SPRING
Internet-Draft
Intended status: Standards Track
Expires: October 8, 2020

J. Guichard, Ed.
H. Song
Futurewei Technologies
J. Tantsura
Apstra inc.
J. Halpern
Ericsson
W. Henderickx
Nokia
M. Boucadair
Orange
S. Hassan
Cisco Systems
April 6, 2020

Network Service Header (NSH) and Segment Routing Integration for Service Function Chaining (SFC)

draft-ietf-spring-nsh-sr-02

Abstract

This document describes two application scenarios where Network Service Header (NSH) and Segment Routing (SR) techniques can be deployed together to support Service Function Chaining (SFC) in an efficient manner while maintaining separation of the service and transport planes as originally intended by the SFC architecture.

In the first scenario, an NSH-based SFC is created using SR as the transport between Service Function Forwarders (SFFs). SR in this case is just one of many encapsulations that could be used to maintain the transport-independent nature of NSH-based service chains.

In the second scenario, SR is used to represent each service hop of the NSH-based SFC as a segment within the segment-list. SR and NSH in this case are integrated.

In both scenarios SR is responsible for steering packets between SFFs along a given Service Function Path (SFP) while NSH is responsible for maintaining the integrity of the service plane, the SFC instance context, and any associated metadata.

These application scenarios demonstrate 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

Copyright Notice

Table of Contents

1. Introduction

1.1. SFC Overview and Rationale

The dynamic enforcement of a service-derived, 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, for example at the Gi interface in the context of mobile network infrastructure, have shown their limits, such as the same redundant classification features must be supported by many SFs in order to execute their function, some SFs are receiving 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 avoid the adherence with the underlying physical network topology while allowing for simplified service delivery, Service Function Chaining (SFC) techniques have been introduced.

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 SFC problem space and [RFC7665] specifies an SFC architecture. The SFC architecture has the merit to not make assumptions on how advanced features (e.g., load-balancing, loose or strict service paths) have to be enabled with 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, disseminate a service chain identifier together with a set of context information, etc.). 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 fields, define an IPv6 extension header, etc.). Among all these approaches, the IETF endorsed a transport-independent SFC encapsulation scheme: NSH [RFC8300]; which is the most mature SFC encapsulation solution. 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 segments of the network 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 [RFC2119] when, and only when, they appear in all capitals, as shown here.

1.3. SFC within SR Networks

As described in [RFC8402], Segment Routing (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.xuclad-spring-sr-service-programming]. The two SR flavors, namely MPLS-SR [RFC8660] and SRv6 [RFC8754], can both encode a Service Function (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 should be taken into account 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 MPLS-SR or SRv6:

- NSH-based SFC with SR-based transport plane: in this scenario segment routing provides the transport encapsulation between SFFs while NSH is used to convey and trigger SFC polices.

- 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 responsible for steering traffic through the

necessary SFFs as part of the segment routing path and NSH is responsible for maintaining the service plane, and holding the SFC instance context and associated metadata.

It is of course possible to combine both of these two scenarios so as 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.

2. NSH-based SFC with SR-based Transport Tunnel

Because of the transport-independent nature of NSH-based service chains, it is expected that the NSH has broad applicability across different domains of a network. By way of illustration the various SFs involved in a service chain are available in a single data center, or spread throughout multiple locations (e.g., data centers, different POPs), depending upon the operator preference and/or availability of service resources. Regardless of where the service resources are deployed it is necessary to provide traffic steering through a set of SFFs and NSH-based service chains provide the flexibility for the network operator to choose which particular transport encapsulation to use between SFFs, which may be different depending upon which area of the network the SFFs/SFs are currently deployed. Therefore from an SFC architecture perspective, segment routing is simply one of multiple available transport encapsulations that can be used for traffic steering between SFFs. Concretely, NSH does not require to use a unique transport encapsulation when traversing a service chain. NSH-based service forwarding relies upon underlying service node capabilities.

The following three figures provide an example of an SFC established for flow F that has SF instances located in different data centers, DC1 and DC2. For the purpose of illustration, let the SFC's Service Path Identifier (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 chain).

After removing the outer transport encapsulation, that may or may not be MPLS-SR or SRv6, 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.

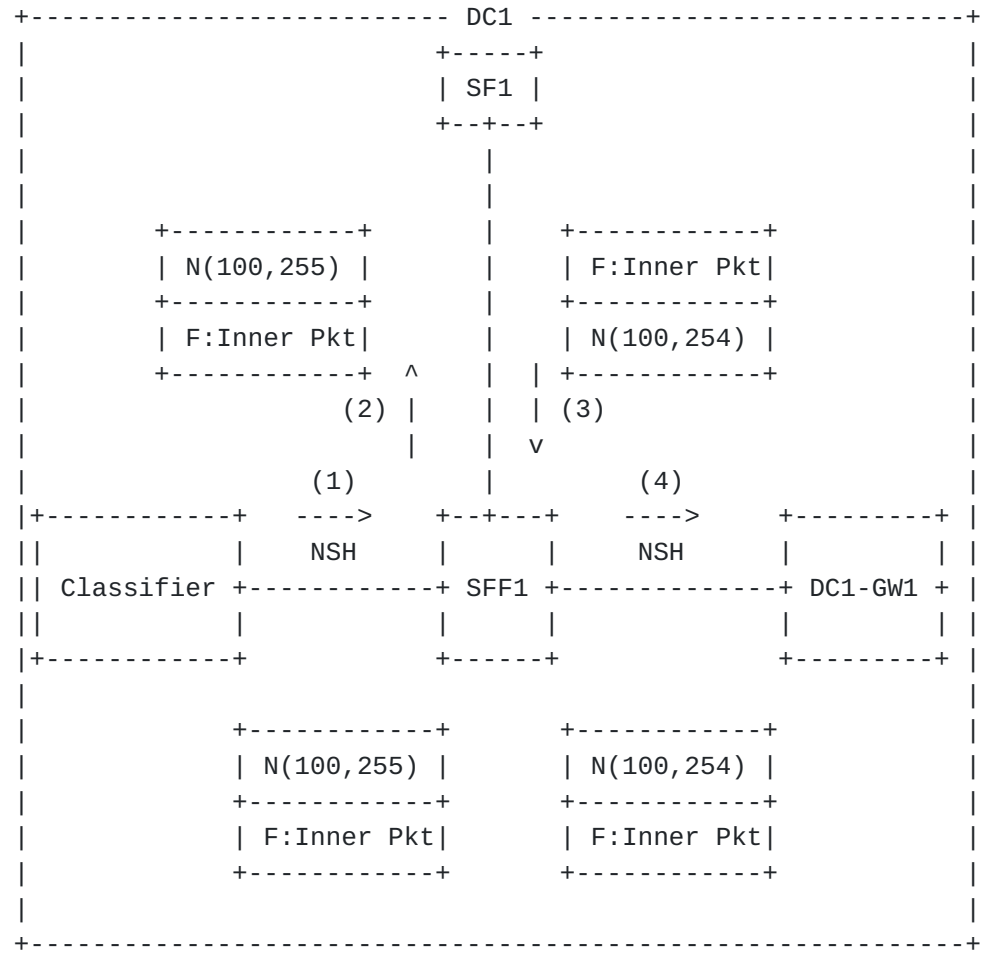

Figure 1: SR for inter-DC SFC - Part 1

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

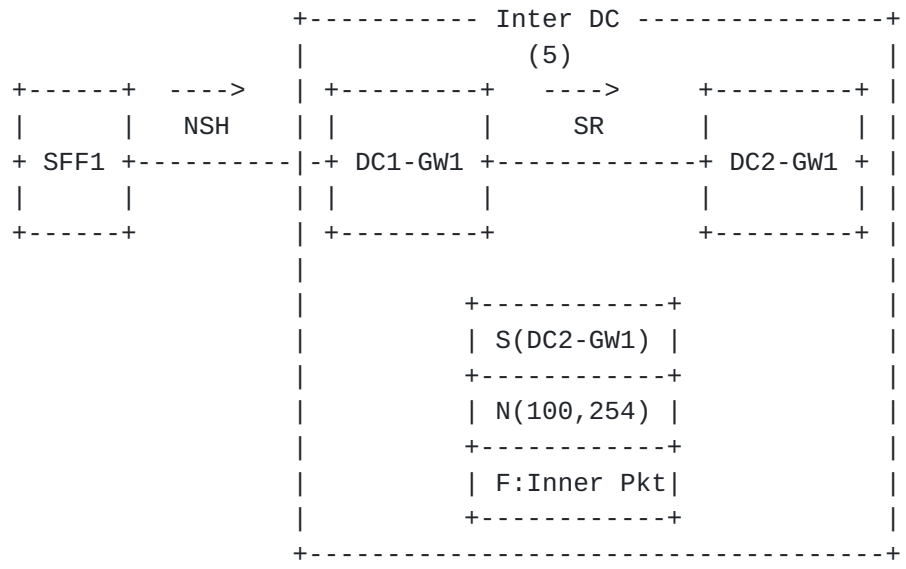

Figure 2: SR for inter-DC SFC - Part 2

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. The outer transport encapsulation may be any transport that is able to identify NSH as the next protocol.

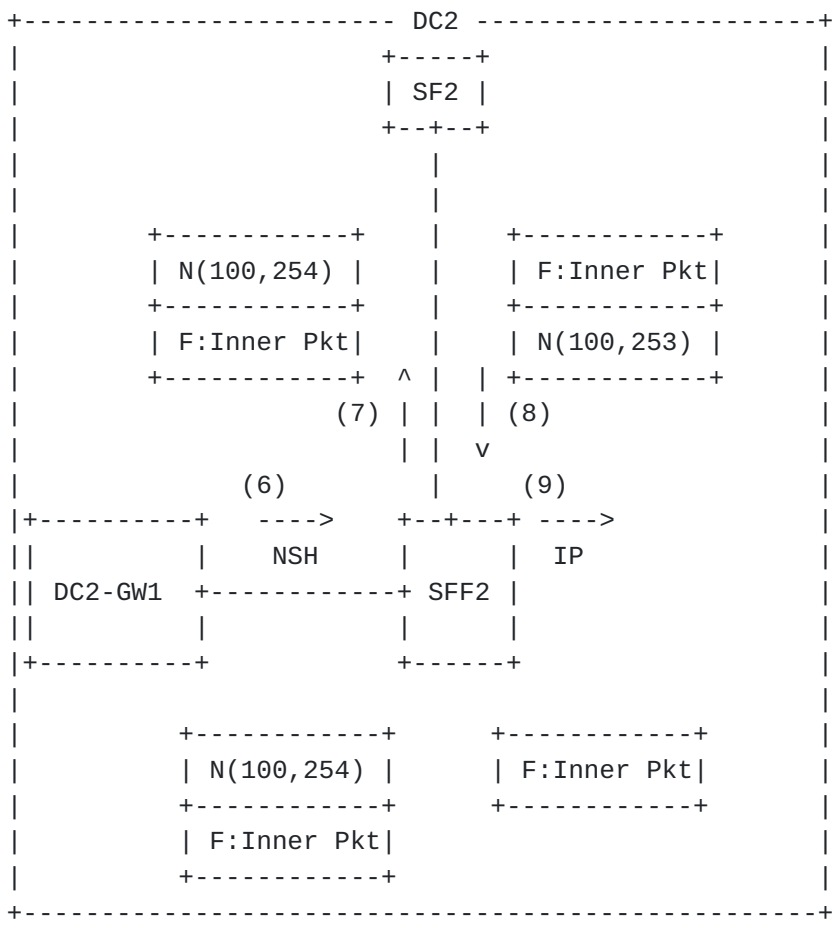

Figure 3: SR for inter-DC SFC - Part 3

The benefits of this scheme are listed hereafter:

- The network operator is able to take advantage of the transport-independent nature of the NSH encapsulation.

- The network operator is able to take advantage of the traffic steering capability of SR where appropriate.

- Light-weight NSH is used in the data center for SFC and avoids more complex hierarchical SFC schemes between data centers.

- Clear responsibility division and scope between NSH and SR.

Note that this scenario is applicable to any case where multiple segments of a service chain are distributed into 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).

3. 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 Service Function Chaining. The operation relies upon SR to perform SFF-SFF transport and NSH to provide the service plane between SFs thereby maintaining SFC context and metadata.

When a service chain is established, a packet associated with that chain will first encapsulate an NSH that will be used to maintain the end-to-end service plane through use of the SFC context. The SFC context (e.g., the service plane path referenced by the SPI) 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 (transport-specific) SR header and forwarded in the SR domain following normal SR operation.

When a packet has to be forwarded to an SF attached to an SFF, the SFF performs a lookup on the prefix SID associated with the SF to retrieve the next-hop context between the SFF and SF. E.g. to retrieve the destination MAC address in case native Ethernet encapsulation is used between SFF and SF. How the next-hop context is populated is out of the scope of this document. The SFF strips the SR information of the packet, updates the SR information, and saves it to a cache indexed by the NSH SPI. This saved SR information is used to encapsulate and forward the packet(s) coming back from the SF.

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 as the index into the cache. The SFF then pushes the 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.

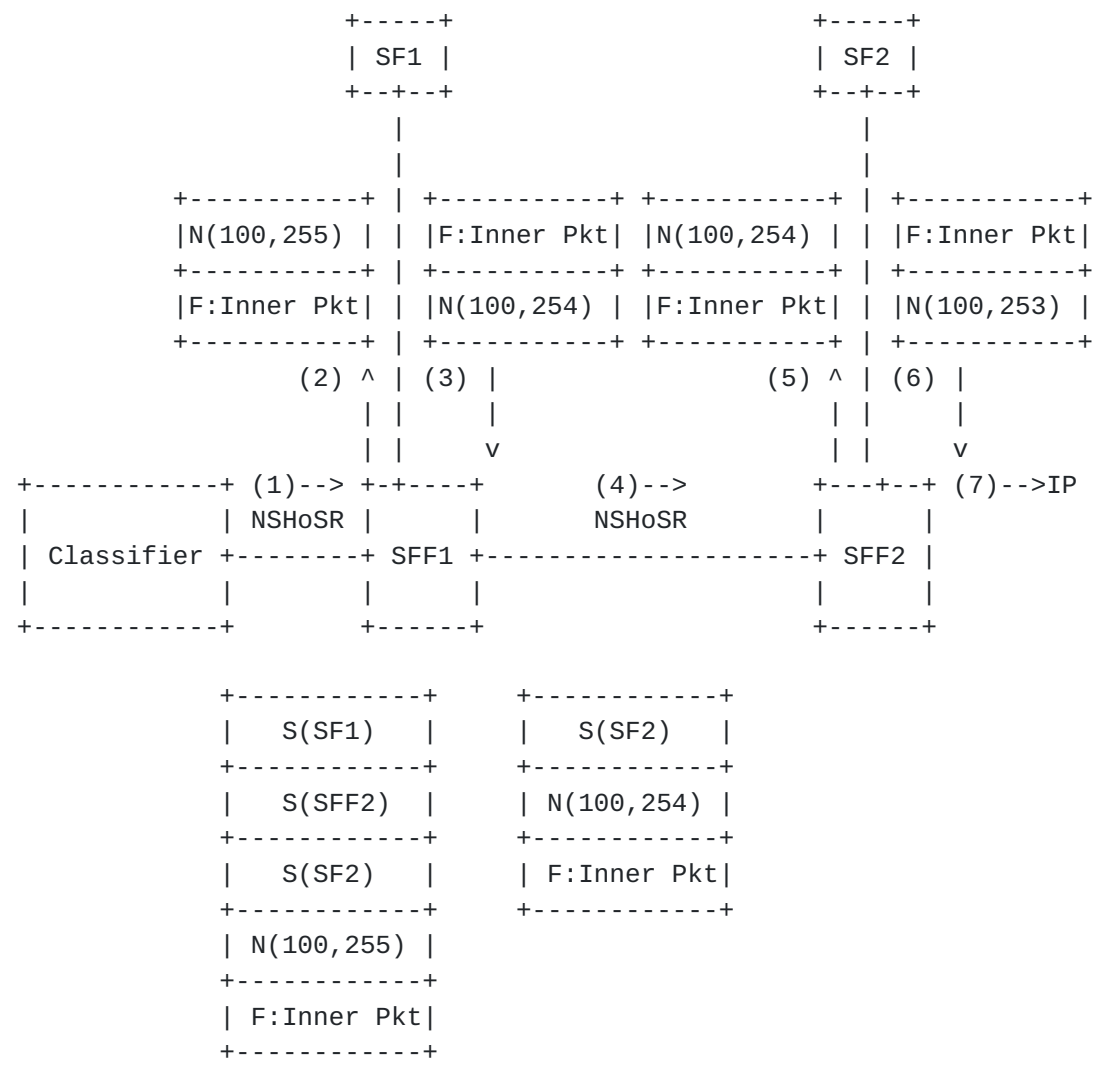

Figure 4: NSH over SR for SFC

The benefits of this scheme include:

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

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

- It provides a unique and standard way to pass metadata to SFs. Note that currently there is no solution for MPLS-SR to carry metadata and there is no solution to pass metadata to SR-unaware SFs.

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

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

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

4. Encapsulation Details

4.1. NSH using MPLS-SR Transport

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

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

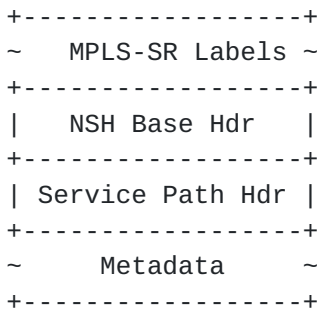

Figure 5: NSH using MPLS-SR Transport

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 MPLS-SR it is necessary to terminate the NSH-based SFC at the tail-end node of the MPLS-SR 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 MPLS-SR label stack and thereby expose NSH to the wrong SFF. 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 MPLS-SR path it is necessary to provide an indication to the tail-end that NSH follows the MPLS-SR label stack.

There are several ways to achieve this but its specification is outside the scope of this document.

4.2. NSH using SRv6 Transport

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

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].

5. 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

6. IANA Considerations

6.1. 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

6.2. 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 Extension Header	Reference
TBA3	NSH	Network Service Header	N	[ThisDocument]

7. Acknowledgments

TBD.

8. References

8.1. Normative References

[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-12 (work in progress), February 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>.

[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>.

[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>.

[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>.

[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>.

8.2. Informative References

[I-D.ietf-6man-segment-routing-header] Previdi, S., Filsfils, C., Raza, K., Dukes, D., Leddy, J., Field, B., daniel.voyer@bell.ca, d., daniel.bernier@bell.ca, d., Matsushima, S., Leung, I., Linkova, J., Aries, E., Kosugi, T., Vyncke, E., Lebrun, D., Steinberg, D., and R. Raszuk, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-09 (work in progress), March 2018.

[I-D.xuclad-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-xuclad-spring-sr-service-programming-02 (work in progress), April 2019.

[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

Haoyu Song
Futurewei Technologies
2330 Central Express Way
Santa Clara
USA

Email: haoyu.song@futurewei.com

Jeff Tantsura
Apstra inc.
USA

Email: jefftant.ietf@gmail.com

Joel Halpern
Ericsson
USA

Email: joel.halpern@ericsson.com

Wim Henderickx
Nokia
USA

Email: wim.henderickx@nokia.com

Mohamed Boucadair
Orange
USA

Email: mohamed.boucadair@orange.com

Syed Hassan
Cisco Systems
USA

Email: shassan@cisco.com