Internet Engineering Task Force (IETF)
Request for Comments: 8300
Category: Standards Track
ISSN: 2070-1721

P. Quinn, Ed.
Cisco
U. Elzur, Ed.
Intel
C. Pignataro, Ed.
Cisco

January 2018

Network Service Header (NSH)

Abstract

This document describes a Network Service Header (NSH) imposed on packets or frames to realize Service Function Paths (SFPs). The NSH also provides a mechanism for metadata exchange along the instantiated service paths. The NSH is the Service Function Chaining (SFC) encapsulation required to support the SFC architecture (defined in RFC 7665).

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Service Functions are widely deployed and essential in many networks. These Service Functions provide a range of features such as security, WAN acceleration, and server load balancing. Service Functions may be instantiated at different points in the network infrastructure such as the WAN, data center, and so forth.

Prior to development of the SFC architecture [RFC7665] and the protocol specified in this document, current Service Function deployment models have been relatively static and bound to topology for insertion and policy selection. Furthermore, they do not adapt well to elastic service environments enabled by virtualization.

New data-center network and cloud architectures require more flexible Service Function deployment models. Additionally, the transition to virtual platforms demands an agile service insertion model that supports dynamic and elastic service delivery. Specifically, the following functions are necessary:

1. The movement of Service Functions and application workloads in the network.

2. The ability to easily bind service policy to granular information, such as per-subscriber state.

3. The capability to steer traffic to the requisite Service Function(s).

This document, the Network Service Header (NSH) specification, defines a new data-plane protocol, which is an encapsulation for SFCs. The NSH is designed to encapsulate an original packet or frame and, in turn, be encapsulated by an outer transport encapsulation (which is used to deliver the NSH to NSH-aware network elements), as shown in Figure 1:

```
+------------------------------+
|    Transport Encapsulation   |
+------------------------------+
| Network Service Header (NSH) |
+------------------------------+
|    Original Packet / Frame   |
+------------------------------+
```

Figure 1: Network Service Header Encapsulation

The NSH is composed of the following elements:

1. Service Function Path identification.

2. Indication of location within a Service Function Path.

3. Optional, per-packet metadata (fixed-length or variable).

[RFC7665] provides an overview of a service chaining architecture that clearly defines the roles of the various elements and the scope of a SFC encapsulation. Figure 3 of [RFC7665] depicts the SFC architectural components after classification. The NSH is the SFC encapsulation referenced in [RFC7665].

1.1. Applicability

The NSH is designed to be easy to implement across a range of devices, both physical and virtual, including hardware platforms.

The intended scope of the NSH is for use within a single provider's operational domain. This deployment scope is deliberately constrained, as explained also in [RFC7665], and limited to a single network administrative domain. In this context, a "domain" is a set of network entities within a single administration. For example, a network administrative domain can include a single data center, or an overlay domain using virtual connections and tunnels. A corollary is that a network administrative domain has a well-defined perimeter.

An NSH-aware control plane is outside the scope of this document.

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.3. Definition of Terms

Byte: All references to "bytes" in this document refer to 8-bit bytes, or octets.

Classification: Defined in [RFC7665].

Classifier: Defined in [RFC7665].

Metadata (MD): Defined in [RFC7665]. The metadata, or context information shared between Classifiers and SFs, and among SFs, is carried on the NSH's Context Headers. It allows summarizing a classification result in the packet itself, avoiding subsequent re-classifications. Examples of metadata include classification information used for policy enforcement and network context for forwarding after service delivery.

Network Locator: Data-plane address, typically IPv4 or IPv6, used to send and receive network traffic.

Network Node/Element: Device that forwards packets or frames based on an outer header (i.e., transport encapsulation) information.

Network Overlay: Logical network built on top of an existing network (the underlay). Packets are encapsulated or tunneled to create the overlay network topology.

NSH-aware: NSH-aware means SFC-encapsulation-aware, where the NSH provides the SFC encapsulation. This specification uses NSH-aware as a more specific term from the more generic term "SFC-aware" [RFC7665].

Service Classifier: Logical entity providing classification function. Since they are logical, Classifiers may be co-resident with SFC elements such as SFs or SFFs. Service Classifiers perform classification and impose the NSH. The initial Classifier imposes the initial NSH and sends the NSH packet to the first SFF in the path. Non-initial (i.e., subsequent) classification can occur as needed and can alter, or create a new service path.

Service Function (SF): Defined in [RFC7665].

Service Function Chain (SFC): Defined in [RFC7665].

Service Function Forwarder (SFF): Defined in [RFC7665].

Service Function Path (SFP): Defined in [RFC7665].

Service Plane: The collection of SFFs and associated SFs creates a service-plane overlay in which all SFs and SFC Proxies reside [RFC7665].

SFC Proxy: Defined in [RFC7665].

1.4. Problem Space

The NSH addresses several limitations associated with Service Function deployments. [RFC7498] provides a comprehensive review of those issues.

1.5. NSH-Based Service Chaining

The NSH creates a dedicated service plane; more specifically, the NSH enables:

1. Topological Independence: Service forwarding occurs within the service plane, so the underlying network topology does not require modification. The NSH provides an identifier used to select the network overlay for network forwarding.

2. Service Chaining: The NSH enables service chaining per [RFC7665]. The NSH contains path identification information needed to realize a service path. Furthermore, the NSH provides the ability to monitor and troubleshoot a service chain, end-to-end via service-specific Operations, Administration, and Maintenance (OAM) messages. The NSH fields can be used by administrators (for example, via a traffic analyzer) to verify the path specifics (e.g., accounting, ensuring correct chaining, providing reports, etc.) of packets being forwarded along a service path.

3. The NSH provides a mechanism to carry shared metadata between participating entities and Service Functions. The semantics of the shared metadata are communicated via a control plane (which is outside the scope of this document) to participating nodes. Section 3.3 of [SFC-CONTROL-PLANE] provides an example of this. Examples of metadata include classification information used for policy enforcement and network context for forwarding post service delivery. Sharing the metadata allows Service Functions to share initial and intermediate classification results with downstream Service Functions saving re-classification, where enough information was enclosed.

4. The NSH offers a common and standards-based header for service chaining to all network and service nodes.

5. Transport Encapsulation Agnostic: The NSH is transport encapsulation independent: meaning it can be transported by a variety of encapsulation protocols. An appropriate (for a given deployment) encapsulation protocol can be used to carry NSH-encapsulated traffic. This transport encapsulation may form an

overlay network; and if an existing overlay topology provides the required service path connectivity, that existing overlay may be used.

2. Network Service Header

An NSH is imposed on the original packet/frame. This NSH contains service path information and, optionally, metadata that are added to a packet or frame and used to create a service plane. Subsequently, an outer transport encapsulation is imposed on the NSH, which is used for network forwarding.

A Service Classifier adds the NSH. The NSH is removed by the last SFF in the service chain or by an SF that consumes the packet.

2.1. Network Service Header Format

The NSH is composed of a 4-byte Base Header, a 4-byte Service Path Header, and optional Context Headers, as shown in Figure 2.

```
     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
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                Base Header                                    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                Service Path Header                            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    ~                Context Header(s)                              ~
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Network Service Header

Base Header: Provides information about the service header and the payload protocol.

Service Path Header: Provides path identification and location within a service path.

Context Header: Carries metadata (i.e., context data) along a service path.

2.2. NSH Base Header

Figure 3 depicts the NSH Base Header:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Ver|O|U|    TTL    |   Length  |U|U|U|U|MD Type| Next Protocol |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: NSH Base Header

The field descriptions are as follows:

Version: The Version field is used to ensure backward compatibility going forward with future NSH specification updates. It MUST be set to 0x0 by the sender, in this first revision of the NSH. If a packet presumed to carry an NSH header is received at an SFF, and the SFF does not understand the version of the protocol as indicated in the base header, the packet MUST be discarded, and the event SHOULD be logged. Given the widespread implementation of existing hardware that uses the first nibble after an MPLS label stack for Equal-Cost Multipath (ECMP) decision processing, this document reserves version 01b. This value MUST NOT be used in future versions of the protocol. Please see [RFC7325] for further discussion of MPLS-related forwarding requirements.

O bit: Setting this bit indicates an OAM packet (see [RFC6291]). The actual format and processing of SFC OAM packets is outside the scope of this specification (for example, see [SFC-OAM-FRAMEWORK] for one approach).

The O bit MUST be set for OAM packets and MUST NOT be set for non-OAM packets. The O bit MUST NOT be modified along the SFP.

SF/SFF/SFC Proxy/Classifier implementations that do not support SFC OAM procedures SHOULD discard packets with O bit set, but MAY support a configurable parameter to enable forwarding received SFC OAM packets unmodified to the next element in the chain. Forwarding OAM packets unmodified by SFC elements that do not support SFC OAM procedures may be acceptable for a subset of OAM functions, but it can result in unexpected outcomes for others; thus, it is recommended to analyze the impact of forwarding an OAM packet for all OAM functions prior to enabling this behavior. The configurable parameter MUST be disabled by default.

TTL: Indicates the maximum SFF hops for an SFP. This field is used for service-plane loop detection. The initial TTL value SHOULD be configurable via the control plane; the configured initial value can be specific to one or more SFPs. If no initial value is explicitly provided, the default initial TTL value of 63 MUST be used. Each SFF involved in forwarding an NSH packet MUST decrement the TTL value by 1 prior to NSH forwarding lookup. Decrementing by 1 from an incoming value of 0 shall result in a TTL value of 63. The packet MUST NOT be forwarded if TTL is, after decrement, 0.

This TTL field is the primary loop-prevention mechanism. This TTL mechanism represents a robust complement to the Service Index (see Section 2.3), as the TTL is decremented by each SFF. The handling of an incoming 0 TTL allows for better, although not perfect, interoperation with pre-standard implementations that do not support this TTL field.

Length: The total length, in 4-byte words, of the NSH including the Base Header, the Service Path Header, the Fixed-Length Context Header, or Variable-Length Context Header(s). The length MUST be 0x6 for MD Type 0x1, and it MUST be 0x2 or greater for MD Type 0x2. The length of the Network Service Header MUST be an integer multiple of 4 bytes; thus, variable-length metadata is always padded out to a multiple of 4 bytes.

Unassigned bits: All other flag fields, marked U, are unassigned and available for future use; see Section 9.1.1. Unassigned bits MUST be set to zero upon origination, and they MUST be ignored and preserved unmodified by other NSH supporting elements. At reception, all elements MUST NOT modify their actions based on these unknown bits.

Metadata (MD) Type: Indicates the format of the NSH beyond the mandatory NSH Base Header and the Service Path Header. MD Type defines the format of the metadata being carried. Please see the IANA Considerations in Section 9.1.3.

This document specifies the following four MD Type values:

0x0: This is a reserved value. Implementations SHOULD silently discard packets with MD Type 0x0.

0x1: This indicates that the format of the header includes a Fixed-Length Context Header (see Figure 5 below).

0x2: This does not mandate any headers beyond the Base Header and Service Path Header, but may contain optional Variable-Length Context Header(s). With MD Type 0x2, a length of 0x2 implies there are no Context Headers. The semantics of the Variable-Length Context Header(s) are not defined in this document. The format of the optional Variable-Length Context Headers is provided in Section 2.5.1.

0xF: This value is reserved for experimentation and testing, as per [RFC3692]. Implementations not explicitly configured to be part of an experiment SHOULD silently discard packets with MD Type 0xF.

The format of the Base Header and the Service Path Header is invariant and not affected by MD Type.

The NSH MD Type 1 and MD Type 2 are described in detail in Sections 2.4 and 2.5, respectively. NSH implementations MUST support MD Types 0x1 and 0x2 (where the length is 0x2). NSH implementations SHOULD support MD Type 0x2 with length greater than 0x2. Devices that do not support MD Type 0x2 with a length greater than 0x2 MUST ignore any optional Context Headers and process the packet without them; the Base Header Length field can be used to determine the original payload offset if access to the original packet/frame is required. This specification does not disallow the MD Type value from changing along an SFP; however, the specification of the necessary mechanism to allow the MD Type to change along an SFP are outside the scope of this document and would need to be defined for that functionality to be available. Packets with MD Type values not supported by an implementation MUST be silently dropped.

Next Protocol: Indicates the protocol type of the encapsulated data. The NSH does not alter the inner payload, and the semantics on the inner protocol remain unchanged due to NSH SFC. Please see the IANA Considerations in Section 9.1.6.

This document defines the following Next Protocol values:

0x1: IPv4
0x2: IPv6
0x3: Ethernet
0x4: NSH
0x5: MPLS
0xFE: Experiment 1
0xFF: Experiment 2

The functionality of hierarchical NSH using a Next Protocol value of 0x4 (NSH) is outside the scope of this specification. Packets with Next Protocol values not supported SHOULD be silently dropped by default, although an implementation MAY provide a configuration parameter to forward them. Additionally, an implementation not explicitly configured for a specific experiment [RFC3692] SHOULD silently drop packets with Next Protocol values 0xFE and 0xFF.

2.3. Service Path Header

Figure 4 shows the format of the Service Path Header:

```
     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
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          Service Path Identifier (SPI)        | Service Index |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

    Service Path Identifier (SPI): 24 bits
    Service Index (SI): 8 bits
```

Figure 4: NSH Service Path Header

The meaning of these fields is as follows:

Service Path Identifier (SPI): Uniquely identifies a Service Function Path (SFP). Participating nodes MUST use this identifier for SFP selection. The initial Classifier MUST set the appropriate SPI for a given classification result.

Service Index (SI): Provides location within the SFP. The initial Classifier for a given SFP SHOULD set the SI to 255; however, the control plane MAY configure the initial value of the SI as appropriate (i.e., taking into account the length of the SFP). The Service Index MUST be decremented by a value of 1 by Service Functions or by SFC Proxy nodes after performing required services; the new decremented SI value MUST be used in the egress packet's NSH. The initial Classifier MUST send the packet to the first SFF in the identified SFP for forwarding along an SFP. If re-classification occurs, and that re-classification results in a new SPI, the (re-)Classifier is, in effect, the initial Classifier for the resultant SPI.

The SI is used in conjunction with the Service Path Identifier for SFP selection and for determining the next SFF/SF in the path. The SI is also valuable when troubleshooting or reporting service paths. While the TTL provides the primary SFF-based loop prevention for this mechanism, SI decrement by SF serves as a limited loop-prevention mechanism. NSH packets, as described above, are discarded when an SFF decrements the TTL to 0. In addition, an SFF that is not the terminal SFF for an SFP will discard any NSH packet with an SI of 0, as there will be no valid next SF information.

2.4. NSH MD Type 1

When the Base Header specifies MD Type 0x1, a Fixed-Length Context Header (16-bytes) MUST be present immediately following the Service Path Header, as per Figure 5. The value of a Fixed-Length Context Header that carries no metadata MUST be set to zero.

```
     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
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Ver|O|U|    TTL    |   Length  |U|U|U|U|MD Type| Next Protocol |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          Service Path Identifier              | Service Index |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    |                 Fixed-Length Context Header                   |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: NSH MD Type 0x1

This specification does not make any assumptions about the content of the 16-byte Context Header that must be present when the MD Type field is set to 1, and it does not describe the structure or meaning of the included metadata.

An SFC-aware SF or SFC Proxy needs to receive the data structure and semantics first in order to process the data placed in the mandatory context field. The data structure and semantics include both the allocation schema and order as well as the meaning of the included data. How an SFC-aware SF or SFC Proxy gets the data structure and semantics is outside the scope of this specification.

An SF or SFC Proxy that does not know the format or semantics of the Context Header for an NSH with MD Type 1 MUST discard any packet with such an NSH (i.e., MUST NOT ignore the metadata that it cannot process), and MUST log the event at least once per the SPI for which the event occurs (subject to thresholding).

[NSH-DC-ALLOCATION] and [NSH-BROADBAND-ALLOCATION] provide specific examples of how metadata can be allocated.

2.5. NSH MD Type 2

When the Base Header specifies MD Type 0x2, zero or more Variable-Length Context Headers MAY be added, immediately following the Service Path Header (see Figure 6). Therefore, Length = 0x2, indicates that only the Base Header and Service Path Header are present (and in that order). The optional Variable-Length Context Headers MUST be of an integer number of 4-bytes. The Base Header Length field MUST be used to determine the offset to locate the original packet or frame for SFC nodes that require access to that information.

```
     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
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Ver|O|U|    TTL    |   Length  |U|U|U|U|MD Type| Next Protocol |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          Service Path Identifier              | Service Index |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    ~              Variable-Length Context Headers  (opt.)          ~
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: NSH MD Type 0x2

2.5.1. Optional Variable-Length Metadata

The format of the optional Variable-Length Context Headers, is as depicted in Figure 7.

```
     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
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          Metadata Class       |      Type     |U|    Length   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                   Variable-Length Metadata                    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: Variable-Length Context Headers

Metadata Class (MD Class): Defines the scope of the Type field to provide a hierarchical namespace. Section 9.1.4 defines how the MD Class values can be allocated to standards bodies, vendors, and others.

Type: Indicates the explicit type of metadata being carried. The definition of the Type is the responsibility of the MD Class owner.

Unassigned bit: One unassigned bit is available for future use. This bit MUST NOT be set, and it MUST be ignored on receipt.

Length: Indicates the length of the variable-length metadata, in bytes. In case the metadata length is not an integer number of 4-byte words, the sender MUST add pad bytes immediately following the last metadata byte to extend the metadata to an integer number of 4-byte words. The receiver MUST round the Length field up to the nearest 4-byte-word boundary, to locate and process the next field in the packet. The receiver MUST access only those bytes in the metadata indicated by the Length field (i.e., actual number of bytes) and MUST ignore the remaining bytes up to the nearest 4-byte-word boundary. The length may be 0 or greater.

A value of 0 denotes a Context Header without a Variable-Length Metadata field.

This specification does not make any assumption about Context Headers that are mandatory to implement or those that are mandatory to process. These considerations are deployment specific. However, the control plane is entitled to instruct SFC-aware SFs with the data structure of the Context Header together with its scoping (see e.g., Section 3.3.3 of [SFC-CONTROL-PLANE]).

Upon receipt of a packet that belongs to a given SFP, if a mandatory-to-process Context Header is missing in that packet, the SFC-aware SF MUST NOT process the packet and MUST log an error at least once per the SPI for which the mandatory metadata is missing.

If multiple mandatory-to-process Context Headers are required for a given SFP, the control plane MAY instruct the SFC-aware SF with the order to consume these Context Headers. If no instructions are provided and the SFC-aware SF will make use of or modify the specific Context Header, then the SFC-aware SF MUST process these Context Headers in the order they appear in an NSH packet.

If multiple instances of the same metadata are included in an NSH packet, but the definition of that Context Header does not allow for it, the SFC-aware SF MUST process the first instance and ignore subsequent instances. The SFC-aware SF MAY log or increase a counter for this event.

3. NSH Actions

NSH-aware nodes (which include Service Classifiers, SFFs, SFs, and SFC Proxies) may alter the contents of the NSH headers. These nodes have several possible NSH-related actions:

1. Insert or remove the NSH: These actions can occur respectively at the start and end of a service path. Packets are classified, and if determined to require servicing, an NSH will be imposed. A

   Service Classifier MUST insert an NSH at the start of an SFP. An imposed NSH MUST contain both a valid Base Header and Service Path Header. At the end of an SFP, an SFF MUST remove the NSH before forwarding or delivering the un-encapsulated packet. Therefore, it is the last node operating on the service header.

   Multiple logical Classifiers may exist within a given service path. Non-initial Classifiers may re-classify data, and that re-classification MAY result in the selection of a different SFP. When the logical Classifier performs re-classification that results in a change of service path, it MUST replace the existing NSH with a new NSH with the Base Header and Service Path Header reflecting the new service path information and MUST set the initial SI. The O bit, the TTL field, and unassigned flags MUST be copied transparently from the old NSH to a new NSH. Metadata MAY be preserved in the new NSH.

2. Select service path: The Service Path Header provides service path information and is used by SFFs to determine correct service path selection. SFFs MUST use the Service Path Header for selecting the next SF or SFF in the service path.

3. Update the NSH: SFs MUST decrement the service index by one. If an SFF receives a packet with an SPI and SI that do not correspond to a valid next hop in a valid SFP, that packet MUST be dropped by the SFF.

   Classifiers MAY update Context Headers if new/updated context is available.

   If an SFC proxy is in use (acting on behalf of an NSH-unaware Service Function for NSH actions), then the proxy MUST update the Service Index and MAY update contexts. When an SFC Proxy receives an NSH-encapsulated packet, it MUST remove the NSH before forwarding it to an NSH-unaware SF. When the SFC Proxy receives a packet back from an NSH-unaware SF, it MUST re-encapsulate it with the correct NSH, and it MUST decrement the Service Index by one.

4. Service policy selection: Service Functions derive policy (i.e., service actions such as permit or deny) selection and enforcement from the NSH. Metadata shared in the NSH can provide a range of service-relevant information such as traffic classification.

Figure 8 maps each of the four actions above to the components in the SFC architecture that can perform it.

Component	Insert, remove, or replace the NSH			Forward the NSH packets	Update the NSH		Service policy sel.
	Insert	Remove	Replace		Dec. Service Index	Update Context Header	
Classifier	+		+			+	
Service Function Forwarder (SFF)		+		+			
Service Function (SF)					+	+	+
SFC Proxy	+	+			+	+	

Figure 8: NSH Action and Role Mapping

4. NSH Transport Encapsulation

Once the NSH is added to a packet, an outer transport encapsulation is used to forward the original packet and the associated metadata to the start of a service chain. The encapsulation serves two purposes:

1. Creates a topologically independent services plane. Packets are forwarded to the required services without changing the underlying network topology.

2. Transit network nodes simply forward the encapsulated packets without modification.

The service header is independent of the transport encapsulation used. Existing transport encapsulations can be used. The presence of an NSH is indicated via a protocol type or another indicator in the outer transport encapsulation.

5. Fragmentation Considerations

The NSH and the associated transport encapsulation header are "added" to the encapsulated packet/frame. This additional information increases the size of the packet.

Within a managed administrative domain, an operator can ensure that the underlay MTU is sufficient to carry SFC traffic without requiring fragmentation. Given that the intended scope of the NSH is within a single provider's operational domain, that approach is sufficient.

However, although explicitly outside the scope of this specification, there might be cases where the underlay MTU is not large enough to carry the NSH traffic. Since the NSH does not provide fragmentation support at the service plane, the transport encapsulation protocol ought to provide the requisite fragmentation handling. For instance, Section 9 of [RTG-ENCAP] provides exemplary approaches and guidance for those scenarios.

When the transport encapsulation protocol supports fragmentation, and fragmentation procedures needs to be used, such fragmentation is part of the transport encapsulation logic. If, as it is common, fragmentation is performed by the endpoints of the transport encapsulation, then fragmentation procedures are performed at the sending NSH entity as part of the transport encapsulation, and reassembly procedures are performed at the receiving NSH entity during transport de-encapsulation handling logic. In no case would such fragmentation result in duplication of the NSH header.

For example, when the NSH is encapsulated in IP, IP-level fragmentation coupled with Path MTU Discovery (PMTUD) (e.g., [RFC8201]) is used. Since PMTUD relies on ICMP messages, an operator should ensure ICMP packets are not blocked. When, on the other hand, the underlay does not support fragmentation procedures, an error message SHOULD be logged when dropping a packet too big. Lastly, NSH-specific fragmentation and reassembly methods may be defined as well, but these methods are outside the scope of this document and subject for future work.

6. Service Path Forwarding with NSH

6.1. SFFs and Overlay Selection

As described above, the NSH contains a Service Path Identifier (SPI) and a Service Index (SI). The SPI is, as per its name, an identifier. The SPI alone cannot be used to forward packets along a service path. Rather, the SPI provides a level of indirection between the service path / topology and the network transport encapsulation. Furthermore, there is no requirement for, or expectation of, an SPI being bound to a predetermined or static network path.

The Service Index provides an indication of location within a service path. The combination of SPI and SI provides the identification of a logical SF and its order within the service plane. This combination is used to select the appropriate network locator(s) for overlay forwarding. The logical SF may be a single SF or a set of eligible SFs that are equivalent. In the latter case, the SFF provides load distribution amongst the collection of SFs as needed.

SI serves as a mechanism for detecting invalid SFPs. In particular, an SI value of zero indicates that forwarding is incorrect and the packet must be discarded.

This indirection -- SPI to overlay -- creates a true service plane. That is, the SFF/SF topology is constructed without impacting the network topology, but, more importantly, service-plane-only participants (i.e., most SFs) need not be part of the network overlay topology and its associated infrastructure (e.g., control plane, routing tables, etc.). SFs need to be able to return a packet to an appropriate SFF (i.e., has the requisite NSH information) when service processing is complete. This can be via the overlay or underlay and, in some cases, can require additional configuration on the SF. As mentioned above, an existing overlay topology may be used, provided it offers the requisite connectivity.

The mapping of SPI to transport encapsulation occurs on an SFF (as discussed above, the first SFF in the path gets an NSH encapsulated packet from the Classifier). The SFF consults the SPI/ID values to determine the appropriate overlay transport encapsulation protocol (several may be used within a given network) and next hop for the requisite SF. Table 1 depicts an example of a single next-hop SPI/SI-to-network overlay network locator mapping.

SPI	SI	Next Hop(s)	Transport Encapsulation
10	255	192.0.2.1	VXLAN-gpe
10	254	198.51.100.10	GRE
10	251	198.51.100.15	GRE
40	251	198.51.100.15	GRE
50	200	01:23:45:67:89:ab	Ethernet
15	212	Null (end of path)	None

Table 1: SFF NSH Mapping Example

Additionally, further indirection is possible: the resolution of the required SF network locator may be a localized resolution on an SFF, rather than an SFC control plane responsibility, as per Tables 2 and 3.

Please note: VXLAN-gpe and GRE in the above table refer to [VXLAN-GPE] and [RFC2784] [RFC7676], respectively.

SPI	SI	Next Hop(s)
10	3	SF2
245	12	SF34
40	9	SF9

Table 2: NSH-to-SF Mapping Example

SF	Next Hop(s)	Transport Encapsulation
SF2	192.0.2.2	VXLAN-gpe
SF34	198.51.100.34	UDP
SF9	2001:db8::1	GRE

Table 3: SF Locator Mapping Example

Since the SPI is a representation of the service path, the lookup may return more than one possible next hop within a service path for a given SF, essentially a series of weighted (equally or otherwise) paths to be used (for load distribution, redundancy, or policy); see Table 4. The metric depicted in Table 4 is an example to help illustrate weighing SFs. In a real network, the metric will range from a simple preference (similar to routing next-hop) to a true dynamic composite metric based on the state of a Service Function (including load, session state, capacity, etc.).

SPI	SI	NH	Metric
10	3	203.0.113.1	1
		203.0.113.2	1
20	12	192.0.2.1	1
		203.0.113.4	1
30	7	192.0.2.10	10
		198.51.100.1	5

(encapsulation type omitted for formatting)

Table 4: NSH Weighted Service Path

The information contained in Tables 1-4 may be received from the control plane, but the exact mechanism is outside the scope of this document.

6.2. Mapping the NSH to Network Topology

As described above, the mapping of the SPI to network topology may result in a single path, or it might result in a more complex topology. Furthermore, the SPI-to-overlay mapping occurs at each SFF independently. Any combination of topology selection is possible. Please note, there is no requirement to create a new overlay topology if a suitable one already exists. NSH packets can use any (new or existing) overlay, provided the requisite connectivity requirements are satisfied.

Examples of mapping for a topology:

1. Next SF is located at SFFb with locator 2001:db8::1
   SFFa mapping: SPI=10 --> VXLAN-gpe, dst-ip: 2001:db8::1

2. Next SF is located at SFFc with multiple network locators for load-distribution purposes:
   SFFb mapping: SPI=10 --> VXLAN-gpe, dst_ip:203.0.113.1, 203.0.113.2, 203.0.113.3, equal cost

3. Next SF is located at SFFd with two paths from SFFc, one for redundancy:
   SFFc mapping: SPI=10 --> VXLAN-gpe, dst_ip:192.0.2.10 cost=10, 203.0.113.10, cost=20

In the above example, each SFF makes an independent decision about the network overlay path and policy for that path. In other words, there is no a priori mandate about how to forward packets in the network (only the order of services that must be traversed).

The network operator retains the ability to engineer the network paths as required. For example, the overlay path between SFFs may utilize traffic engineering, QoS marking, or ECMP, without requiring complex configuration and network protocol support to be extended to the service path explicitly. In other words, the network operates as expected, and evolves as required, as does the service plane.

6.3. Service Plane Visibility

The SPI and SI serve an important function for visibility into the service topology. An operator can determine what service path a packet is "on" and its location within that path simply by viewing NSH information (packet capture, IP Flow Information Export (IPFIX), etc.). The information can be used for service scheduling and placement decisions, troubleshooting, and compliance verification.

6.4. Service Graphs

While a given realized SFP is a specific sequence of Service Functions, the service, as seen by a user, can actually be a collection of SFPs, with the interconnection provided by Classifiers (in-service path, non-initial re-classification). These internal re-Classifiers examine the packet at relevant points in the network, and, if needed, SPI and SI are updated (whether this update is a re-write, or the imposition of a new NSH with new values is implementation specific) to reflect the "result" of the classification. These Classifiers may, of course, also modify the metadata associated with the packet.
Section 2.1 of [RFC7665] describes Service Graphs in detail.

7. Policy Enforcement with NSH

7.1. NSH Metadata and Policy Enforcement

As described in Section 2, NSH provides the ability to carry metadata along a service path. This metadata may be derived from several sources. Common examples include:

Network nodes/devices: Information provided by network nodes can indicate network-centric information (such as VPN Routing and Forwarding (VRF) or tenant) that may be used by Service Functions or conveyed to another network node post service path egress.

External (to the network) systems: External systems, such as orchestration systems, often contain information that is valuable for Service Function policy decisions. In most cases, this information cannot be deduced by network nodes. For example, a cloud orchestration platform placing workloads "knows" what application is being instantiated and can communicate this information to all NSH nodes via metadata carried in the Context Header(s).

Service Functions: A Classifier co-resident with Service Functions often performs very detailed and valuable classification.

Regardless of the source, metadata reflects the "result" of classification. The granularity of classification may vary. For example, a network switch, acting as a Classifier, might only be able to classify based on a 2-tuple, or based on a 5-tuple, while a Service Function may be able to inspect application information. Regardless of granularity, the classification information can be represented in the NSH.

Once the data is added to the NSH, it is carried along the service path. NSH-aware SFs receive the metadata, and can use that metadata for local decisions and policy enforcement. Figures 9 and 10 highlight the relationship between metadata and policy.

```
                +-------+        +-------+        +-------+
                |  SFF  )------->(  SFF  |------->|  SFF  |
                +---+---+        +---+---+        +---+---+
                    ^                |                |
                  ,-|-.            ,-|-.            ,-|-.
                 /     \          /     \          /     \
                ( Class )        (  SF1  )        (  SF2  )
                 \ ify /          \     /          \     /
                  `---'            `---'            `---'
                 5-tuple:        Permit             Inspect
                 Tenant A        Tenant A           AppY
                 AppY
```

Figure 9: Metadata and Policy

```
               +-----+          +-----+            +-----+
               | SFF |---------> | SFF |----------> | SFF |
               +--+--+          +--+--+            +--+--+
                  ^                |                  |
                ,-+-.            ,-+-.              ,-+-.
               /     \          /     \            /     \
              ( Class )        (  SF1  )          (  SF2  )
               \ ify /          \     /            \     /
                `-+-'            `---'              `---'
                  |              Permit             Deny AppZ
              +---+---+          employees
              |       |
              +-------+
              External
              system:
              Employee
              AppZ
```

Figure 10: External Metadata and Policy

In both of the examples above, the Service Functions perform policy decisions based on the result of the initial classification: the SFs did not need to perform re-classification; instead, they rely on an antecedent classification for local policy enforcement.

Depending on the information carried in the metadata, data privacy impact needs to be considered. For example, if the metadata conveys tenant information, that information may need to be authenticated and/or encrypted between the originator and the intended recipients (which may include intended SFs only); one approach to an optional capability to do this is explored in [NSH-ENCRYPT]. The NSH itself does not provide privacy functions, rather it relies on the transport encapsulation/overlay. An operator can select the appropriate set of transport encapsulation protocols to ensure confidentiality (and other security) considerations are met. Metadata privacy and security considerations are a matter for the documents that define metadata format.

7.2. Updating/Augmenting Metadata

Post-initial metadata imposition (typically, performed during initial service path determination), the metadata may be augmented or updated:

1. Metadata Augmentation: Information may be added to the NSH's existing metadata, as depicted in Figure 11. For example, if the initial classification returns the tenant information, a secondary classification (perhaps co-resident with deep packet inspection (DPI) or server load balancing (SLB)) may augment the tenant classification with application information, and impose that new information in NSH metadata. The tenant classification is still valid and present, but additional information has been added to it.

2. Metadata Update: Subsequent Classifiers may update the initial classification if it is determined to be incorrect or not descriptive enough. For example, the initial Classifier adds metadata that describes the traffic as "Internet", but a security Service Function determines that the traffic is really "attack". Figure 12 illustrates an example of updating metadata.

```
               +-----+           +-----+            +-----+
               | SFF |---------> | SFF |----------> | SFF |
               +--+--+           +--+--+            +--+--+
                  ^                 |                  |
                ,---.             ,---.              ,---.
               /     \           /     \            /     \
              ( Class )         (  SF1  )          (  SF2  )
               \     /           \     /            \     /
                `-+-'             `---'              `---'
                  |              Inspect           Deny
              +---+---+          employees         employee+
              |       |          Class=AppZ        appZ
              +-------+
              External
              system:
              Employee
```

Figure 11: Metadata Augmentation

```
                +-----+           +-----+            +-----+
                | SFF |---------> | SFF |----------> | SFF |
                +--+--+           +--+--+            +--+--+
                   ^                 |                  |
                 ,---.             ,---.              ,---.
                /     \           /     \            /     \
               ( Class )         (  SF1  )          (  SF2  )
                \     /           \     /            \     /
                 `---'             `---'              `---'
              5-tuple:            Inspect             Deny
              Tenant A            Tenant A            attack
                                   --> attack
```

Figure 12: Metadata Update

7.3. Service Path Identifier and Metadata

Metadata information may influence the service path selection since the Service Path Identifier values can represent the result of classification. A given SPI can be defined based on classification results (including metadata classification). The imposition of the SPI and SI results in the packet being placed on the newly specified SFP at the position indicated by the imposed SPI and SI.

This relationship provides the ability to create a dynamic service plane based on complex classification, without requiring each node to be capable of such classification or requiring a coupling to the network topology. This yields Service Graph functionality as described in Section 6.4. Figure 13 illustrates an example of this behavior.

```
               +-----+           +-----+            +-----+
               | SFF |---------> | SFF |------+---> | SFF |
               +--+--+           +--+--+      |     +--+--+
                  |                 |         |        |
                ,---.             ,---.       |      ,---.
               /     \           / SF1 \      |     /     \
              (  SCL  )         (   +   )     |    (  SF2  )
               \     /           \SCL2 /      |     \     /
                `---'             `---'    +-----+   `---'
             5-tuple:            Inspect   | SFF |    Original
             Tenant A            Tenant A  +--+--+    next SF
                                  --> DoS     |
                                              V
                                            ,-+-.
                                           /     \
                                          (  SF10 )
                                           \     /
                                            `---'
                                             DoS
                                          "Scrubber"

             Legend:
             SCL = Service Classifier
```

Figure 13: Path ID and Metadata

Specific algorithms for mapping metadata to an SPI are outside the scope of this document.

8. Security Considerations

NSH security must be considered in the contexts of the SFC architecture and operators' environments. One important characteristic of NSH is that it is not an end-to-end protocol. As opposed to a protocol that "starts" on a host and "ends" on a server or another host, NSH is typically imposed by a network device on ingress to the SFC domain and removed at the egress of the SFC domain. As such, and as with any other network-centric protocols (e.g., IP Tunneling, Traffic Engineering, MPLS, or Provider-Provisioned Virtual Private Networks), there is an underlying trust in the network devices responsible for imposing, removing, and acting on NSH information.

The following sections detail an analysis and present a set of requirements and recommendations in those two areas.

8.1. NSH Security Considerations from Operators' Environments

Trusted Devices

All Classifiers, SFFs and SFs (hereinafter referred to as "SFC devices") within an operator's environment are assumed to have been selected, vetted, and actively maintained; therefore, they are trusted by that operator. This assumption differs from the oft held view that devices are untrusted, often referred to as the "zero-trust model". Operators SHOULD regularly monitor (i.e., continuously audit) these devices to help ensure compliant behavior. This trust, therefore, extends into NSH operations: SFC devices are not, themselves, considered to be attack vectors. This assumption, and the resultant conclusion is reasonable since this is the very basis of an operator posture; the operator depends on this reality to function. If these devices are not trusted, and indeed are compromised, almost the entirety of the operator's standard-based IP and MPLS protocol suites are vulnerable; therefore, the operation of the entire network is compromised. Although there are well-documented monitoring-based methods for detecting compromise (such as included continuous monitoring and audit and log review), these may not be sufficient to contain damage by a completely compromised element.

Methods and best practices to secure devices are also widely documented and outside the scope of this document.

Single Domain Boundary

As per [RFC7665], NSH is designed for use within a single administrative domain. This scoping provides two important characteristics:

i) Clear NSH boundaries

NSH egress devices MUST strip the NSH headers before they send the users' packets or frames out of the NSH domain.

Means to prevent leaking privacy-related information outside an administrative domain are natively supported by the NSH given that the last SFF of a service path will systematically remove the NSH encapsulation before forwarding a packet exiting the service path.

The second step in such prevention is to filter the transport encapsulation protocol used by NSH at the domain edge. The transport encapsulation protocol MUST be filtered and MUST NOT leave the domain edge.

Depending upon the transport encapsulation protocol used for NSH, this can be done either by completely blocking the transport encapsulation (e.g., if MPLS is the chosen NSH transport encapsulation protocol, it is therefore never allowed to leave the domain) or by examining the carried protocol with the transport encapsulation (e.g., if VXLAN-gpe is used as the NSH transport encapsulation protocol, all domain edges need to filter based on the carried protocol in the VXLAN-gpe.)

The other consequence of this bounding is that ingress packets MUST also be filtered to prevent attackers from sending in NSH packets with service path identification and metadata of their own selection. The same filters as described above for both the NSH at SFC devices and for the transport encapsulation protocol as general edge protections MUST be applied on ingress.

In summary, packets originating outside the SFC-enabled domain MUST be dropped if they contain an NSH. Similarly, packets exiting the SFC-enabled domain MUST be dropped if they contain an NSH.

ii) Mitigation of external threats

As per the trusted SFC device points raised above, given that NSH is scoped within an operator's domain, that operator can ensure that the environment and its transitive properties comply with that operator's required security posture. Continuous audits for assurance are recommended with this reliance on a fully trusted environment. The term "continuous audits" describes a method (automated or manual) of checking security-control compliance on a regular basis, at some set period of time.

8.2. NSH Security Considerations from the SFC Architecture

The SFC architecture defines functional roles (e.g., SFF), as well as protocol elements (e.g., Metadata). This section considers each role and element in the context of threats posed in the areas of integrity and confidentiality. As with routing, the distributed computation model assumes a distributed trust model.

An important consideration is that NSH contains mandatory-to-mute fields, and further, the SFC architecture describes cases where other fields in NSH change, all on a possible SFP hop-by-hop basis. This means that any cryptographic solution requires complex key distribution and life-cycle operations.

8.2.1. Integrity

SFC devices

SFC devices MAY perform various forms of verification on received NSH packets such as only accepting NSH packets from expected devices, checking that NSH SPI and SI values received from expected devices conform to expected values and so on. Implementation of these additional checks are a local matter and, thus, out of scope of this document.

NSH Base and Service Path Headers

Attackers who can modify packets within the operator's network may be able to modify the SFP, path position, and/or the metadata associated with a packet.

One specific concern is an attack in which a malicious modification of the SPI/SI results in an alteration of the path to avoid security devices. The options discussed in this section help thwart that attack, and so does the use of the optional "Proof of Transit" method [PROOF-OF-TRANSIT].

As stated above, SFC devices are trusted; in the case where an SFC device is compromised, NSH integrity protection would be subject to forging (in many cases) as well.

NSH itself does not mandate protocol-specific integrity protection. However, if an operator deems protection is required, several options are viable:

1. SFF/SF NSH verification

   Although, strictly speaking, not integrity protection, some of the techniques mentioned above, such as checking expected NSH values are received from expected SFC device(s), can provide a form of verification without incurring the burden of a full-fledged integrity-protection deployment.

2. Transport Security

   NSH is always encapsulated by an outer transport encapsulation as detailed in Section 4 of this specification, and as depicted in Figure 1. If an operator deems cryptographic integrity protection necessary due to their risk analysis, then an outer transport encapsulation that provides such protection [RFC6071], such as IPsec, MUST be used.

Although the threat model and recommendations of Section 5 of BCP 72 [RFC3552] would normally require cryptographic data origin authentication for the header, this document does not mandate such mechanisms in order to reflect the operational and technical realities of deployment.

Given that NSH is transport independent, as mentioned above, a secure transport, such as IPsec can be used for carry NSH. IPsec can be used either alone or in conjunction with other transport encapsulation protocols, in turn, encapsulating NSH.

Operators MUST ensure the selected transport encapsulation protocol can be supported by the transport encapsulation/ underlay of all relevant network segments as well as SFFs, SFs, and SFC Proxies in the service path.

If connectivity between SFC-enabled devices traverses the public Internet, then such connectivity MUST be secured at the transport encapsulation layer. IPsec is an example of such a transport.

3. NSH Variable Header-Based Integrity

Lastly, NSH MD Type 2 provides, via variable-length headers, the ability to append cryptographic integrity protection to the NSH packet. The implementation of such a scheme is outside the scope of this document.

NSH metadata

As with the Base and Service Path Headers, if an operator deems cryptographic integrity protection needed, then an existing, standard transport protocol MUST be used since the integrity protection applies to entire encapsulated NSH packets. As mentioned above, a risk assessment that deems data-plane traffic subject to tampering will apply not only to NSH but to the transport information; therefore, the use of a secure transport is likely needed already to protect the entire stack.

If an MD Type 2 variable header integrity scheme is in place, then the integrity of the metadata can be ensured via that mechanism as well.

8.2.2. Confidentiality

SFC devices

SFC devices can "see" (and need to use) NSH information.

NSH Base and Service Path Headers

SPI and other base / service path information does not typically require confidentiality; however, if an operator does deem confidentiality to be required, then, as with integrity, an existing transport encapsulation that provides encryption MUST be utilized.

NSH metadata

An attacker with access to the traffic in an operator's network can potentially observe the metadata NSH carries with packets, potentially discovering privacy-sensitive information.

Much of the metadata carried by NSH is not sensitive. It often reflects information that can be derived from the underlying packet or frame. Direct protection of such information is not necessary, as the risks are simply those of carrying the underlying packet or frame.

Implementers and operators MUST be aware that metadata can have privacy implications, and those implications are sometimes hard to predict. Therefore, attached metadata should be limited to that necessary for correct operation of the SFP. Further, [RFC8165] defines metadata considerations that operators can take into account when using NSH.

Protecting NSH metadata information between SFC components can be done using transport encapsulation protocols with suitable security capabilities, along the lines discussed above. If a security analysis deems these protections necessary, then security features in the transport encapsulation protocol (such as IPsec) MUST be used.

One useful element of providing privacy protection for sensitive metadata is described under the "SFC Encapsulation" area of the Security Considerations of [RFC7665]. Operators can and should use indirect identification for metadata deemed to be sensitive (such as personally identifying information), significantly mitigating the risk of a privacy violation. In particular, subscriber-identifying information should be handled carefully, and, in general, SHOULD be obfuscated.

For those situations where obfuscation is either inapplicable or judged to be insufficient, an operator can also encrypt the metadata. An approach to an optional capability to do this was explored in [NSH-ENCRYPT]. For other situations where greater assurance is desired, optional mechanisms such as [PROOF-OF-TRANSIT] can be used.

9. IANA Considerations

9.1. NSH Parameters

IANA has created a new "Network Service Header (NSH) Parameters" registry. The following subsections detail new registries within the "Network Service Header (NSH) Parameters" registry.

9.1.1. NSH Base Header Bits

There are five unassigned bits (U bits) in the NSH Base Header, and one assigned bit (O bit). New bits are assigned via Standards Action [RFC8126].

Bit 2 - O (OAM) bit
Bit 3 - Unassigned
Bits 16-19 - Unassigned

9.1.2. NSH Version

IANA has set up the "NSH Version" registry. New values are assigned via Standards Action [RFC8126].

Version	Description	Reference
Version 00b	Protocol as defined by RFC 8300	RFC 8300
Version 01b	Reserved	RFC 8300
Version 10b	Unassigned	
Version 11b	Unassigned	

Table 5: NSH Version

9.1.3. NSH MD Types

IANA has set up the "NSH MD Types" registry, which contains 4-bit values. MD Type values 0x0, 0x1, 0x2, and 0xF are specified in this document; see Table 6. Registry entries are assigned via the "IETF Review" policy defined in RFC 8126 [RFC8126].

MD Type	Description	Reference
0x0	Reserved	RFC 8300
0x1	NSH MD Type 1	RFC 8300
0x2	NSH MD Type 2	RFC 8300
0x3 - 0xE	Unassigned	
0xF	Experimentation	RFC 8300

Table 6: MD Type Values

9.1.4. NSH MD Class

IANA has set up the "NSH MD Class" registry, which contains 16-bit values. New allocations are to be made according to the following policies:

0x0000 to 0x01ff: IETF Review
0x0200 to 0xfff5: Expert Review

IANA has assigned the values as follows:

Value	Meaning	Reference
0x0000	IETF Base NSH MD Class	RFC 8300
0xfff6 to 0xfffe	Experimental	RFC 8300
0xffff	Reserved	RFC 8300

Table 7: NSH MD Class

A registry for Types for the MD Class of 0x0000 is defined in Section 9.1.5.

Designated Experts evaluating new allocation requests from the "Expert Review" range should principally consider whether a new MD class is needed compared to adding MD Types to an existing class. The Designated Experts should also encourage the existence of an associated and publicly visible registry of MD Types although this registry need not be maintained by IANA.

When evaluating a request for an allocation, the Expert should verify that the allocation plan includes considerations to handle privacy and security issues associated with the anticipated individual MD Types allocated within this class. These plans should consider, when appropriate, alternatives such as indirection, encryption, and limited-deployment scenarios. Information that can't be directly derived from viewing the packet contents should be examined for privacy and security implications.

9.1.5. NSH IETF-Assigned Optional Variable-Length Metadata Types

The Type values within the IETF Base NSH MD Class, i.e., when the MD Class is set to 0x0000 (see Section 9.1.4), are the Types owned by the IETF. Per this document, IANA has created a registry for the Type values for the IETF Base NSH MD Class called the "NSH IETF-Assigned Optional Variable-Length Metadata Types" registry, as specified in Section 2.5.1.

The type values are assigned via Standards Action [RFC8126].

No initial values are assigned at the creation of the registry.

9.1.6. NSH Next Protocol

IANA has set up the "NSH Next Protocol" registry, which contains 8-bit values. Next Protocol values 0, 1, 2, 3, 4, and 5 are defined in this document (see Table 8). New values are assigned via "Expert Review" as per [RFC8126].

Next Protocol	Description	Reference
0x00	Unassigned	
0x01	IPv4	RFC 8300
0x02	IPv6	RFC 8300
0x03	Ethernet	RFC 8300
0x04	NSH	RFC 8300
0x05	MPLS	RFC 8300
0x06 - 0xFD	Unassigned	
0xFE	Experiment 1	RFC 8300
0xFF	Experiment 2	RFC 8300

Table 8: NSH Base Header Next Protocol Values

Expert Review requests MUST include a single codepoint per request. Designated Experts evaluating new allocation requests from this registry should consider the potential scarcity of codepoints for an 8-bit value, and check both for duplications and availability of documentation. If the actual assignment of the Next Protocol field allocation reaches half of the range (that is, when there are 128 unassigned values), IANA needs to alert the IESG. At that point, a new more strict allocation policy SHOULD be considered.

10. NSH-Related Codepoints

10.1. NSH Ethertype

An IEEE Ethertype, 0x894F, has been allocated for NSH.

11. References

11.1. 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>.

[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>.

[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>.

[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>.

11.2. Informative References

[NSH-BROADBAND-ALLOCATION]
Napper, J., Kumar, S., Muley, P., Henderickx, W., and M. Boucadair, "NSH Context Header Allocation -- Broadband", Work in Progress, draft-napper-sfc-nsh-broadband-allocation-04, November 2017.

[NSH-DC-ALLOCATION]
Guichard, J., Smith, M., Kumar, S., Majee, S., Agarwal, P., Glavin, K., Laribi, Y., and T. Mizrahi, "Network Service Header (NSH) MD Type 1: Context Header Allocation (Data Center)", Work in Progress, draft-guichard-sfc-nsh-dc-allocation-07, August 2017.

[NSH-ENCRYPT]
Reddy, T., Patil, P., Fluhrer, S., and P. Quinn, "Authenticated and encrypted NSH service chains", Work in Progress, draft-reddy-sfc-nsh-encrypt-00, April 2015.

[PROOF-OF-TRANSIT]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Leddy, J., Youell, S., Mozes, D., and T. Mizrahi, "Proof of Transit", Work in Progress, draft-brockners-proof-of-transit-04, October 2017.

[RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <https://www.rfc-editor.org/info/rfc2784>.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC3692] Narten, T., "Assigning Experimental and Testing Numbers Considered Useful", BCP 82, RFC 3692, DOI 10.17487/RFC3692, January 2004, <https://www.rfc-editor.org/info/rfc3692>.

[RFC6071] Frankel, S. and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap", RFC 6071, DOI 10.17487/RFC6071, February 2011, <https://www.rfc-editor.org/info/rfc6071>.

[RFC6291] Andersson, L., van Helvoort, H., Bonica, R., Romascanu, D., and S. Mansfield, "Guidelines for the Use of the "OAM" Acronym in the IETF", BCP 161, RFC 6291, DOI 10.17487/RFC6291, June 2011, <https://www.rfc-editor.org/info/rfc6291>.

[RFC7325] Villamizar, C., Ed., Kompella, K., Amante, S., Malis, A., and C. Pignataro, "MPLS Forwarding Compliance and Performance Requirements", RFC 7325, DOI 10.17487/RFC7325, August 2014, <https://www.rfc-editor.org/info/rfc7325>.

[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>.

[RFC7676] Pignataro, C., Bonica, R., and S. Krishnan, "IPv6 Support for Generic Routing Encapsulation (GRE)", RFC 7676, DOI 10.17487/RFC7676, October 2015, <https://www.rfc-editor.org/info/rfc7676>.

[RFC8165] Hardie, T., "Design Considerations for Metadata Insertion", RFC 8165, DOI 10.17487/RFC8165, May 2017, <https://www.rfc-editor.org/info/rfc8165>.

[RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <https://www.rfc-editor.org/info/rfc8201>.

[RTG-ENCAP] Nordmark, E., Tian, A., Gross, J., Hudson, J., Kreeger, L., Garg, P., Thaler, P., and T. Herbert, "Encapsulation Considerations", Work in Progress, draft-ietf-rtgwg-dt-encap-02, October 2016.

[SFC-CONTROL-PLANE] Boucadair, M., "Service Function Chaining (SFC) Control Plane Components & Requirements", Work in Progress, draft-ietf-sfc-control-plane-08, October 2016.

[SFC-OAM-FRAMEWORK] Aldrin, S., Pignataro, C., Kumar, N., Akiya, N., Krishnan, R., and A. Ghanwani, "Service Function Chaining (SFC) Operation, Administration and Maintenance (OAM) Framework", Work in Progress, draft-ietf-sfc-oam-framework-03, September 2017.

[VXLAN-GPE] Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol Extension for VXLAN", Work in Progress, draft-ietf-nvo3-vxlan-gpe-05, October 2017.

Acknowledgments

The authors would like to thank Sunil Vallamkonda, Nagaraj Bagepalli, Abhijit Patra, Peter Bosch, Darrel Lewis, Pritesh Kothari, Tal Mizrahi, and Ken Gray for their detailed reviews, comments, and contributions.

A special thank you goes to David Ward and Tom Edsall for their guidance and feedback.

Additionally, the authors would like to thank Larry Kreeger for his invaluable ideas and contributions, which are reflected throughout this document.

Loa Andersson provided a thorough review and valuable comments; we thank him for that.

Reinaldo Penno deserves a particular thank you for his architecture and implementation work that helped guide the protocol concepts and design.

The editors also acknowledge comprehensive reviews and respective useful suggestions by Med Boucadair, Adrian Farrel, Juergen Schoenwaelder, Acee Lindem, and Kathleen Moriarty.

Lastly, David Dolson has provided significant review, feedback, and suggestions throughout the evolution of this document. His contributions are very much appreciated.

Contributors

This WG document originated as draft-quinn-sfc-nsh; the following are its coauthors and contributors along with their respective affiliations at the time of WG adoption. The editors of this document would like to thank and recognize them and their contributions. These coauthors and contributors provided invaluable concepts and content for this document's creation.

- Jim Guichard, Cisco Systems, Inc.
- Surendra Kumar, Cisco Systems, Inc.
- Michael Smith, Cisco Systems, Inc.
- Wim Henderickx, Alcatel-Lucent
- Tom Nadeau, Brocade
- Puneet Agarwal
- Rajeev Manur, Broadcom
- Abhishek Chauhan, Citrix
- Joel Halpern, Ericsson
- Sumandra Majee, F5
- David Melman, Marvell
- Pankaj Garg, Microsoft
- Brad McConnell, Rackspace
- Chris Wright, Red Hat, Inc.
- Kevin Glavin, Riverbed
- Hong (Cathy) Zhang, Huawei US R&D
- Louis Fourie, Huawei US R&D
- Ron Parker, Affirmed Networks
- Myo Zarny, Goldman Sachs
- Andrew Dolganow, Alcatel-Lucent
- Rex Fernando, Cisco Systems, Inc.
- Praveen Muley, Alcatel-Lucent
- Navindra Yadav, Cisco Systems, Inc.

Authors' Addresses

Paul Quinn (editor)
Cisco Systems, Inc.

Email: paulq@cisco.com

Uri Elzur (editor)
Intel

Email: uri.elzur@intel.com

Carlos Pignataro (editor)
Cisco Systems, Inc.

Email: cpignata@cisco.com