Integration of Network Service Header (NSH) and Segment Routing for Service Function Chaining (SFC)
draft-ietf-spring-nsh-sr-04

Abstract

   This document describes the integration of Network Service Header
   (NSH) [RFC8300] and Segment Routing (SR) [RFC8402], as well as
   encapsulation details, to support Service Function Chaining (SFC)
   [RFC7665] in an efficient manner while maintaining separation of the
   service and transport planes as originally intended by the SFC
   architecture.

   Combining these technologies allows SR to be used for steering
   packets between Service Function Forwarders (SFF) along a given
   Service Function Path (SFP) while NSH has the responsibility for
   maintaining the integrity of the service plane, the SFC instance
   context, and any associated metadata.

   The integration described in this document demonstrates that NSH and
   SR can work jointly and complement each other leaving the network
   operator with the flexibility to use whichever transport technology
   makes sense in specific areas of their network infrastructure, and
   still maintain an end-to-end service plane using NSH.

Status of This Memo

Table of Contents

1. Introduction

1.1. SFC Overview and Rationale

   The dynamic enforcement of a service-derived and adequate forwarding
   policy for packets entering a network that supports advanced Service
   Functions (SFs) has become a key challenge for network operators.
   Particularly, cascading SFs at the so-called Third Generation
   Partnership Project (3GPP) Gi interface (N6 interface in 5G
   architecture) in the context of mobile network infrastructure, have

shown their limitations, such as the same redundant classification
features must be supported by many SFs to execute their function,
some SFs receive traffic that they are not supposed to process (e.g.,
TCP proxies receiving UDP traffic), which inevitably affects their
dimensioning and performance, an increased design complexity related
to the properly ordered invocation of several SFs, etc.

In order to solve those problems and to decouple the services
topology from the underlying physical network while allowing for
simplified service delivery, Service Function Chaining (SFC)
techniques have been introduced [RFC7665].

SFC techniques are meant to rationalize the service delivery logic
and master the companion complexity while optimizing service
activation time cycles for operators that need more agile service
delivery procedures to better accommodate ever-demanding customer
requirements. Indeed, SFC allows to dynamically create service
planes that can be used by specific traffic flows. Each service
plane is realized by invoking and chaining the relevant service
functions in the right sequence. [RFC7498] provides an overview of
the overall SFC problem space and [RFC7665] specifies an SFC data
plane architecture. The SFC architecture does not make assumptions
on how advanced features (e.g., load-balancing, loose or strict
service paths) could be enabled within a domain. Various deployment
options are made available to operators with the SFC architecture and
this approach is fundamental to accommodate various and heterogeneous
deployment contexts.

Many approaches can be considered for encoding the information
required for SFC purposes (e.g., communicate a service chain pointer,
encode a list of loose/explicit paths, or disseminate a service chain
identifier together with a set of context information). Likewise,
many approaches can also be considered for the channel to be used to
carry SFC-specific information (e.g., define a new header, re-use
existing packet header fields, or define an IPv6 extension header).
Among all these approaches, the IETF created a transport-independent
SFC encapsulation scheme: NSH. This design is pragmatic as it does
not require replicating the same specification effort as a function
of underlying transport encapsulation. Moreover, this design
approach encourages consistent SFC-based service delivery in networks
enabling distinct transport protocols in various network segments or
even between SFFs vs SF-SFF hops.

1.2. 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

[RFC2119] [RFC8174] when, and only when, they appear in all capitals,
as shown here.

2. SFC within Segment Routing Networks

As described in [RFC8402], SR leverages the source routing technique.
Concretely, a node steers a packet through an SR policy instantiated
as an ordered list of instructions called segments. While initially
designed for policy-based source routing, SR also finds its
application in supporting SFC
[I-D.ietf-spring-sr-service-programming].

The two SR data plane encapsulations, namely SR-MPLS [RFC8660] and
SRv6 [RFC8754], can both encode an SF as a segment so that an SFC can
be specified as a segment list. Nevertheless, and as discussed in
[RFC7498], traffic steering is only a subset of the issues that
motivated the design of the SFC architecture. Further considerations
such as simplifying classification at intermediate SFs and allowing
for coordinated behaviors among SFs by means of supplying context
information (a.k.a. metadata) should be considered when designing an
SFC data plane solution.

While each scheme (i.e., NSH-based SFC and SR-based SFC) can work
independently, this document describes how the two can be used
together in concert and complement each other through two
representative application scenarios. Both application scenarios may
be supported using either SR-MPLS or SRv6:

o NSH-based SFC with SR-based transport plane: in this scenario SR-
   MPLS or SRv6 provides the transport encapsulation between SFFs
   while NSH is used to convey and trigger SFC policies.

o SR-based SFC with integrated NSH service plane: in this scenario
   each service hop of the SFC is represented as a segment of the SR
   segment-list. SR is thus responsible for steering traffic through
   the necessary SFFs as part of the segment routing path while NSH
   is responsible for maintaining the service plane and holding the
   SFC instance context (including associated metadata).

It is of course possible to combine both of these two scenarios to
support specific deployment requirements and use cases.

A classifier SHOULD assign an NSH Service Path Identifier (SPI) per
SR policy so that different traffic flows that use the same NSH
Service Function Path (SFP) but different SR policy can coexist on
the same SFP without conflict during SFF processing.

3. NSH-based SFC with SR-MPLS or SRv6 Transport Tunnel

   Because of the transport-independent nature of NSH-based service
   function chains, it is expected that the NSH has broad applicability
   across different network domains (e.g., access, core). By way of
   illustration the various SFs involved in a service function chain may
   be available in a single data center, or spread throughout multiple
   locations (e.g., data centers, different POPs), depending upon the
   network operator preference and/or availability of service resources.
   Regardless of where the SFs are deployed it is necessary to provide
   traffic steering through a set of SFFs, and when NSH and SR are
   integrated, this is provided by SR-MPLS or SRv6.

   The following three figures provide an example of an SFC established
   flow F that has SF instances located in different data centers, DC1
   and DC2. For the purpose of illustration, let the SFC's NSH SPI be
   100 and the initial Service Index (SI) be 255.

   Referring to Figure 1, packets of flow F in DC1 are classified into
   an NSH-based SFC and encapsulated after classification as <Inner
   Pkt><NSH: SPI 100, SI 255><Outer-transport> and forwarded to SFF1
   (which is the first SFF hop for this service function chain).

   After removing the outer transport encapsulation, SFF1 uses the SPI
   and SI carried within the NSH encapsulation to determine that it
   should forward the packet to SF1. SF1 applies its service,
   decrements the SI by 1, and returns the packet to SFF1. SFF1
   therefore has <SPI 100, SI 254> when the packet comes back from SF1.
   SFF1 does a lookup on <SPI 100, SI 254> which results in <next-hop:
   DC1-GW1> and forwards the packet to DC1-GW1.

```
+------------------------- DC1 ----------------------------+
|                      +-----+                             |
|                      | SF1 |                             |
|                      +--+--+                             |
|                         |                                |
|                         |                                |
|     +-----------+       |     +-----------+              |
|     | N(100,255) |      |     | F:Inner Pkt|             |
|     +-----------+       |     +-----------+              |
|     | F:Inner Pkt|      |     | N(100,254) |             |
|     +-----------+  ^    |  |  +-----------+              |
|               (2) |    |  | (3)                          |
|                   |    |  | v                            |
|             (1)        |       (4)                       |
|+-----------+   ---->   +--+---+   ---->   +---------+     |
||           |    NSH    |      |    NSH    |         |  |  |
|| Classifier +-----------+ SFF1 +--------------+ DC1-GW1 + |
||           |           |      |           |         |  |  |
|+-----------+           +------+           +---------+  |  |
|                                                         |
|           +-----------+       +-----------+             |
|           | N(100,255) |       | N(100,254) |            |
|           +-----------+       +-----------+             |
|           | F:Inner Pkt|       | F:Inner Pkt|           |
|           +-----------+       +-----------+             |
|                                                         |
+---------------------------------------------------------+
```

                    Figure 1: SR for inter-DC SFC - Part 1

   Referring now to Figure 2, DC1-GW1 performs a lookup using the
   information conveyed in the NSH which results in <next-hop: DC2-GW1,
   encapsulation: SR>. The SR encapsulation, which may be SR-MPLS or
   SRv6, has the SR segment-list to forward the packet across the inter-
   DC network to DC2.

```
                    +----------- Inter DC ----------------+
                    |                (5)                   |
   +------+  ---->  | +---------+   ---->      +---------+ |
   |      |   NSH   | |         |      SR      |         | |
   + SFF1 +---------|-+ DC1-GW1 +-------------+ DC2-GW1 + |
   |      |         | |         |             |         | |
   +------+         | +---------+             +---------+ |
                    |                                     |
                    |         +------------+              |
                    |         | S(DC2-GW1) |              |
                    |         +------------+              |
                    |         | N(100,254) |              |
                    |         +------------+              |
                    |         | F:Inner Pkt|              |
                    |         +------------+              |
                    +-------------------------------------+
```
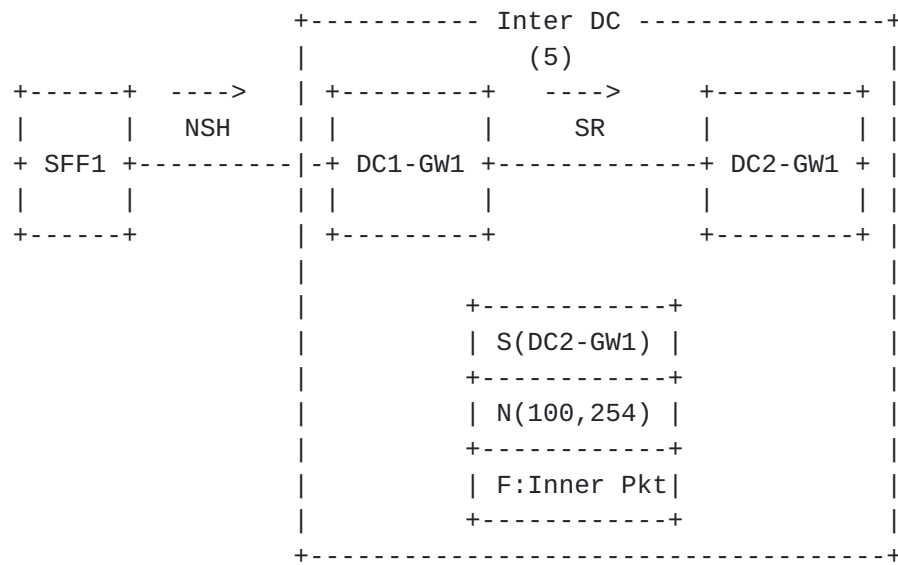
Figure 2: SR for inter-DC SFC - Part 2

When the packet arrives at DC2, as shown in Figure 3, the SR
encapsulation is removed and DC2-GW1 performs a lookup on the NSH
which results in next hop: SFF2. When SFF2 receives the packet, it
performs a lookup on <NSH: SPI 100, SI 254> and determines to forward
the packet to SF2. SF2 applies its service, decrements the SI by 1,
and returns the packet to SFF2. SFF2 therefore has <NSH: SPI 100, SI
253> when the packet comes back from SF2. SFF2 does a lookup on
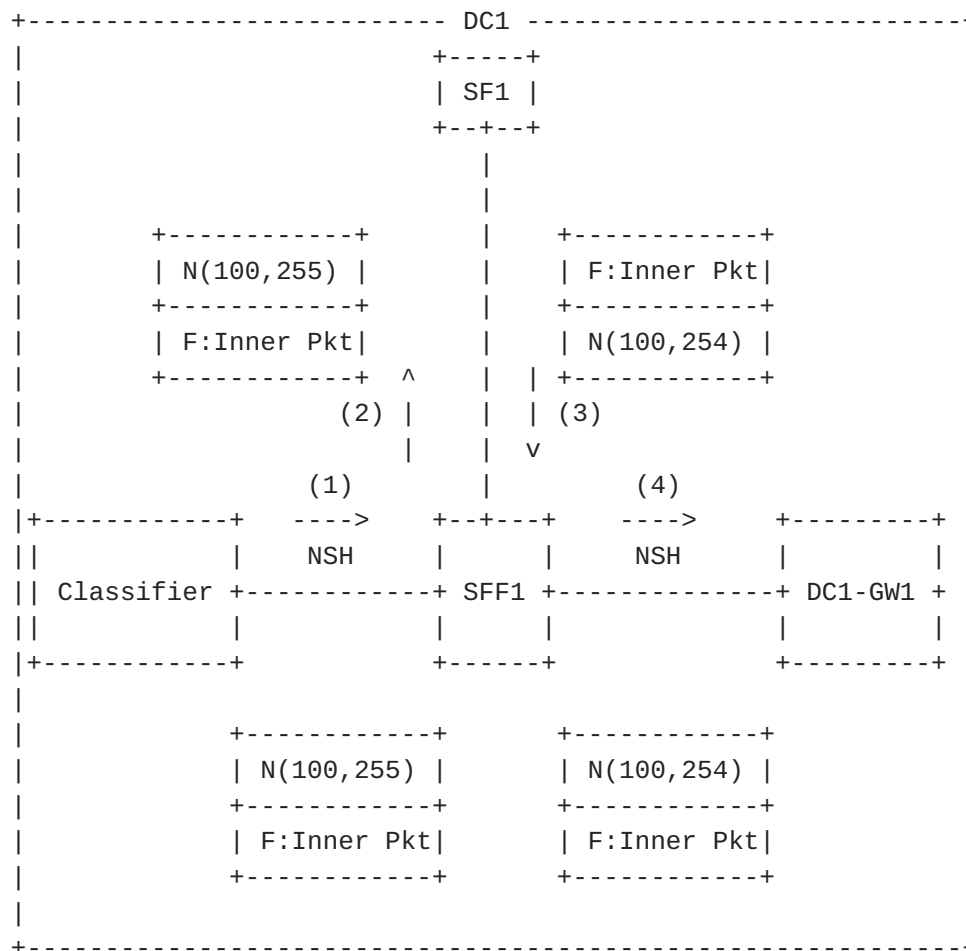<NSH: SPI 100, SI 253> which results in end of service function
chain.

```
+----------------------- DC2 --------------------+
|                      +-----+                    |
|                      | SF2 |                    |
|                      +--+--+                    |
|                         |                       |
|                         |                       |
|      +-----------+      |     +-----------+     |
|      | N(100,254) |     |     | F:Inner Pkt|    |
|      +-----------+      |     +-----------+     |
|      | F:Inner Pkt|     |     | N(100,253) |    |
|      +-----------+  ^ | | +-----------+         |
|               (7) | | | (8)                     |
|                   | |  v                        |
|          (6)      |     (9)                     |
|+----------+  ---->   +--+---+ ---->             |
||          |   NSH    |      |  IP               |
|| DC2-GW1  +-----------+ SFF2 |                   |
||          |          |      |                    |
|+----------+          +------+                    |
|                                                 |
|        +-----------+     +-----------+          |
|        | N(100,254) |    | F:Inner Pkt|         |
|        +-----------+     +-----------+          |
|        | F:Inner Pkt|                           |
|        +-----------+                            |
+-------------------------------------------------+
```

                 Figure 3: SR for inter-DC SFC - Part 3

   The benefits of this scheme are listed hereafter:

   o The network operator is able to take advantage of the transport-
      independent nature of the NSH encapsulation, while the service is
      provisioned end2end.

   o The network operator is able to take advantage of the traffic
      steering (traffic engineering) capability of SR where appropriate.

   o Clear responsibility division and scope between NSH and SR.

   Note that this scenario is applicable to any case where multiple
   segments of a service function chain are distributed across multiple
   domains or where traffic-engineered paths are necessary between SFFs
   (strict forwarding paths for example). Further note that the above
   example can also be implemented using end to end segment routing
   between SFF1 and SFF2. (As such DC-GW1 and DC-GW2 are forwarding the

packets based on segment routing instructions and are not looking at
the NSH header for forwarding).

4. SR-based SFC with Integrated NSH Service Plane

In this scenario we assume that the SFs are NSH-aware and therefore
it should not be necessary to implement an SFC proxy to achieve SFC.
The operation relies upon SR-MPLS or SRv6 to perform SFF-SFF
transport and NSH to provide the service plane between SFs thereby
maintaining SFC context (e.g., the service plane path referenced by
the SPI) and any associated metadata.

When a service function chain is established, a packet associated
with that chain will first carry an NSH that will be used to maintain
the end-to-end service plane through use of the SFC context. The SFC
context is used by an SFF to determine the SR segment list for
forwarding the packet to the next-hop SFFs. The packet is then
encapsulated using the SR header and forwarded in the SR domain
following normal SR operations.

When a packet has to be forwarded to an SF attached to an SFF, the
SFF performs a lookup on the SID associated with the SF. In the case
of SR-MPLS this will be a prefix SID [RFC8402]. In the case of SRv6,
the behavior described within this document is assigned the name
END.NSH, and section 8.3 requests allocation of a code point by IANA.
The result of this lookup allows the SFF to retrieve the next hop
context between the SFF and SF (e.g., the destination MAC address in
case native Ethernet encapsulation is used between SFF and SF). In
addition the SFF strips the SR information from the packet, updates
the SR information, and saves it to a cache indexed by the NSH
Service Path Identifier (SPI) and the Service Index (SI) decremented
by 1. This saved SR information is used to encapsulate and forward
the packet(s) coming back from the SF.

The behavior of remembering the SR stack occurs at the end of the
regularly defined logic. The behavior of reattaching the stack
occurs before the SR process of forwarding the packet to the next
entry in the segment-list. Both behaviors are further detailed in
section 5.

When the SF receives the packet, it processes it as usual and sends
it back to the SFF. Once the SFF receives this packet, it extracts
the SR information using the NSH SPI and SI as the index into the
cache. The SFF then pushes the retrieved SR header on top of the NSH
header, and forwards the packet to the next segment in the segment
list.

Figure 4 illustrates an example of this scenario.

```
                    +-----+                      +-----+
                    | SF1 |                      | SF2 |
                    +--+--+                      +--+--+
                       |                            |
                       |                            |
          +-----------+ | +-----------+ +-----------+ | +-----------+
          |N(100,255) | | |F:Inner Pkt| |N(100,254) | | |F:Inner Pkt|
          +-----------+ | +-----------+ +-----------+ | +-----------+
          |F:Inner Pkt| | |N(100,254) | |F:Inner Pkt| | |N(100,253) |
          +-----------+ | +-----------+ +-----------+ | +-----------+
             (2) ^ | (3) |                   (5) ^ | (6) |
                 | |     |                       | |     |
                 | |     v                       | |     v
    +-----------+ (1)--> +-+----+       (4)-->        +---+--+ (7)-->IP
    |           | NSHoSR |      |       NSHoSR        |      |
    | Classifier +--------+ SFF1 +--------------------+ SFF2 |
    |           |        |      |                     |      |
    +-----------+        +------+                     +------+

             +------------+    +------------+
             |   S(SF1)   |    |   S(SF2)   |
             +------------+    +------------+
             |   S(SFF2)  |    | N(100,254) |
             +------------+    +------------+
             |   S(SF2)   |    | F:Inner Pkt|
             +------------+    +------------+
             | N(100,255) |
             +------------+
             | F:Inner Pkt|
             +------------+
```
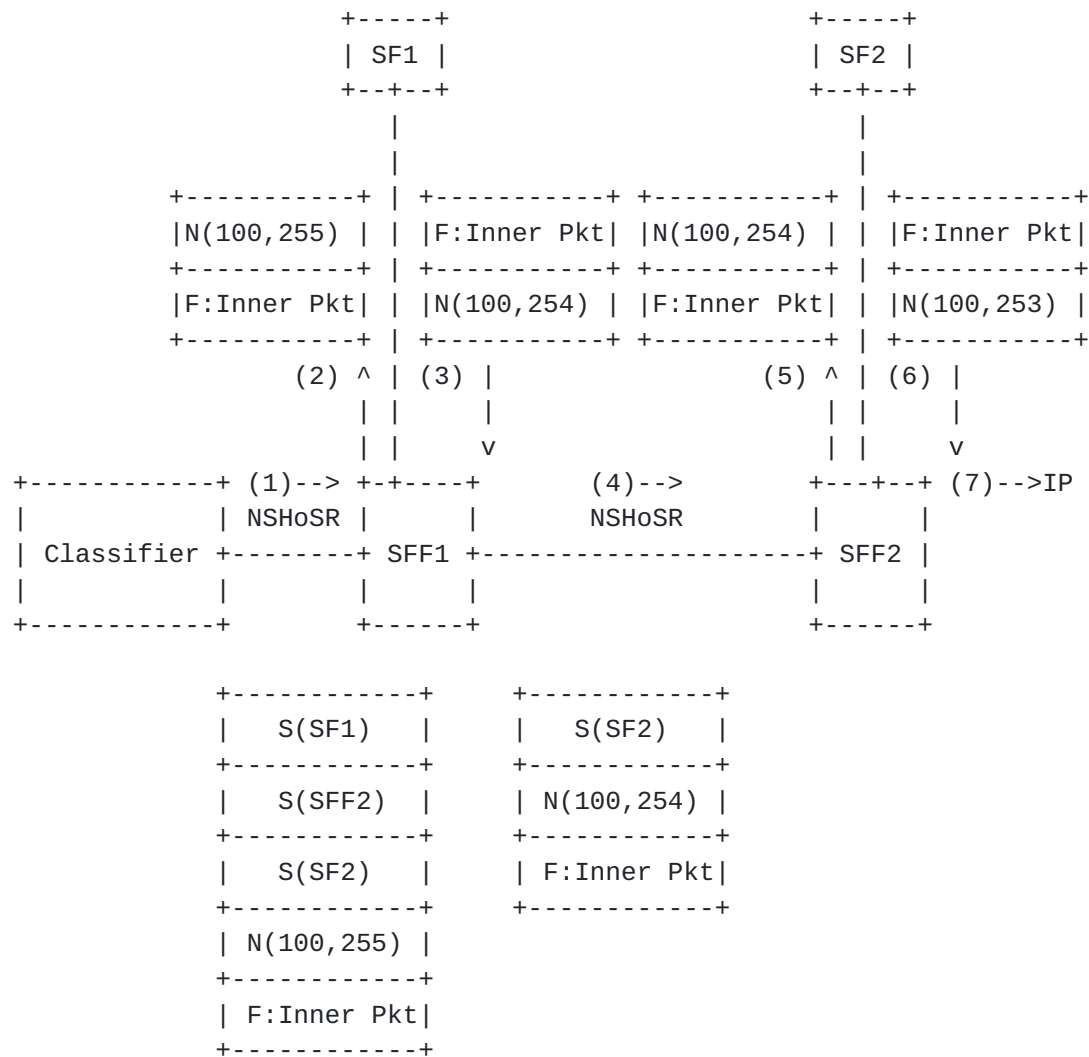
                      Figure 4: NSH over SR for SFC

   The benefits of this scheme include:

   o It is economically sound for SF vendors to only support one
      unified SFC solution. The SF is unaware of the SR.

   o It simplifies the SFF (i.e., the SR router) by nullifying the
      needs for re-classification and SR proxy.

   o SR is also used for forwarding purposes including between SFFs.

   o It takes advantage of SR to eliminate the NSH forwarding state in
      SFFs. This applies each time strict or loose SFPs are in use.

o It requires no interworking as would be the case if SR-MPLS based
   SFC and NSH-based SFC were deployed as independent mechanisms in
   different parts of the network.

5. Packet Processing Details

This section describes the End.NSH behavior and NSH processing logic.
The pseudo code is shown below.

When N receives a packet destined to S and S is a local End.NSH SID,
the processing is the same as that specified by RFC 8754 section
4.3.1.1, up through line S.16.

After S.15, if S is a local End.NSH SID, then:

S15.1. Remove and store IPv6 and SRH headers in local cache indexed
by <NSH: path-id, service-index -1>

S15.2. Submit the packet to the NSH FIB lookup and transmit to the
destination associated with <NSH: path-id, service-index>

Note: The End.NSH behavior interrupts the normal SRH packet
processing as described in RFC8754 section 4.3.1.1, which does not
continue to S16 at this time.

When a packet is returned to the SFF from the SF, reattach the cached
IPv6 and SRH headers based on the <NSH: path-id, service-index> from
the NSH header. Then resume processing from RFC 8754 section 4.3.1.1
with line S.16.

6. Encapsulation Details

6.1. NSH using SR-MPLS Transport

SR-MPLS instantiates Segment IDs (SIDs) as MPLS labels and therefore
the segment routing header is a stack of MPLS labels.

When carrying NSH within an SR-MPLS transport, the full encapsulation
headers are as illustrated in Figure 5.

```
                    +-----------------+
                    ~   MPLS-SR Labels ~
                    +-----------------+
                    |   NSH Base Hdr   |
                    +-----------------+
                    | Service Path Hdr |
                    +-----------------+
                    ~      Metadata     ~
                    +-----------------+
```

                 Figure 5: NSH using SR-MPLS Transport

   As described in [RFC8402], the IGP signaling extension for IGP-Prefix
   segment includes a flag to indicate whether directly connected
   neighbors of the node on which the prefix is attached should perform
   the NEXT operation or the CONTINUE operation when processing the SID.
   When NSH is carried beneath SR-MPLS it is necessary to terminate the
   NSH-based SFC at the tail-end node of the SR-MPLS label stack. This
   is the equivalent of MPLS Ultimate Hop Popping (UHP) and therefore
   the prefix-SID associated with the tail-end of the SFC MUST be
   advertised with the CONTINUE operation so that the penultimate hop
   node does not pop the top label of the SR-MPLS label stack and
   thereby expose NSH to the wrong SFF. This is realized by setting No-
   PHP flag in Prefix-SID Sub-TLV [RFC8667], [RFC8665]. It is
   RECOMMENDED that a specific prefix-SID be allocated at each node for
   use by the SFC application for this purpose.

   At the end of the SR-MPLS path it is necessary to provide an
   indication to the tail-end that NSH follows the SR-MPLS label stack
   as described by [RFC8596].

6.2. NSH using SRv6 Transport

   When carrying NSH within an SRv6 transport the full encapsulation is
   as illustrated in Figure 6.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Next Header   | Hdr Ext Len  | Routing Type  | Segments Left |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Last Entry    |    Flags     |               Tag             | S
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ e
     |                                                               | g
     |             Segment List[0] (128 bits IPv6 address)           | m
     |                                                               | e
     |                                                               | n
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ t
     |                                                               |
     |                                                               | R
     ~                             ...                               ~ o
     |                                                               | u
     |                                                               | t
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ i
     |                                                               | n
     |             Segment List[n] (128 bits IPv6 address)           | g
     |                                                               |
     |                                                               | S
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ R
     //                                                           // H
     //        Optional Type Length Value objects (variable)      //
     //                                                           //
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |Ver|O|U|   TTL    |   Length  |U|U|U|U|MD Type| Next Protocol |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ N
     |          Service Path Identifier        | Service Index | S
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ H
     |                                                               |
     ~           Variable-Length Context Headers  (opt.)            ~
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
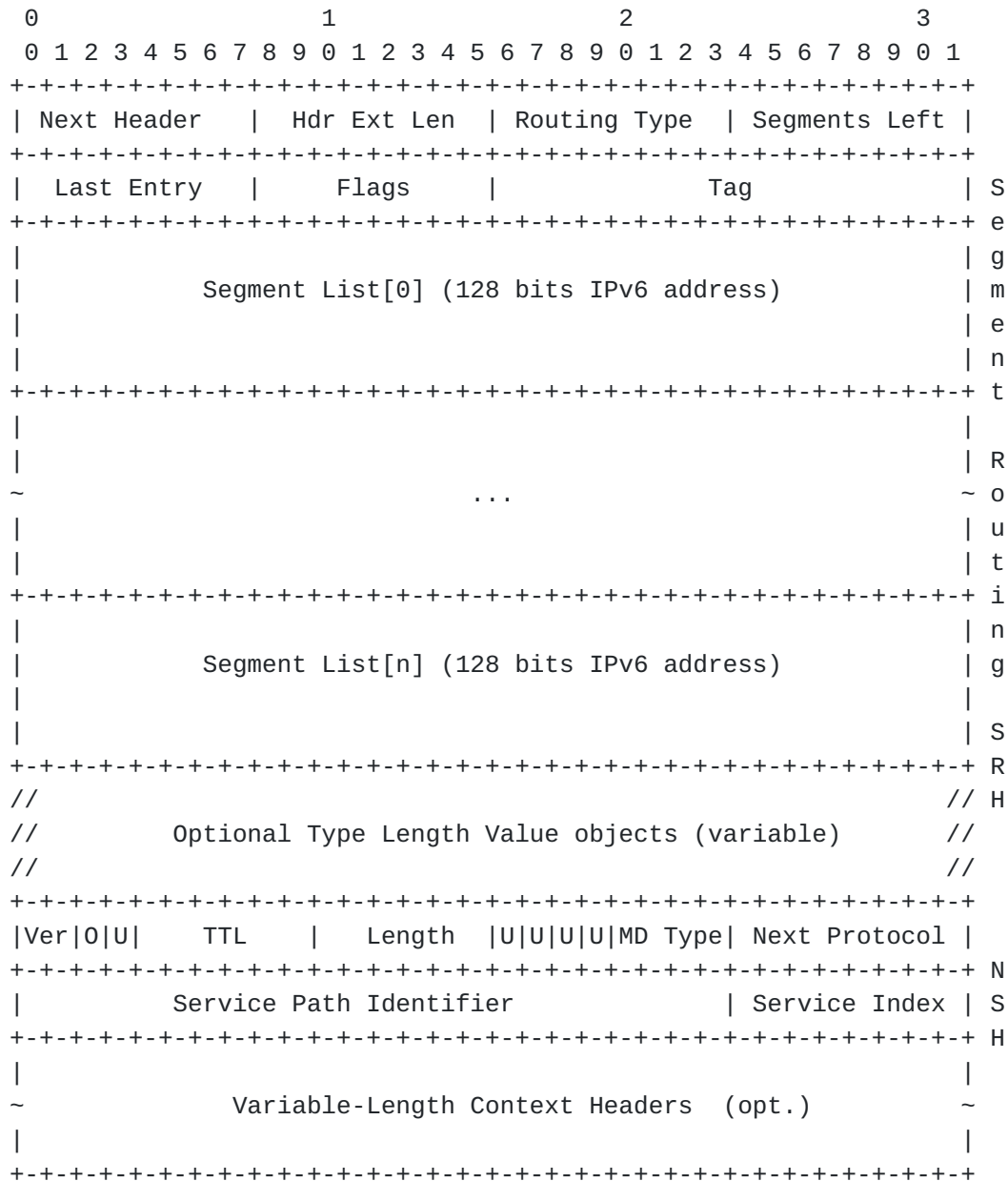
Figure 6: NSH using SRv6 Transport

Encapsulation of NSH following SRv6 may be indicated either by
encapsulating NSH in UDP (UDP port TBA1) and indicating UDP in the
Next Header field of the SRH, or by indicating an IP protocol number
for NSH in the Next Header of the SRH. The behavior for
encapsulating NSH over UDP, including the selection of the source
port number in particular, adheres to similar considerations as those
discussed in [RFC8086].

7. Security Considerations

    Generic SFC-related security considerations are discussed in
    [RFC7665].

    NSH-specific security considerations are discussed in [RFC8300].

    NSH-in-UDP with DTLS [RFC6347] should follow the considerations
    discussed in Section 5 of [RFC8086], with a destination port number
    set to TBA2.

    Generic segment routing related security considerations are discussed
    in section 7 of [RFC8754] and section 5 of [RFC8663].

8. IANA Considerations

8.1. UDP Port Number for NSH

 IANA is requested to assign the UDP port numbers TBA1 and TBA2 to the
 NSH from the "Service Name and Transport Protocol Port Number Registry"
 available at
 https://www.iana.org/assignments/service-names-port-numbers/
 service-names-port-numbers.xhtml:

 Service Name: NSH-in-UDP
 Transport Protocol(s): UDP
 Assignee: IESG iesg@ietf.org
 Contact: IETF Chair chair@ietf.org
 Description: NSH-in-UDP Encapsulation
 Reference: [ThisDocument]
 Port Number: TBA1
 Service Code: N/A
 Known Unauthorized Uses: N/A
 Assignment Notes: N/A

 Service Name: NSH-UDP-DTLS
 Transport Protocol(s): UDP
 Assignee: IESG iesg@ietf.org
 Contact: IETF Chair chair@ietf.org
 Description: NSH-in-UDP with DTLS Encapsulation
 Reference: [ThisDocument]
 Port Number: TBA2
 Service Code: N/A
 Known Unauthorized Uses: N/A
 Assignment Notes: N/A

8.2. Protocol Number for NSH

IANA is requested to assign a protocol number TBA3 for the NSH from the
"Assigned Internet Protocol Numbers" registry available at
https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml

```
+---------+---------+-------------+--------------+----------------+
| Decimal | Keyword |  Protocol   |     IPv6     |   Reference    |
|         |         |             |  Extension   |                |
|         |         |             |   Header     |                |
+---------+---------+-------------+--------------+----------------+
|  TBA3   |   NSH   |   Network   |      N       | [ThisDocument] |
|         |         |   Service   |              |                |
|         |         |   Header    |              |                |
+---------+---------+-------------+--------------+----------------+
```

8.3. SRv6 Endpoint Behavior for NSH

This I-D requests IANA to allocate, within the "SRv6 Endpoint Behaviors"
sub-registry belonging to the top-level "Segment-routing with IPv6 data
plane (SRv6) Parameters" registry, the following allocations:

```
Value      Description                              Reference
-----------------------------------------------------------
TBA4       End.NSH  - NSH Segment                   [This.ID]
```

9. Contributing Authors

The following co-authors, along with their respective affiliations at
the time of WG adoption, provided valuable inputs and text contributions
to this document.

Mohamed Boucadair
Orange
mohamed.boucadair@orange.com

Joel Halpern
Ericsson
joel.halpern@ericsson.com

Syed Hassan
Cisco System, inc.
shassan@cisco.com

Wim Henderickx
Nokia
wim.henderickx@nokia.com

Haoyu Song
Futurewei Technologies
haoyu.song@futurewei.com

10. References

10.1. 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>.

   [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer
              Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347,
              January 2012, <https://www.rfc-editor.org/info/rfc6347>.

   [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function
              Chaining (SFC) Architecture", RFC 7665,
              DOI 10.17487/RFC7665, October 2015,
              <https://www.rfc-editor.org/info/rfc7665>.

   [RFC8086] Yong, L., Ed., Crabbe, E., Xu, X., and T. Herbert, "GRE-
              in-UDP Encapsulation", RFC 8086, DOI 10.17487/RFC8086,
              March 2017, <https://www.rfc-editor.org/info/rfc8086>.

   [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>.

   [RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed.,
              "Network Service Header (NSH)", RFC 8300,
              DOI 10.17487/RFC8300, January 2018,
              <https://www.rfc-editor.org/info/rfc8300>.

   [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>.

   [RFC8596] Malis, A., Bryant, S., Halpern, J., and W. Henderickx,
              "MPLS Transport Encapsulation for the Service Function
              Chaining (SFC) Network Service Header (NSH)", RFC 8596,
              DOI 10.17487/RFC8596, June 2019,
              <https://www.rfc-editor.org/info/rfc8596>.

   [RFC8660] Bashandy, A., Ed., Filsfils, C., Ed., Previdi, S.,
              Decraene, B., Litkowski, S., and R. Shakir, "Segment
              Routing with the MPLS Data Plane", RFC 8660,
              DOI 10.17487/RFC8660, December 2019,
              <https://www.rfc-editor.org/info/rfc8660>.

   [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>.

   [RFC8665] Psenak, P., Ed., Previdi, S., Ed., Filsfils, C., Gredler,
              H., Shakir, R., Henderickx, W., and J. Tantsura, "OSPF
              Extensions for Segment Routing", RFC 8665,
              DOI 10.17487/RFC8665, December 2019,
              <https://www.rfc-editor.org/info/rfc8665>.

   [RFC8667] Previdi, S., Ed., Ginsberg, L., Ed., Filsfils, C.,
              Bashandy, A., Gredler, H., and B. Decraene, "IS-IS
              Extensions for Segment Routing", RFC 8667,
              DOI 10.17487/RFC8667, December 2019,
              <https://www.rfc-editor.org/info/rfc8667>.

   [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>.

10.2. **Informative References**

   [I-D.ietf-spring-sr-service-programming]
            Clad, F., Xu, X., Filsfils, C., daniel.bernier@bell.ca,
            d., Li, C., Decraene, B., Ma, S., Yadlapalli, C.,
            Henderickx, W., and S. Salsano, "Service Programming with
            Segment Routing", draft-ietf-spring-sr-service-
            programming-03 (work in progress), September 2020.

   [RFC7498] Quinn, P., Ed. and T. Nadeau, Ed., "Problem Statement for
            Service Function Chaining", RFC 7498,
            DOI 10.17487/RFC7498, April 2015,
            <https://www.rfc-editor.org/info/rfc7498>.

Authors' Addresses

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

   Email: james.n.guichard@futurewei.com

   Jeff Tantsura (editor)
   Apstra inc.
   USA

   Email: jefftant.ietf@gmail.com