the scope of this document.

- Segment List[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 path. I.e., the first element of the segment list (Segment List [0]) contains the last segment of the path, the second element contains the penultimate segment of the path and so on.

- Type Length Value (TLV) are described in Section 3.1.

3.1. SRH TLVs

This section defines TLVs of the Segment Routing Header.

```
 0                   1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-----------------------
|     Type      |    Length     | Variable length data
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-----------------------
```

Type: An 8 bit value. Unrecognized Types MUST be ignored on receipt.

Length: The length of the Variable length data. It is RECOMMENDED that the total length of new TLVs be multiple of 8 bytes to avoid the use of Padding TLVs.

Variable length data: Length bytes of data that is specific to the Type.

Type Length Value (TLV) contain optional information that may be used by the node identified in the DA of the packet. The information carried in the TLVs is not intended to be used by the routing layer. Typically, TLVs carry information that is consumed by other components (e.g.: OAM) than the routing function.

Each TLV has its own length, format and semantic. The code-point allocated (by IANA) to each TLV Type defines both the format and the semantic of the information carried in the TLV. Multiple TLVs may be encoded in the same SRH.

TLVs may change en route at each segment. To identify when a TLV type may change en route the most significant bit of the Type has the following significance:

0: TLV data does not change en route

1: TLV data does change en route

Identifying which TLVs change en route, without having to understand the Type, is required for Authentication Header Integrity Check Value (ICV) computation. Any TLV that changes en route is considered mutable for the purpose of ICV computation, the Type Length and Variable Length Data is ignored for the purpose of ICV Computation as defined in [RFC4302].

The "Length" field of the TLV is used to skip the TLV while inspecting the SRH in case the node doesn't support or recognize the Type. The "Length" defines the TLV length in octets, not including the "Type" and "Length" fields.

The following TLVs are defined in this document:

Padding TLV

HMAC TLV

Additional TLVs may be defined in the future.

3.1.1. Padding TLVs

There are two types of padding TLVs, pad0 and padN, the following applies to both:

Padding TLVs are optional and more than one Padding TLV MUST NOT appear in the SRH.

The Padding TLVs are used to align the SRH total length on the 8 octet boundary.

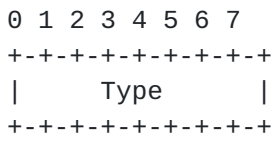
When present, a single Pad0 or PadN TLV MUST appear as the last TLV before the HMAC TLV (if HMAC TLV is present).

When present, a PadN TLV MUST have a length from 0 to 5 in order to align the SRH total length on a 8-octet boundary.

Padding TLVs are ignored by a node processing the SRH TLV, even if more than one is present.

Padding TLVs are ignored during ICV calculation.

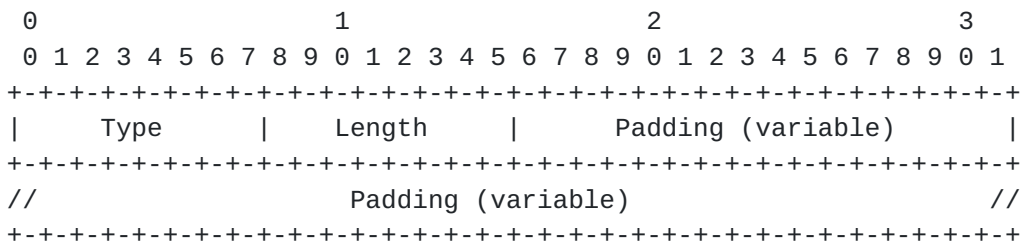
3.1.1.1. PAD0



Type: to be assigned by IANA (Suggested value 128)

A single Pad0 TLV MUST be used when a single byte of padding is required. If more than one byte of padding is required a Pad0 TLV MUST NOT be used, the PadN TLV MUST be used.

3.1.1.2. PADN



Type: to be assigned by IANA (suggested value 129).

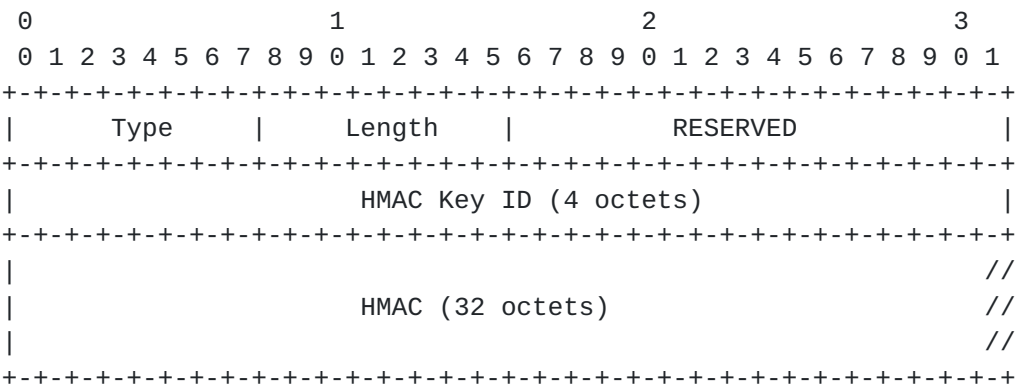
Length: 0 to 5

Padding: Length octets of padding. Padding bits have no semantics. They SHOULD be set to 0 on transmission and MUST be ignored on receipt.

The PadN TLV MUST be used when more than one byte of padding is required.

3.1.2. HMAC TLV

HMAC TLV is optional and contains the HMAC information. The HMAC TLV has the following format:



where:

- Type: to be assigned by IANA (suggested value 5).
- Length: 38.
- RESERVED: 2 octets. SHOULD be unset on transmission and MUST be ignored on receipt.
- HMAC Key ID: 4 octets.
- HMAC: 32 octets.
- HMAC and HMAC Key ID usage is described in Section 6

The Following applies to the HMAC TLV:

- When present, the HMAC TLV MUST be encoded as the last TLV of the SRH.
- If the HMAC TLV is present, the SRH H-Flag (Figure 2) MUST be set. Nodes processing SRH SHOULD process the HMAC TLV only when the H-Flag is set, and local policy.
- When the H-flag is set in the SRH, the router inspecting the SRH MUST find the HMAC TLV in the last 38 octets of the SRH.

3.2. SRH Processing

Extension header processing is as defined in RFC8200 for packets destined to a local IPv6 address. If SRH is present, it is processed as follows (DA represents the destination address in the IPv6 header):

1. IF the DA is in My Local SID Table
2. The function associated with the DA is processed.
   (Examples are as described in section 4).
3. ELSE
4. Treat the extension header as unrecognized
   and process as defined in RFC8200 section 4.4.

4. SRH Functions

Segment Routing architecture, as defined in [I-D.ietf-spring-segment-routing] defines a segment as an instruction or, more generally, a set of instructions (function).

In this section we review two functions that may be associated to a segment:

- End: the endpoint (End) function is the base of the source routing paradigm. It consists of updating the DA with the next segment and forward the packet accordingly.

- End.X: The endpoint layer-3 cross-connect function.

Other functions may be defined in other documents.

4.1. Endpoint Function (End)

The Endpoint function (End) is the most basic function.

When the endpoint node receives a packet destined to DA "S" and S is an entry in the My Local SID Table and the function associated with S in that entry is "End" then the endpoint does:

1. IF SegmentsLeft > 0 THEN
2. decrement SL
3. update the IPv6 DA with SRH[SL]
4. FIB lookup on updated DA
5. forward accordingly to the matched entry
6. ELSE IF SID is assigned to an interface
7. proceed to process the next header
8. ELSE
9. drop the packet

It has to be noted that:

- The End function performs the FIB lookup in the forwarding table of the ingress interface.

- An SRv6-capable node MUST include the FlowLabel of the IPv6 header in its hash computation for ECMP load-balancing as described in Section 5.4.

4.2. End.X: Endpoint with Layer-3 cross-connect

When the endpoint node receives a packet destined to DA "S" and S is an entry in the My Local SID Table and the function associated with S in that entry is "End.X" then the endpoint does:

```
1. IF SegmentsLeft > 0 THEN
2.    decrement SL
3.    update the IPv6 DA with SRH[SL]
4.    forward to layer-3 adjacency bound to the SID "S"
5.  ELSE
6.    drop the packet
```

It has to be noted that:

- If an array of layer-3 adjacencies is bound to the End.X SID, one entry of the array is selected based on a hash of the packet's header as described in Section 5.4

- An End.X function does not allow decaps. An End.X SID cannot be the last SID of an SRH and cannot be the DA of a packet without SRH.

- The End.X function is required to express an explicit path through an IP network.

- The End.X function is the SRv6 instantiation of a Adjacency SID as defined in [I-D.ietf-spring-segment-routing].

- Note that with SR-MPLS ([I-D.ietf-spring-segment-routing-mpls]), an AdjSID is typically preceded by a PrefixSID. This is unlikely in SRv6, most likely an End.X SID is globally routed address of N.

- If a node N has 30 outgoing interfaces to 30 neighbors, usually the operator would explicitly instantiate 30 End.X SIDs at N: one per layer-3 adjacency to a neighbor. Potentially, more End.X could be explicitly defined (groups of layer-3 adjacencies to the same neighbor or to different neighbors).

- If node N has a bundle outgoing interface I to a neighbor Q made of 10 member links, N may allocate up to 11 End.X local SIDs for that bundle: one for the bundle itself and then up to one for each member link. This is the equivalent of the L2-Link Adj SID in SR-MPLS ([I-D.ietf-isis-l2bundles]).

5. SR Nodes

There are different types of nodes that may be involved in segment routing networks: source nodes that originate packets with an SRH, transit nodes that forward packets having an SRH and segment endpoint nodes that MUST process the SRH.

5.1. Source SR Node

A Source SR Node can be any node originating an IPv6 packet with its IPv6 and Segment Routing Headers. This include either:

A host originating an IPv6 packet.

An SR domain ingress router encapsulating a received packet in an outer IPv6 header followed by an SRH.

The mechanism through which a Segment List is derived is outside the scope of this document. As an example, the Segment List may be obtained through:

Local path computation.

Local configuration.

Interaction with a centralized controller delivering the path.

Any other mechanism.

The following are the steps of the creation of the SRH:

Next Header and Hdr Ext Len fields are set as specified in [RFC8200].

Routing Type field is set as TBD (to be allocated by IANA, suggested value 4).

The DA of the packet is set with the value of the first segment.

The first element of the segment list is the last segment (the final DA of the packet). The second element is the penultimate segment and so on.

In other words, the Segment List is encoded in the reverse order of the path.

The Segments Left field is set to n-1 where n is the number of elements in the Segment List.

The Last_Entry field is set to n-1 where n is the number of elements in the Segment List.

HMAC TLV may be set according to Section 6.

If the segment list contains a single segment and there is no need for information in flag or TLV, then the SRH MAY be omitted.

The packet is sent towards the packet's DA (the first segment).

5.2. Transit Node

As specified in [RFC8200], the only node who is allowed to inspect the Routing Extension Header (and therefore the SRH), is the node corresponding to the DA of the packet. Any other transit node MUST NOT inspect the underneath routing header and MUST forward the packet towards the DA and according to the IPv6 routing table.

As a generic security measure, any service provider will block any packet destined to one of its internal routers, especially if these packets have an extension header in it.

5.3. SR Segment Endpoint Node

The SR segment endpoint node is the node whose My Local SID Table contains an entry for the DA of the packet.

The segment endpoint node executes the function bound to the SID as per the matched My Local SID entry (Section 4).

5.4. Load Balancing and ECMP

Within an SR domain, an SR source node encapsulates a packet in an outer IPv6 header for transport to an endpoint. The SR source node MUST impose a Flow Label computed based on the inner packet. The computation of the flow label is as recommended in [RFC6438] for the sending TEP.

At any transit node within an SR domain, the flow label MUST be used as defined in [RFC6438] to calculate the ECMP hash toward the destination address.

At an SR segment endpoint node, the flow label MUST be used as defined in [RFC6438] to calculate any ECMP hash used to forward the processed packet to the next segment.

6. Security Considerations

This section analyzes the security threat model, the security issues and proposed solutions related to the new Segment Routing Header.

The Segment Routing Header (SRH) is simply another type of the routing header as described in [RFC8200] and is:

- Added by an SR edge router when entering the segment routing domain or by the originating host itself. The source host can even be outside the SR domain;

- inspected and acted upon when reaching the destination address of the IP header per [RFC8200].

Per [RFC8200], routers on the path that simply forward an IPv6 packet (i.e. the IPv6 destination address is none of theirs) will never inspect and process the content of the SRH. Routers whose one interface IPv6 address equals the destination address field of the IPv6 packet MUST parse the SRH and, if supported and if the local configuration allows it, MUST act accordingly to the SRH content.

As specified in [RFC8200], the default behavior of a non SR-capable router upon receipt of an IPv6 packet with SRH destined to an address of its, is to:

- ignore the SRH completely if the Segment Left field is 0 and proceed to process the next header in the IPv6 packet;

- discard the IPv6 packet if Segment Left field is greater than 0, it MAY send a Parameter Problem ICMP message back to the Source Address.

6.1. Threat model

6.1.1. Source routing threats

Using an SRH is similar to source routing, therefore it has some well-known security issues as described in [RFC4942] section 2.1.1 and [RFC5095]:

- amplification attacks: where a packet could be forged in such a way to cause looping among a set of SR-enabled routers causing

unnecessary traffic, hence a Denial of Service (DoS) against bandwidth;

- reflection attack: where a hacker could force an intermediate node to appear as the immediate attacker, hence hiding the real attacker from naive forensic;

- bypass attack: where an intermediate node could be used as a stepping stone (for example in a De-Militarized Zone) to attack another host (for example in the datacenter or any back-end server).

6.1.2. Applicability of RFC 5095 to SRH

First of all, the reader must remember this specific part of section 1 of [RFC5095], "A side effect is that this also eliminates benign RH0 use-cases; however, such applications may be facilitated by future Routing Header specifications.". In short, it is not forbidden to create new secure type of Routing Header; for example, [RFC6554] (RPL) also creates a new Routing Header type for a specific application confined in a single network.

In the segment routing architecture described in [I-D.ietf-spring-segment-routing] there are basically two kinds of nodes (routers and hosts):

- nodes within the SR domain, which is within one single administrative domain, i.e., where all nodes are trusted anyway else the damage caused by those nodes could be worse than amplification attacks: traffic interception, man-in-the-middle attacks, more server DoS by dropping packets, and so on.

- nodes outside of the SR domain, which is outside of the administrative segment routing domain hence they cannot be trusted because there is no physical security for those nodes, i.e., they can be replaced by hostile nodes or can be coerced in wrong behaviors.

The main use case for SR consists of the single administrative domain where only trusted nodes with SR enabled and configured participate in SR: this is the same model as in [RFC6554]. All non-trusted nodes do not participate as either SR processing is not enabled by default or because they only process SRH from nodes within their domain.

Moreover, all SR nodes ignore SRH created by outsiders based on topology information (received on a peering or internal interface) or on presence and validity of the HMAC field. Therefore, if intermediate nodes ONLY act on valid and authorized SRH (such as

within a single administrative domain), then there is no security threat similar to RH-0. Hence, the [RFC5095] attacks are not applicable.

6.1.3. Service stealing threat

Segment routing is used for added value services, there is also a need to prevent non-participating nodes to use those services; this is called 'service stealing prevention'.

6.1.4. Topology disclosure

The SRH may also contains IPv6 addresses of some intermediate SR-nodes in the path towards the destination, this obviously reveals those addresses to the potentially hostile attackers if those attackers are able to intercept packets containing SRH. On the other hand, if the attacker can do a traceroute whose probes will be forwarded along the SR path, then there is little learned by intercepting the SRH itself.

6.1.5. ICMP Generation

Per Section 4.4 of [RFC8200], when destination nodes (i.e. where the destination address is one of theirs) receive a Routing Header with unsupported Routing Type, the required behavior is:

- If Segments Left is zero, the node must ignore the Routing header and proceed to process the next header in the packet.

- If Segments Left is non-zero, the node must discard the packet and send an ICMP Parameter Problem, Code 0, message to the packet's Source Address, pointing to the unrecognized Routing Type.

This required behavior could be used by an attacker to force the generation of ICMP message by any node. The attacker could send packets with SRH (with Segment Left not set to 0) destined to a node not supporting SRH. Per [RFC8200], the destination node could generate an ICMP message, causing a local CPU utilization and if the source of the offending packet with SRH was spoofed could lead to a reflection attack without any amplification.

It must be noted that this is a required behavior for any unsupported Routing Type and not limited to SRH packets. So, it is not specific to SRH and the usual rate limiting for ICMP generation is required anyway for any IPv6 implementation and has been implemented and deployed for many years.

6.2. Security fields in SRH

This section summarizes the use of specific fields in the SRH. They are based on a key-hashed message authentication code (HMAC).

The security-related fields in the SRH are instantiated by the HMAC TLV, containing:

- HMAC Key-id, 32 bits wide;
- HMAC, 256 bits wide (optional, exists only if HMAC Key-id is not 0).

The HMAC field is the output of the HMAC computation (per [RFC2104]) using a pre-shared key identified by HMAC Key-id and of the text which consists of the concatenation of:

- the source IPv6 address;
- Last Entry field;
- an octet of bit flags;
- HMAC Key-id;
- all addresses in the Segment List.

The purpose of the HMAC TLV is to verify the validity, the integrity and the authorization of the SRH itself. If an outsider of the SR domain does not have access to a current pre-shared secret, then it cannot compute the right HMAC field and the first SR router on the path processing the SRH and configured to check the validity of the HMAC will simply reject the packet.

The HMAC TLV is located at the end of the SRH simply because only the router on the ingress of the SR domain needs to process it, then all other SR nodes can ignore it (based on local policy) because they trust the upstream router. This is to speed up forwarding operations because SR routers which do not validate the SRH do not need to parse the SRH until the end.

The HMAC Key-id field allows for the simultaneous existence of several hash algorithms (SHA-256, SHA3-256 ... or future ones) as well as pre-shared keys. The HMAC Key-id field is opaque, i.e., it has neither syntax nor semantic except as an index to the right combination of pre-shared key and hash algorithm and except that a value of 0 means that there is no HMAC field. Having an HMAC Key-id field allows for pre-shared key roll-over when two pre-shared keys

are supported for a while when all SR nodes converged to a fresher pre-shared key. It could also allow for interoperation among different SR domains if allowed by local policy and assuming a collision-free HMAC Key Id allocation.

When a specific SRH is linked to a time-related service (such as turbo-QoS for a 1-hour period) where the DA, Segment ID (SID) are identical, then it is important to refresh the shared-secret frequently as the HMAC validity period expires only when the HMAC Key-id and its associated shared-secret expires.

6.2.1. Selecting a hash algorithm

The HMAC field in the HMAC TLV is 256 bit wide. Therefore, the HMAC MUST be based on a hash function whose output is at least 256 bits. If the output of the hash function is 256, then this output is simply inserted in the HMAC field. If the output of the hash function is larger than 256 bits, then the output value is truncated to 256 by taking the least-significant 256 bits and inserting them in the HMAC field.

SRH implementations can support multiple hash functions but MUST implement SHA-2 [FIPS180-4] in its SHA-256 variant.

NOTE: SHA-1 is currently used by some early implementations used for quick interoperations testing, the 160-bit hash value must then be right-hand padded with 96 bits set to 0. The authors understand that this is not secure but is ok for limited tests.

6.2.2. Performance impact of HMAC

While adding an HMAC to each and every SR packet increases the security, it has a performance impact. Nevertheless, it must be noted that:

- the HMAC field is used only when SRH is added by a device (such as a home set-up box) which is outside of the segment routing domain. If the SRH is added by a router in the trusted segment routing domain, then, there is no need for an HMAC field, hence no performance impact.

- when present, the HMAC field MUST only be checked and validated by the first router of the segment routing domain, this router is named 'validating SR router'. Downstream routers may not inspect the HMAC field.

- this validating router can also have a cache of <IPv6 header + SRH, HMAC field value> to improve the performance. It is not the

same use case as in IPsec where HMAC value was unique per packet, in SRH, the HMAC value is unique per flow.

- Last point, hash functions such as SHA-2 have been optimized for security and performance and there are multiple implementations with good performance.

With the above points in mind, the performance impact of using HMAC is minimized.

6.2.3. Pre-shared key management

The field HMAC Key-id allows for:

- key roll-over: when there is a need to change the key (the hash pre-shared secret), then multiple pre-shared keys can be used simultaneously. The validating routing can have a table of <HMAC Key-id, pre-shared secret> for the currently active and future keys.

- different algorithms: by extending the previous table to <HMAC Key-id, hash function, pre-shared secret>, the validating router can also support simultaneously several hash algorithms (see section Section 6.2.1)

The pre-shared secret distribution can be done:

- in the configuration of the validating routers, either by static configuration or any SDN oriented approach;

- dynamically using a trusted key distribution such as [RFC6407]

The intent of this document is NOT to define yet-another-key-distribution-protocol.

6.3. Deployment Models

6.3.1. Nodes within the SR domain

An SR domain is defined as a set of interconnected routers where all routers at the perimeter are configured to add and act on SRH. Some routers inside the SR domain can also act on SRH or simply forward IPv6 packets.

The routers inside an SR domain can be trusted to generate SRH and to process SRH received on interfaces that are part of the SR domain. These nodes MUST drop all SRH packets received on an interface that is not part of the SR domain and containing an SRH whose HMAC field

cannot be validated by local policies. This includes obviously packet with an SRH generated by a non-cooperative SR domain.

If the validation fails, then these packets MUST be dropped, ICMP error messages (parameter problem) SHOULD be generated (but rate limited) and SHOULD be logged.

6.3.2. Nodes outside of the SR domain

Nodes outside of the SR domain cannot be trusted for physical security; hence, they need to request by some trusted means (outside the scope of this document) a complete SRH for each new connection (i.e. new destination address). The received SRH MUST include an HMAC TLV which is computed correctly (see Section 6.2).

When an outside node sends a packet with an SRH and towards an SR domain ingress node, the packet MUST contain the HMAC TLV (with a Key-id and HMAC fields) and the the destination address MUST be an address of an SR domain ingress node .

The ingress SR router, i.e., the router with an interface address equal to the destination address, MUST verify the HMAC TLV.

If the validation is successful, then the packet is simply forwarded as usual for an SR packet. As long as the packet travels within the SR domain, no further HMAC check needs to be done. Subsequent routers in the SR domain MAY verify the HMAC TLV when they process the SRH (i.e. when they are the destination).

If the validation fails, then this packet MUST be dropped, an ICMP error message (parameter problem) SHOULD be generated (but rate limited) and SHOULD be logged.

6.3.3. SR path exposure

As the intermediate SR nodes addresses appears in the SRH, if this SRH is visible to an outsider then he/she could reuse this knowledge to launch an attack on the intermediate SR nodes or get some insider knowledge on the topology. This is especially applicable when the path between the source node and the first SR domain ingress router is on the public Internet.

The first remark is to state that 'security by obscurity' is never enough; in other words, the security policy of the SR domain MUST assume that the internal topology and addressing is known by the attacker. A simple traceroute will also give the same information (with even more information as all intermediate nodes between SID will also be exposed). IPsec Encapsulating Security Payload

[RFC4303] cannot be use to protect the SRH as per RFC4303 the ESP header must appear after any routing header (including SRH).

To prevent a user to leverage the gained knowledge by intercepting SRH, it it recommended to apply an infrastructure Access Control List (iACL) at the edge of the SR domain. This iACL will drop all packets from outside the SR-domain whose destination is any address of any router inside the domain. This security policy should be tuned for local operations.

6.3.4. Impact of BCP-38

BCP-38 [RFC2827], also known as "Network Ingress Filtering", checks whether the source address of packets received on an interface is valid for this interface. The use of loose source routing such as SRH forces packets to follow a path which differs from the expected routing. Therefore, if BCP-38 was implemented in all routers inside the SR domain, then SR packets could be received by an interface which is not expected one and the packets could be dropped.

As an SR domain is usually a subset of one administrative domain, and as BCP-38 is only deployed at the ingress routers of this administrative domain and as packets arriving at those ingress routers have been normally forwarded using the normal routing information, then there is no reason why this ingress router should drop the SRH packet based on BCP-38. Routers inside the domain commonly do not apply BCP-38; so, this is not a problem.

7. IANA Considerations

This document makes the following registrations in the Internet Protocol Version 6 (IPv6) Parameters "Routing Type" registry maintained by IANA:

Suggested Value	Description	Reference
4	Segment Routing Header (SRH)	This document

This document request IANA to create and maintain a new Registry: "Segment Routing Header TLVs"

7.1. Segment Routing Header Flags Register

This document requests the creation of a new IANA managed registry to identify SRH Flags Bits. The registration procedure is "Expert Review" as defined in [RFC8126]. Suggested registry name is "Segment

Routing Header Flags". Flags is 8 bits, the following bits are defined in this document:

Suggested Bit	Description	Reference
4	HMAC	This document

7.2. Segment Routing Header TLVs Register

This document requests the creation of a new IANA managed registry to identify SRH TLVs. The registration procedure is "Expert Review" as defined in [RFC8126]. Suggested registry name is "Segment Routing Header TLVs". A TLV is identified through an unsigned 8 bit codepoint value. The following codepoints are defined in this document:

Suggested Value	Description	Reference
5	HMAC TLV	This document
128	Pad0 TLV	This document
129	PadN TLV	This document

8. Implementation Status

This section is to be removed prior to publishing as an RFC.

8.1. Linux

Name: Linux Kernel v4.14

Status: Production

Implementation: adds SRH, performs END and END.X processing, supports HMAC TLV

Details: https://irtf.org/anrw/2017/anrw17-final3.pdf and [I-D.filsfils-spring-srv6-interop]

8.2. Cisco Systems

Name: IOS XR and IOS XE

Status: Pre-production

Implementation: adds SRH, performs END and END.X processing, no TLV processing

Details: [I-D.filsfils-spring-srv6-interop]

8.3. FD.io

Name: VPP/Segment Routing for IPv6

Status: Production

Implementation: adds SRH, performs END and END.X processing, no TLV processing

Details: https://wiki.fd.io/view/VPP/Segment_Routing_for_IPv6 and [I-D.filsfils-spring-srv6-interop]

8.4. Barefoot

Name: Barefoot Networks Tofino NPU

Status: Prototype

Implementation: performs END and END.X processing, no TLV processing

Details: [I-D.filsfils-spring-srv6-interop]

8.5. Juniper

Name: Juniper Networks Trio and vTrio NPU's

Status: Prototype & Experimental

Implementation: SRH insertion mode, Process SID where SID is an interface address, no TLV processing

9. Contributors

Kamran Raza, Darren Dukes, Brian Field, Daniel Bernier, Ida Leung, Jen Linkova, Ebben Aries, Tomoya Kosugi, Eric Vyncke, David Lebrun, Dirk Steinberg, Robert Raszuk, Dave Barach, John Brzozowski, Pierre Francois, Nagendra Kumar, Mark Townsley, Christian Martin, Roberta Maglione, James Connolly, Aloys Augustin contributed to the content of this document.

10. Acknowledgements

The authors would like to thank Ole Troan, Bob Hinden, Fred Baker, Brian Carpenter, Alexandru Petrescu, Punit Kumar Jaiswal, and David Lebrun for their comments to this document.

11. References

11.1. Normative References

[FIPS180-4]
National Institute of Standards and Technology, "FIPS 180-4 Secure Hash Standard (SHS)", March 2012, <http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>.

[I-D.ietf-spring-segment-routing]
Filsfils, C., Previdi, S., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", draft-ietf-spring-segment-routing-15 (work in progress), January 2018.

[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>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

[RFC5095] Abley, J., Savola, P., and G. Neville-Neil, "Deprecation of Type 0 Routing Headers in IPv6", RFC 5095, DOI 10.17487/RFC5095, December 2007, <https://www.rfc-editor.org/info/rfc5095>.

[RFC6407] Weis, B., Rowles, S., and T. Hardjono, "The Group Domain of Interpretation", RFC 6407, DOI 10.17487/RFC6407, October 2011, <https://www.rfc-editor.org/info/rfc6407>.

[RFC7855] Previdi, S., Ed., Filsfils, C., Ed., Decraene, B., Litkowski, S., Horneffer, M., and R. Shakir, "Source Packet Routing in Networking (SPRING) Problem Statement and Requirements", RFC 7855, DOI 10.17487/RFC7855, May 2016, <https://www.rfc-editor.org/info/rfc7855>.

[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>.

[RFC8354] Brzozowski, J., Leddy, J., Filsfils, C., Maglione, R., Ed., and M. Townsley, "Use Cases for IPv6 Source Packet Routing in Networking (SPRING)", RFC 8354, DOI 10.17487/RFC8354, March 2018, <https://www.rfc-editor.org/info/rfc8354>.

11.2. Informative References

[I-D.filsfils-spring-srv6-interop]
Filsfils, C., Clad, F., Camarillo, P., Abdelsalam, A., Salsano, S., Bonaventure, O., Horn, J., and J. Liste, "SRv6 interoperability report", draft-filsfils-spring-srv6-interop-00 (work in progress), March 2018.

[I-D.filsfils-spring-srv6-network-programming]
Filsfils, C., Li, Z., Leddy, J., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Steinberg, D., Raszuk, R., Matsushima, S., Lebrun, D., Decraene, B., Peirens, B., Salsano, S., Naik, G., Elmalky, H., Jonnalagadda, P., and M. Sharif, "SRv6 Network Programming", draft-filsfils-spring-srv6-network-programming-04 (work in progress), March 2018.

[I-D.ietf-isis-l2bundles]
Ginsberg, L., Bashandy, A., Filsfils, C., Nanduri, M., and E. Aries, "Advertising L2 Bundle Member Link Attributes in IS-IS", draft-ietf-isis-l2bundles-07 (work in progress), May 2017.

[I-D.ietf-isis-segment-routing-extensions]
Previdi, S., Ginsberg, L., Filsfils, C., Bashandy, A., Gredler, H., Litkowski, S., Decraene, B., and J. Tantsura, "IS-IS Extensions for Segment Routing", draft-ietf-isis-segment-routing-extensions-15 (work in progress), December 2017.

[I-D.ietf-ospf-ospfv3-segment-routing-extensions]
Psenak, P., Filsfils, C., Previdi, S., Gredler, H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPFv3 Extensions for Segment Routing", draft-ietf-ospf-ospfv3-segment-routing-extensions-11 (work in progress), January 2018.

[I-D.ietf-spring-segment-routing-mpls]
Bashandy, A., Filsfils, C., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with MPLS data plane", draft-ietf-spring-segment-routing-mpls-13 (work in progress), April 2018.

[RFC1940] Estrin, D., Li, T., Rekhter, Y., Varadhan, K., and D. Zappala, "Source Demand Routing: Packet Format and Forwarding Specification (Version 1)", RFC 1940, DOI 10.17487/RFC1940, May 1996, <https://www.rfc-editor.org/info/rfc1940>.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/info/rfc2104>.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <https://www.rfc-editor.org/info/rfc2827>.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <https://www.rfc-editor.org/info/rfc4302>.

[RFC4942] Davies, E., Krishnan, S., and P. Savola, "IPv6 Transition/ Co-existence Security Considerations", RFC 4942, DOI 10.17487/RFC4942, September 2007, <https://www.rfc-editor.org/info/rfc4942>.

[RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, DOI 10.17487/RFC5308, October 2008, <https://www.rfc-editor.org/info/rfc5308>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, DOI 10.17487/RFC6438, November 2011, <https://www.rfc-editor.org/info/rfc6438>.

[RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, DOI 10.17487/RFC6554, March 2012, <https://www.rfc-editor.org/info/rfc6554>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

Authors' Addresses

Stefano Previdi
Individual
Italy

Email: stefano@previdi.net

Clarence Filsfils (editor)
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

John Leddy
Comcast
4100 East Dry Creek Road
Centennial, CO 80122
US

Email: John_Leddy@comcast.com

Satoru Matsushima
Softbank

Email: satoru.matsushima@g.softbank.co.jp

Daniel Voyer (editor)
Bell Canada

Email: daniel.voyer@bell.ca