Network Service Header
draft-ietf-sfc-nsh-06.txt

Abstract

   This document describes a Network Service Header (NSH) inserted onto
   packets or frames to realize service function paths. NSH also
   provides a mechanism for metadata exchange along the instantiated
   service path. NSH is the SFC encapsulation required to support the
   Service Function Chaining (SFC) Architecture (defined in RFC7665).

1. Requirements Language

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

Status of this Memo

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

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

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

This Internet-Draft will expire on February 24, 2017.

Copyright Notice

Table of Contents

2. 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 wide area network, data center, campus, 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 requires an agile service insertion model that supports dynamic and elastic service delivery; the movement of service functions and application workloads in the network and the ability to easily bind service policy to granular information such as per-subscriber state and steer traffic to the requisite service function(s) are necessary.

NSH defines a new service plane protocol specifically for the creation of dynamic service chains and is composed of the following elements:

1. Service Function Path identification

2. Transport independent service function chain

3. Per-packet network and service metadata or optional variable type-length-value (TLV) metadata.

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

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

[RFC7665] provides an overview of a service chaining architecture that clearly defines the roles of the various elements and the scope of a service function chaining encapsulation. NSH is the SFC encapsulation referenced in RFC7665.

2.1. Definition of Terms

Classification: Defined in [RFC7665].

Classifier: Defined in [RFC7665].

Metadata: Defined in [RFC7665].

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

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

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

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 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].

SFC Proxy: Defined in [RFC7665].

2.2. Problem Space

Network Service Header (NSH) addresses several limitations associated
with service function deployments. [RFC7498] provides a
comprehensive review of those issues.

2.3. NSH-based Service Chaining

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

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

network overlay for network forwarding.

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

3. NSH provides a mechanism to carry shared metadata between
    participating entities and service functions. The semantics of
    the shared metadata is communicated via a control plane, which is
    outside the scope of this document, to participating nodes.
    Examples of metadata include classification information used for
    policy enforcement and network context for forwarding post
    service delivery.

4. Classification and re-classification: 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.

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

6. Transport Agnostic: NSH is transport independent and is often
    carried via a network transport protocol, over existing
    underlays. This transport may form an overlay network and if an
    existing overlay topology provides the required service path
    connectivity, that existing overlay may be used.

3. Network Service Header

   A Network Service Header (NSH) contains service path information and
   optionally metadata that are added to a packet or frame and used to
   create a service plane. An outer transport header is imposed, on NSH
   and the original packet/frame, for network forwarding.

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

3.1. Network Service Header Format

   An NSH is composed of a 4-byte Base Header, a 4-byte Service Path
   Header and Context Headers, as shown in Figure 1 below.

```
 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 Headers                             ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
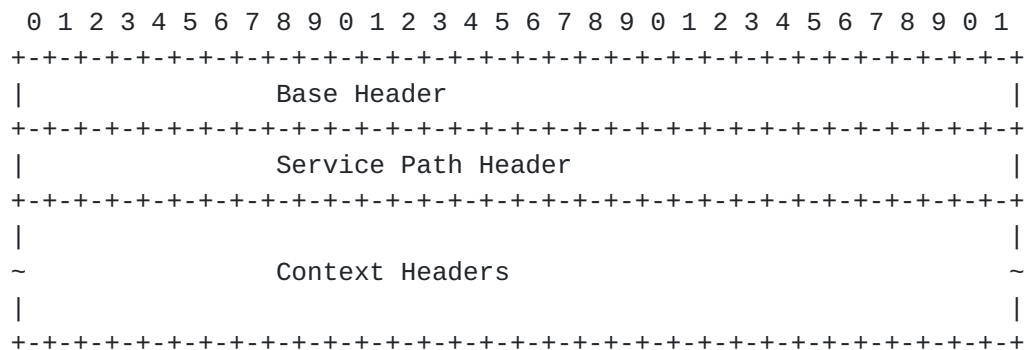
                     Figure 1: Network Service Header

   Base header: provides information about the service header and the
   payload protocol.

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

   Context headers: carry metadata (i.e. context data) along a service
   path.

3.2. NSH Base Header

```
 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|C|R|R|R|R|R|R|   Length  |    MD Type    | Next Protocol |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
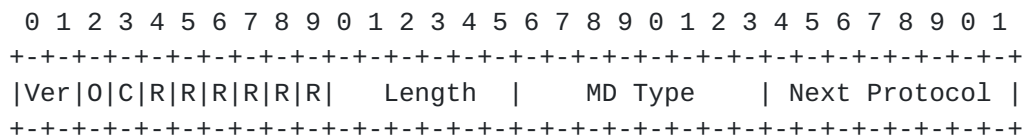
                    Figure 2: NSH Base Header

   Base Header Field Descriptions:

   Version: The version field is used to ensure backward compatibility
   going forward with future NSH updates. It MUST be set to 0x0 by the
   sender, in this first revision of NSH. Given the widespread
   implementation of existing hardware that uses the first nibble after
   an MPLS label stack for ECMP decision processing, this document
   reserves version 01 and 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 Operations, Administration, and
   Maintenance (OAM) packet. The actual packet format and processing of
   SFC OAM messages is outside the scope of this specification (see [I-
   D.ietf-sfc-oam-framework]).

   SF/SFF/SFC Proxy/Classifer implementations, which do not support SFC
   OAM procedures, SHALL discard packets with O-bit set.

   SF/SFF/SFC Proxy/Classifer implementations MAY support a configurable
   parameter to enable forwarding received SFC OAM packets unmodified to
   the next element in the chain. Such behavior may be acceptable for a
   subset of OAM functions, but 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.

   For non OAM packets, the O-bit MUST be cleared and MUST NOT be
   modified along the SFP.

   C bit: Indicates that a critical metadata TLV is present. This bit
   acts as an indication for hardware implementers to decide how to
   handle the presence of a critical TLV without necessarily needing to
   parse all TLVs present. For an MD Type of 0x1 (i.e. no variable
   length metadata is present), the C bit MUST be set to 0x0.

   All other flag fields are reserved for future use. Reserved bits
   MUST be set to zero when sent and MUST be ignored upon receipt.

   Length: total length, in 4-byte words, of NSH including the Base
   Header, the Service Path Header and the context headers or optional
   variable length metadata. The Length MUST be of value 0x6 for MD
   Type equal to 0x1 and MUST be of value 0x2 or greater for MD Type
   equal to 0x2. The NSH header length MUST be an integer number of 4
   bytes. The length field indicates the "end" of NSH and where the

original packet/frame begins.

MD Type: indicates the format of NSH beyond the mandatory Base Header and the Service Path Header. MD Type defines the format of the metadata being carried. Please see IANA Considerations section below.

NSH defines two MD types:

0x1 - which indicates that the format of the header includes fixed length context headers (see Figure 4 below).

0x2 - which does not mandate any headers beyond the Base Header and Service Path Header, but may contain optional variable length context information.

The format of the base header and the service path header is invariant, and not affected by MD Type.

NSH implementations MUST support MD-Type = 0x1, and SHOULD support MD- Type = 0x2. There exists, however, a middle ground, wherein a device will support MD-Type 0x1 (as per the MUST) metadata, yet be deployed in a network with MD-Type 0x2 metadata packets. In that case, the MD-Type 0x1 node, MUST utilize the base header length field to determine the original payload offset if it requires access to the original packet/frame.

Next Protocol: indicates the protocol type of the encapsulated data. NSH does not alter the inner payload, and the semantics on the inner protocol remain unchanged due to NSH service function chaining. Please see IANA Considerations section below.

This draft defines the following Next Protocol values:

0x1 : IPv4
0x2 : IPv6
0x3 : Ethernet
0x4: NSH
0x5: MPLS
0x6-0xFD: Unassigned
0xFE-0xFF: Experimental

3.3. Service Path Header

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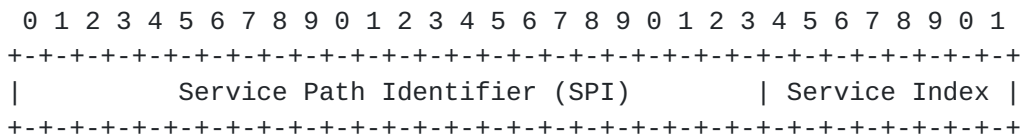
Service Path Identifier (SPI): 24 bits
Service Index (SI): 8 bits

Figure 3: NSH Service Path Header

Service Path Identifier (SPI): identifies a service path.
Participating nodes MUST use this identifier for Service Function
Path selection.

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 SI as appropriate
(i.e. taking into account the length of the service function path).
Service Index MUST be decremented by Service Functions or by SFC
Proxy nodes after performing required services and the new
decremented SI value MUST be used in the egress NSH packet. 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.

SI SHOULD be used in conjunction with SPI for SFP selection and for
determining the next SFF/SF in the path. Service Index (SI) is also
valuable when troubleshooting/ reporting service paths. When an SPI
and SI do not correspond to a valid next Service Function path, it is
an error and the SFF SHOULD generate an error/log message. The value
zero for SI is not valid and indicates a broken SFC or malfunctioning
SF. In addition to indicating the location within a Service Function
Path, SI can be used for service plane loop detection.

3.4. NSH MD-type 1

When the Base Header specifies MD Type = 0x1, four Context Headers,
4-byte each, MUST be added immediately following the Service Path
Header, as per Figure 4. Context Headers that carry no metadata MUST
be set to zero.

```
       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|C|R|R|R|R|R|   Length  | MD-type=0x1 | Next Protocol |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |            Service Path Identifer          | Service Index |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                 Mandatory Context Header                     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                 Mandatory Context Header                     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                 Mandatory Context Header                     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                 Mandatory Context Header                     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
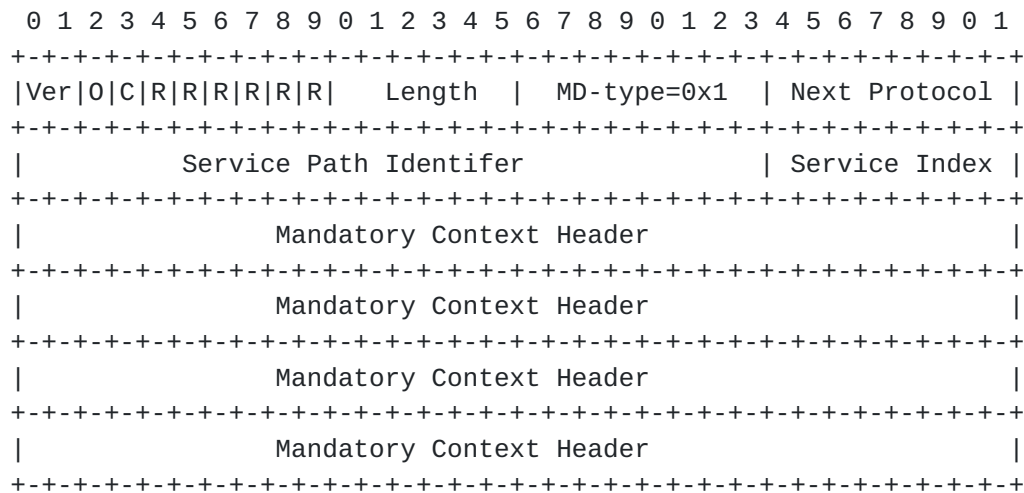
                     Figure 4: NSH MD-type=0x1

   [dcalloc] and [broadalloc] provide specific examples of how metadata
   can be allocated.

3.5. 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. Therefore, Length = 0x2, indicates that only
   the Base Header followed by the Service Path Header are present. 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 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|C|R|R|R|R|R|   Length  |  MD-type=0x2  | Next Protocol |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |            Service Path Identifier          | Service Index |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                                                             |
      ~            Variable Length Context Headers  (opt.)          ~
      |                                                             |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
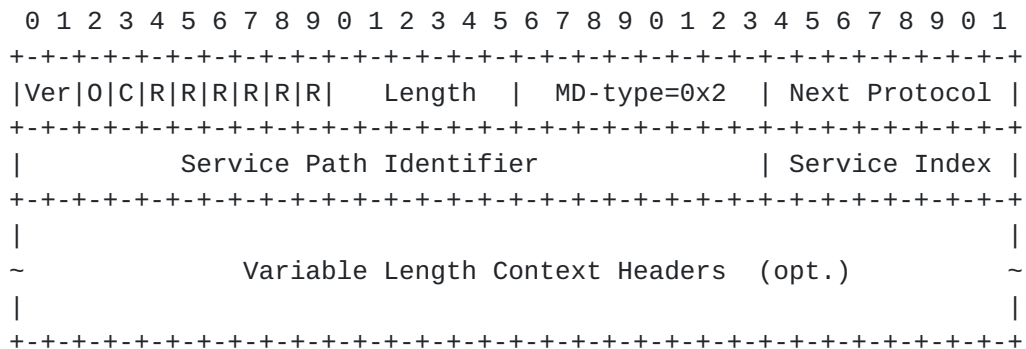
                     Figure 5: NSH MD-type=0x2

3.5.1. Optional Variable Length Metadata

   The format of the optional variable length context headers, is as
   described below.

```
     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       |C|    Type     |R|    Len    |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                      Variable Metadata                       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
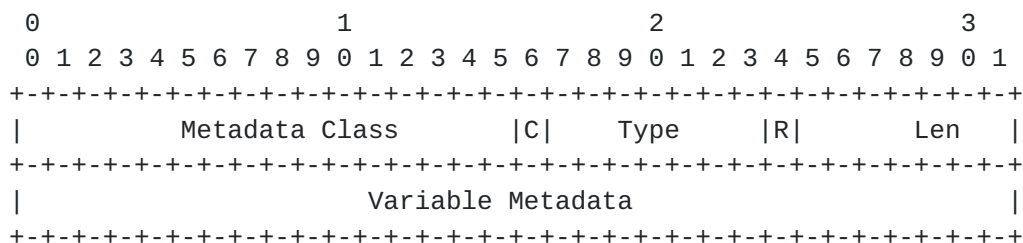
                   Figure 6: Variable Context Headers

   Metadata Class (MD Class): The MD Class defines the scope of the
   'Type' field to provide a hierarchical namespace. The IANA
   Considerations section defines how the MD Class values can be
   allocated to standards bodies, vendors, and others.

   Type: the Type field is split into two ranges - 0 to 127 for non-
   critical options and 128-255 for critical options. While the value
   allocation is the responsibility of the MD Class owner, critical
   options MUST NOT be allocated from the 0 to 127 range and non-
   critical options MUST NOT be allocated from the 128-255 range.

   Figure 7 below illustrates the placement of the Critical bit within
   the Type field.

```
     +-+-+-+-+-+-+-+-+
     |C|    Type     |
     +-+-+-+-+-+-+-+-+
```
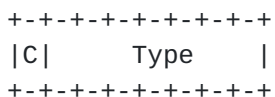
        Figure 7: Critical Bit Placement Within the TLV Type Field

   If an NSH-aware node receives an encapsulated packet containing a TLV
   with the Critical bit set to 0x1 in the Type field and it does not
   understand how to process the Type, it MUST drop the packet. Transit
   devices (i.e. network nodes that do not participate in the service
   plane) MUST NOT drop packets based on the setting of this bit.

   Reserved bits: three reserved bit are present for future use. The

reserved bits MUST be set to 0x0.

Length: Length of the variable metadata, in single byte words. 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 up the length field 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 single byte words) and
MUST ignore the remaining bytes up to the nearest 4-byte word
boundary. A value of 0x0 or higher can be used.

A value of 0x0 denotes a TLV header without a Variable Metadata
field.

[4](#). NSH Actions

   NSH-aware nodes are the only nodes that MAY alter the content of the
   NSH headers. NSH-aware nodes include: service classifiers, SFF, SF
   and SFC proxies. These nodes have several possible header related
   actions:

   1. Insert or remove NSH: These actions can occur at the start and
       end respectively of a service path. Packets are classified, and
       if determined to require servicing, NSH will be imposed. A
       service classifier MUST insert NSH at the start of an SFP. An
       imposed NSH MUST contain valid Base Header and Service Path
       Header. At the end of a service function path, a SFF, MUST be
       the last node operating on the service header and MUST remove it.

       Multiple logical classifiers may exist within a given service
       path. Non-initial classifiers may re-classify data and that re-
       classification MAY result in a new Service Function Path. When
       the logical classifier performs re-classification that results in
       a change of service path, it MUST remove the existing NSH and
       MUST impose a new NSH with the Base Header and Service Path
       Header reflecting the new service path information and set the
       initial SI. Metadata MAY be preserved in the new NSH.

   2. Select service path: The Service Path Header provides service
       chain 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 a Service Path Header: NSH-aware service functions (SF)
       MUST decrement the service index. If an SFF receives a packet
       with an SPI and SI that do not correspond to a valid next Service
       Function Path, that packet MUST be dropped by the SFF.

       Classifier(s) MAY update Context Headers if new/updated context
       is available.

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

4. Service policy selection: Service Function instances derive
   policy (i.e. service actions such as permit or deny) selection
   and enforcement from the service header. Metadata shared in the
   service header can provide a range of service-relevant
   information such as traffic classification. Service functions
   SHOULD use NSH to select local service policy.

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

Component	Insert or remove NSH		Select Service Function Path	Update NSH		Service policy selection
	Insert	Remove		Dec. Service Index	Update Context Header	
Classifier	+	+			+	
Service Function Forwarder(SFF)		+	+			
Service Function (SF)				+	+	+
SFC Proxy	+	+		+		

Figure 8: NSH Action and Role Mapping

5. NSH Encapsulation

   Once NSH is added to a packet, an outer 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 as
       is.

   The service header is independent of the encapsulation used and is
   encapsulated in existing transports. The presence of NSH is
   indicated via protocol type or other indicator in the outer
   encapsulation.

   See Section 9 for NSH encapsulation examples.

6. Fragmentation Considerations

   NSH and the associated transport header are "added" to the
   encapsulated packet/frame. This additional information increases the
   size of the packet. In order to ensure proper forwarding of NSH
   packets, several options for handling fragmentation and re-assembly
   exist:

   As discussed in [draft-ietf-rtgwg-dt-encap-01], within an
   administrative domain, an operator can ensure that the underlay MTU
   is sufficient to carry SFC traffic without requiring fragmentation.

   However, there will be cases where the underlay MTU is not large
   enough to carry the NSH traffic. Since NSH does not provide
   fragmentation support at the service plane, the transport/overlay
   layer MUST provide the requisite fragmentation handling. Section 6
   of [draft-ietf-rtgwg-dt-encap-01] provides guidance for those
   scenarios.

7. **Service Path Forwarding with NSH**

7.1. **SFFs and Overlay Selection**

   As described above, 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 provide a level of indirection between the service
   path/topology and the network transport. Furthermore, there is no
   requirement, or expectation of an SPI being bound to a pre-determined
   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, and 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 can also serve as a mechanism for loop detection within a service
   path since each SF in the path decrements the index; an Service Index
   of 0 indicates that a loop occurred and the packet must be discarded.

   This indirection -- path ID 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.). As mentioned above, an existing overlay
   topology may be used provided it offers the requisite connectivity.

   The mapping of SPI to transport occurs on an SFF (as discussed above,
   the first SFF in the path gets a NSH encapsulated packet from the
   Classifier). The SFF consults the SPI/ID values to determine the
   appropriate overlay transport protocol (several may be used within a
   given network) and next hop for the requisite SF. Figure 9 below
   depicts an example of a single next-hop SPI/SI to network overlay
   network locator mapping.

```
+----------------------------------------------------------+
| SPI | SI | Next hop(s)           | Transport           |
+----------------------------------------------------------+
| 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                |
+----------------------------------------------------------+
```

                   Figure 9: 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 a service function chain control plane responsibility, as
per figures 10 and 11 below.

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

```
 +----------------------------+
| SPI | SI | Next hop(s)   |
 +----------------------------+
| 10  | 3  |     SF2       |
| 245 | 12 |     SF34      |
| 40  | 9  |     SF9       |
 +----------------------------+
```

                   Figure 10: NSH to SF Mapping Example

```
 +----------------------------------------+
| SF  | Next hop(s)      | Transport |
 +----------------------------------------|
|  SF2 | 192.0.2.2        | VXLAN-gpe |
|  SF34| 198.51.100.34    | UDP       |
|  SF9 | 2001:db8::1      | GRE       |
 +--------------------------+-------------
 =
```

Figure 11: 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
Figure 12. The metric depicted in Figure 12 is an example to help
illustrated 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 some service function-centric state
(including load, sessions 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)

Figure 12: NSH Weighted Service Path

7.2. Mapping NSH to Network Transport

As described above, the mapping of 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 existing. 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 192.0.2.1
      SFFa mapping: SPI=10 --> VXLAN-gpe, dst-ip: 192.0.2.1

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.

7.3. 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
the NSH information (packet capture, IPFIX, etc.). The information
can be used for service scheduling and placement decisions,
troubleshooting and compliance verification.

7.4. Service Graphs

While a given realized service function path is a specific sequence
of service functions, the service as seen by a user can actually be a
collection of service function paths, 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 also of course
modify the metadata associated with the packet.
RFC7665, section 2.1 describes Service Graphs in detail.

8. **Policy Enforcement with NSH**

8.1. **NSH Metadata and Policy Enforcement**

   As described in Section 3, 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 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 perform very detailed and valuable classification. In some
      cases they may terminate, and be able to inspect encrypted
      traffic.

   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 5-tuple, whereas, a service function may be
   able to inspect application information. Regardless of granularity,
   the classification information can be represented in NSH.

   Once the data is added to 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. The following two examples
   highlight the relationship between metadata and policy:

```
   +-------+           +-------+          +-------+
   |  SFF  )------->(  SFF  |------->|  SFF  |
   +---^---+           +---|---+          +---|---+
    ,-|-.               ,-|-.              ,-|-.
   /     \             /     \            /     \
  ( Class )             SF1  )          (  SF2  )
   \ ify /             \     /            \     /
    `---'               `---'              `---'
   5-tuple:            Permit             Inspect
   Tenant A            Tenant A           AppY
   AppY
```
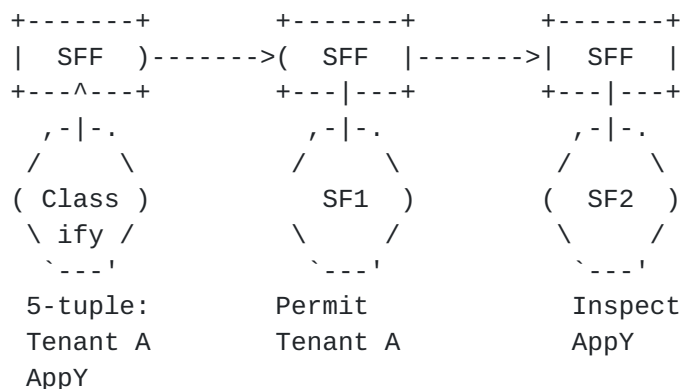
                     Figure 13: Metadata and Policy


```
    +-----+            +-----+           +-----+
    | SFF |---------> | SFF |----------> | SFF |
    +--+--+            +--+--+           +--+--+
       ^                  |                 |
     ,-+-.              ,-+-.             ,-+-.
    /     \            /     \           /     \
   ( Class )          (  SF1  )         (  SF2  )
    \ ify /            \     /           \     /
     `-+-'              `---'             `---'
       |               Permit          Deny AppZ
   +---+---+           employees
   |       |
   +-------+
   external
   system:
   Employee
   AppZ
```
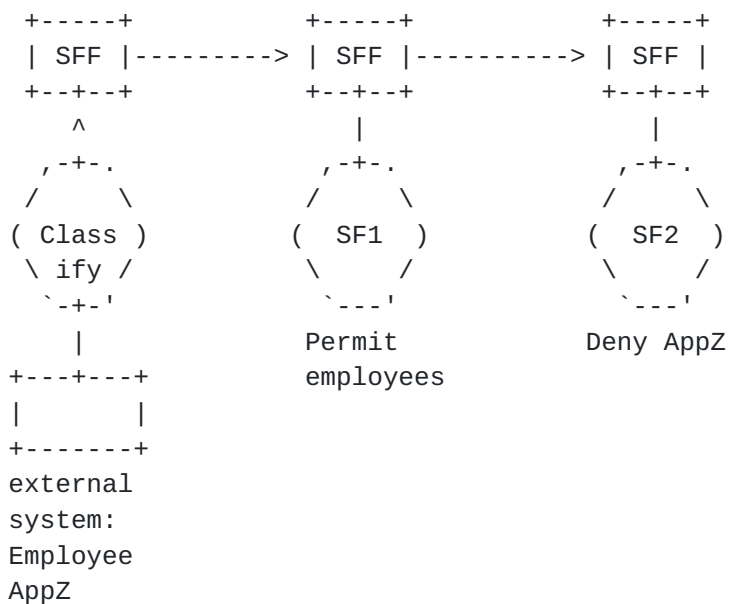
                  Figure 14: External Metadata and Policy

   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, rather they rely on a
   antecedent classification for local policy enforcement.

   Depending on the information carried in the metadata, data privacy
   considerations may need 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) . NSH

itself does not provide privacy functions, rather it relies on the
transport/overlay layer. An operator can select the appropriate
transport to ensure the confidentially (and other security)
considerations are met.

8.2. Updating/Augmenting Metadata

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

1. Metadata Augmentation: Information may be added to NSH's existing
    metadata, as depicted in Figure 17. For example, if the initial
    classification returns the tenant information, a secondary
    classification (perhaps co-resident with DPI or SLB) may augment
    the tenant classification with application information, and
    impose that new information in the 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 16 illustrates an example of updating metadata.

```
    +-----+               +-----+                +-----+
    | SFF |---------> | SFF |----------> | SFF |
    +--+--+               +--+--+                +--+--+
      ^                      |                       |
    ,---.                  ,---.                   ,---.
   /     \                /     \                 /     \
  ( Class )              (  SF1  )               (  SF2  )
   \     /                \     /                 \     /
    `-+-'                  `---'                   `---'
     |                   Inspect              Deny
  +---+---+               employees              employee+
  |       |              Class=AppZ             appZ
  +-------+
  external
  system:
  Employee
```
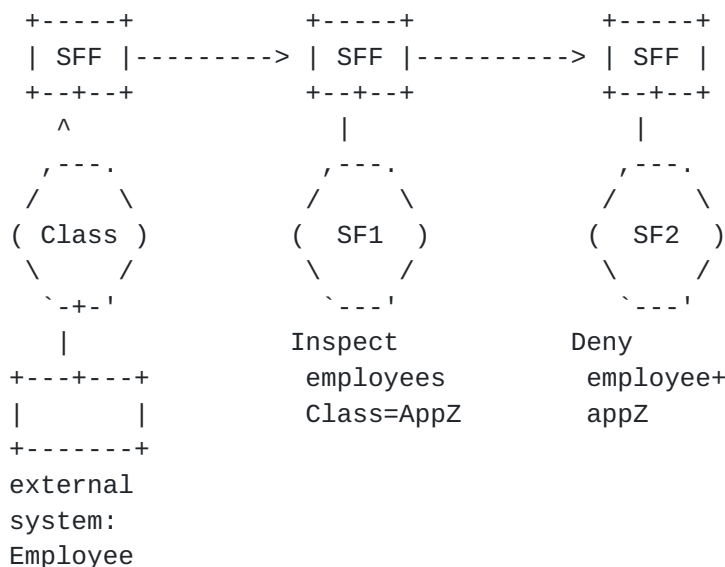
                    Figure 15: Metadata Augmentation

```
   +-----+              +-----+              +-----+
   | SFF |--------->  | SFF |---------->  | SFF |
   +--+--+              +--+--+              +--+--+
      ^                    |                    |
    ,---.                ,---.                ,---.
   /     \              /     \              /     \
  ( Class )           (  SF1  )           (  SF2  )
   \     /              \     /              \     /
    `---'                `---'                `---'
  5-tuple:              Inspect              Deny
  Tenant A              Tenant A             attack
                        --> attack
```
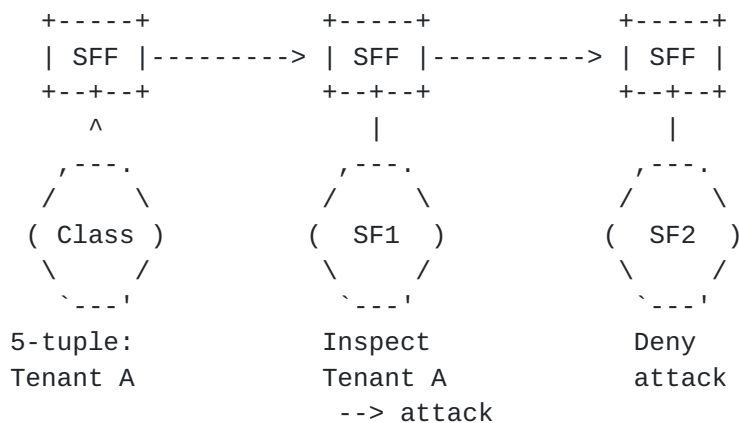
                       Figure 16: Metadata Update

8.3. Service Path Identifier and Metadata

   Metadata information may influence the service path selection since
   the Service Path Identifier and Service Index values can represent
   the result of classification. A given SPI and SI can be defined
   based on classification results (including metadata classification).
   The imposition of the SPI/SI (new or an change of existing) reflect,
   as previously described, the new SFP.

   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 7.4. Figure 17 illustrates an example of this
   behavior.

```
    +-----+            +-----+              +-----+
    | SFF |---------> | SFF |------+---> | SFF |
    +--+--+            +--+--+      |      +--+--+
       |                  |         |         |
     ,---.              ,---.       |       ,---.
    /     \            /     \      |      /     \
   (  SCL  )          (  SF1  )     |     (  SF2  )
    \     /            \     /      |      \     /
     `---'              `---'    +-----+    `---'
   5-tuple:            Inspect   | SFF |    Original
   Tenant A            Tenant A  +--+--+    next SF
                        --> DoS     |
                                    V
                                  ,-+-.
                                 /     \
                                (  SF10 )
                                 \     /
                                  `---'
                                   DoS
                                "Scrubber"
```
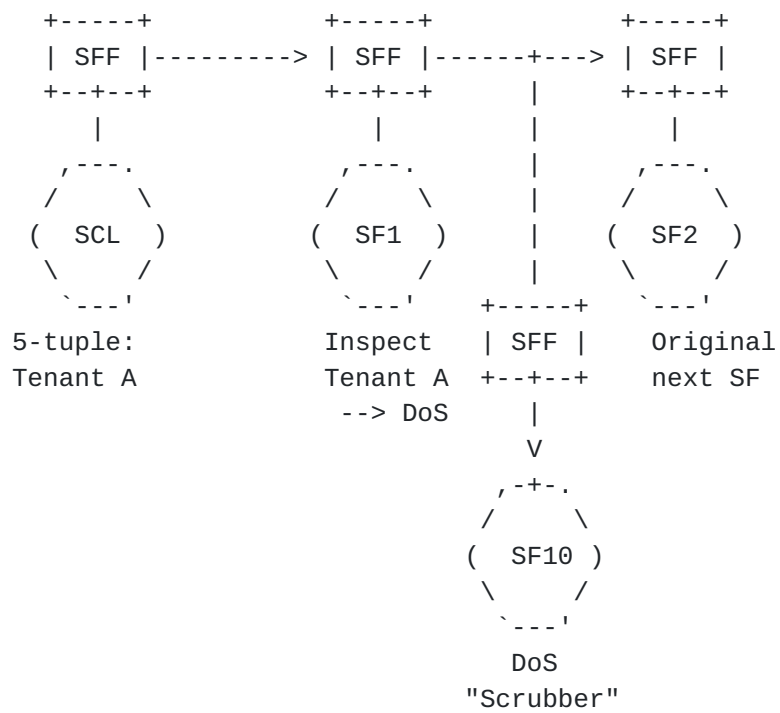
                   Figure 17: Path ID and Metadata

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

9. Security Considerations

   As with many other protocols, NSH data can be spoofed or otherwise
   modified. As noted in the descriptive text in [sfc-security-
   requirements], in many deployments, NSH will be used in a controlled
   environment, with trusted devices (e.g. a data center) thus
   mitigating the risk of unauthorized header manipulation. As noted
   there, far fewer protection mechanisms are needed in these
   environments, which are the primary design target of NSH.

   NSH is always encapsulated in a transport protocol and therefore,
   when required, existing security protocols that provide authenticity
   (e.g. [[RFC6071]]) can be used between SFF or even to SF. Similarly
   if confidentiality is required, existing encryption protocols can be
   used in conjunction with encapsulated NSH.

   Further, existing best practices, such as [BCP38] should be deployed
   at the network layer to ensure that traffic entering the service path
   is indeed "valid". [draft-ietf-rtgwg-dt-encap-01.txt] provides
   additional transport encapsulation considerations.

   NSH metadata authenticity and confidentially must be considered as
   well. In order to protect the metadata, an operator can leverage the
   aforementioned mechanisms provided the transport layer, authenticity
   and/or confidentiality. An operator MUST carefully select the
   transport/underlay services to ensure end to end security services,
   when those are sought after. For example, if RFC6071 is used, the
   operator MUST ensure it can be supported by the transport/underlay of
   all relevant network segments as well as SFF and SFs. Further, as
   describe in [section 8.1], operators can and should use indirect
   identification for personally identifying information, thus
   significantly mitigating the risk of privacy violation.

   Further, the extensibility of MD-2 to add information to packets, and
   where needed to mark that data as critical, allows for attaching
   signatures or even encryption keying information to the NSH header in
   the future. Based on the learnings from the work on [nsh-sec], it
   appears likely that this can provide any needed NSH-specific security
   mechanisms in the future.

   Lastly, SF security, although out of scope of this document, should
   be considered, particularly if an SF needs to access, authenticate or
   update NSH metadata.

   Further security considerations are discussed in [nsh-sec].

10. Contributors

   This WG document originated as draft-quinn-sfc-nsh and had the
   following co-authors and contributors. The editors of this document
   would like to thank and recognize them and their contributions.
   These co-authors and contributors provided invaluable concepts and
   content for this document's creation.

   Surendra Kumar
   Cisco Systems
   smkumar@cisco.com

   Michael Smith
   Cisco Systems
   michsmit@cisco.com

   Jim Guichard
   Cisco Systems
   jguichar@cisco.com

   Rex Fernando
   Cisco Systems
   Email: rex@cisco.com

   Navindra Yadav
   Cisco Systems
   Email: nyadav@cisco.com

   Wim Henderickx
   Alcatel-Lucent
   wim.henderickx@alcatel-lucent.com

   Andrew Dolganow
   Alcaltel-Lucent
   Email: andrew.dolganow@alcatel-lucent.com

   Praveen Muley
   Alcaltel-Lucent
   Email: praveen.muley@alcatel-lucent.com

   Tom Nadeau
   Brocade
   tnadeau@lucidvision.com

   Puneet Agarwal
   puneet@acm.org

   Rajeev Manur

      Broadcom
      rmanur@broadcom.com

      Abhishek Chauhan
      Citrix
      Abhishek.Chauhan@citrix.com

      Joel Halpern
      Ericsson
      joel.halpern@ericsson.com

      Sumandra Majee
      F5
      S.Majee@f5.com

      David Melman
      Marvell
      davidme@marvell.com

      Pankaj Garg
      Microsoft
      pankajg@microsoft.com

      Brad McConnell
      Rackspace
      bmcconne@rackspace.com

      Chris Wright
      Red Hat Inc.
      chrisw@redhat.com

      Kevin Glavin
      Riverbed
      kevin.glavin@riverbed.com

      Hong (Cathy) Zhang
      Huawei US R&D
      cathy.h.zhang@huawei.com

      Louis Fourie
      Huawei US R&D
      louis.fourie@huawei.com

      Ron Parker
      Affirmed Networks
      ron_parker@affirmednetworks.com

      Myo Zarny

      Goldman Sachs
      myo.zarny@gs.com

11. 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 review, 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 Carlos Pignataro and
   Larry Kreeger for their invaluable ideas and contributions which are
   reflected throughout this document.

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

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

12. IANA Considerations

12.1. NSH EtherType

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

12.2. Network Service Header (NSH) Parameters

IANA is requested to create a new "Network Service Header (NSH)
Parameters" registry. The following sub-sections request new
registries within the "Network Service Header (NSH) Parameters "
registry.

12.2.1. NSH Base Header Reserved Bits

There are ten bits at the beginning of the NSH Base Header. New bits
are assigned via Standards Action [RFC5226].

```
Bits 0-1 - Version
Bit 2 - OAM (O bit)
Bit 3 - Critical TLV (C bit)
Bits 4-9 - Reserved
```

12.2.2. NSH Version

IANA is requested to setup a registry of "NSH Version". New values
are assigned via Standards Action [RFC5226].

```
Version 00: This protocol version.  This document.
Version 01: Reserved.  This document.
Version 10: Unassigned.
Version 11: Unassigned.
```

12.2.3. MD Type Registry

IANA is requested to set up a registry of "MD Types". These are
8-bit values. MD Type values 0, 1, 2, 254, and 255 are specified in
this document. Registry entries are assigned by using the "IETF
Review" policy defined in RFC 5226 [RFC5226].

```
+---------+-------------+---------------+
| MD Type | Description | Reference     |
+---------+-------------+---------------+
| 0       | Reserved    | This document |
|         |             |               |
| 1       | NSH         | This document |
|         |             |               |
| 2       | NSH         | This document |
|         |             |               |
| 3..253  | Unassigned  |               |
|         |             |               |
| 254     | Experiment 1 | This document |
|         |             |               |
| 255     | Experiment 2 | This document |
+---------+-------------+---------------+
```

Table 1

12.2.4. MD Class Registry

IANA is requested to set up a registry of "MD Class". These are 16-
bit values. MD Classes defined by this document are assigned as
follows:
0x0000 to 0x01ff: IETF Review
0x0200 to 0x7fff: Expert Review
0xfff6 to 0xfffe: Experimental
0xffff: Reserved

12.2.5. NSH Base Header Next Protocol

IANA is requested to set up a registry of "Next Protocol". These are
8-bit values. Next Protocol values 0, 1, 2, 3, 4 and 5 are defined
in this draft. New values are assigned via Standards Action
[RFC5226].

```
+---------------+-------------+--------------+
| Next Protocol | Description | Reference    |
+---------------+-------------+--------------+
| 0             | Reserved    | This document |
|               |             |              |
| 1             | IPv4        | This document |
|               |             |              |
| 2             | IPv6        | This document |
|               |             |              |
| 3             | Ethernet    | This document |
|               |             |              |
| 4             | NSH         | This document |
|               |             |              |
| 5             | MPLS        | This document |
|               |             |              |
| 6..253        | Unassigned  |              |
|               |             |              |
| 254           | Experiment 1 | This document |
|               |             |              |
| 255           | Experiment 2 | This document |
+---------------+-------------+--------------+
```

Table 2

13. References

13.1. 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,
           <http://www.rfc-editor.org/info/rfc2119>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an
           IANA Considerations Section in RFCs", BCP 26, RFC 5226,
           DOI 10.17487/RFC5226, May 2008,
           <http://www.rfc-editor.org/info/rfc5226>.

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

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

13.2. Informative References

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191,
           DOI 10.17487/RFC1191, November 1990,
           <http://www.rfc-editor.org/info/rfc1191>.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery
           for IP version 6", RFC 1981, DOI 10.17487/RFC1981,
           August 1996, <http://www.rfc-editor.org/info/rfc1981>.

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

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

[RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The
           Locator/ID Separation Protocol (LISP)", RFC 6830,
           DOI 10.17487/RFC6830, January 2013,
           <http://www.rfc-editor.org/info/rfc6830>.

   [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, <http://www.rfc-editor.org/info/rfc7325>.

   [VXLAN-gpe]
              Quinn, P., Manur, R., Agarwal, P., Kreeger, L., Lewis, D.,
              Maino, F., Smith, M., Yong, L., Xu, X., Elzur, U., Garg,
              P., and D. Melman, "Generic Protocol Extension for VXLAN",
              <https://datatracker.ietf.org/doc/
              draft-ietf-nvo3-vxlan-gpe/>.

   [broadalloc]
              Napper, J., Kumar, S., Muley, P., and W. Hendericks, "NSH
              Context Header Allocation -- Mobility", 2016, <https://
              datatracker.ietf.org/doc/
              draft-napper-sfc-nsh-broadband-allocation/>.

   [dcalloc] Guichard, J., Smith, M., and et al., "Network Service
              Header (NSH) Context Header Allocation (Data Center)",
              2016, <https://datatracker.ietf.org/doc/
              draft-guichard-sfc-nsh-dc-allocation/>.

   [nsh-sec] Reddy, T., Migault, D., Pignataro, C., Quinn, P., and C.
              Inacio, "NSH Security and Privacy requirements", 2016,.

Authors' Addresses

    Paul Quinn (editor)
    Cisco Systems, Inc.

    Email: paulq@cisco.com

    Uri Elzur (editor)
    Intel

    Email: uri.elzur@intel.com