Network Service Header (NSH)
draft-ietf-sfc-nsh-15

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 paths. NSH is the SFC encapsulation required to support the
   Service Function Chaining (SFC) architecture (defined in RFC7665).

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 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.

   Network Service Header (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. Indication of location within a Service Function Path.

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

   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].

1.1. 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].

1.2. 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: 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 an 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].

1.3. Problem Space

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

1.4. NSH-based Service Chaining

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. NSH fields can be used by administrators (via, for
    example, a traffic analyzer) 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.
    [I-D.ietf-sfc-control-plane] provides an example of such in
    Section 3.3. 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. NSH offers a common and standards-based header for service
    chaining to all network and service nodes.

5. Transport Agnostic: NSH is transport-independent. An appropriate
    (for a given deployment) network transport protocol can be used
    to transport NSH-encapsulated traffic. 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.

2. 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 NSH. 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

NSH is composed of a 4-byte Base Header, a 4-byte Service Path Header
and optional Context Headers, as shown in Figure 1 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Base Header                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Service Path Header                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                   Context Header(s)                           ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
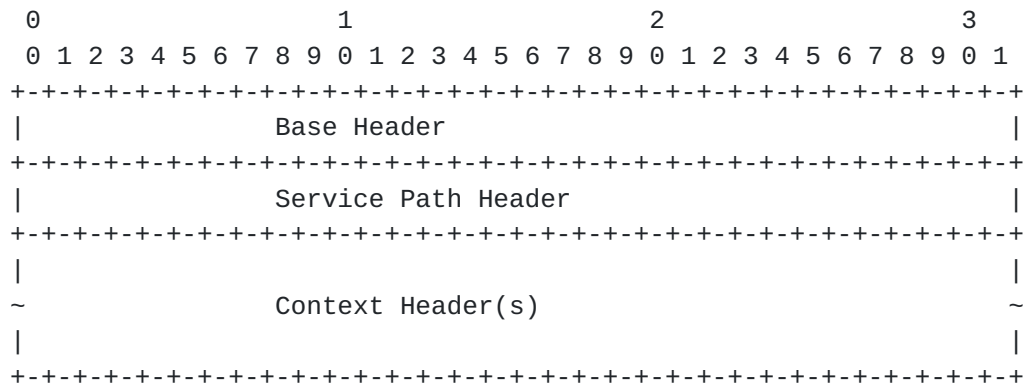
Figure 1: Network Service Header

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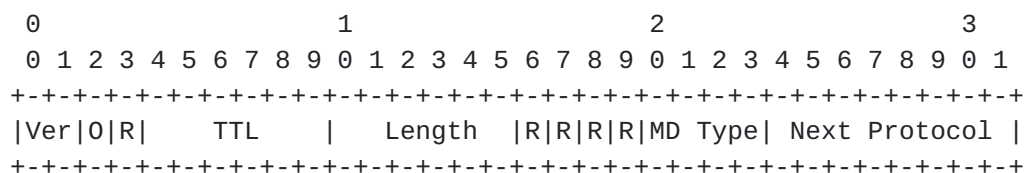
Figure 2: NSH Base Header

Base Header Field Descriptions:

Version: The version field is used to ensure backward compatibility
going forward with future NSH specification 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 format and processing of SFC
OAM packets is outside the scope of this specification (see for
example [I-D.ietf-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 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. 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 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.

All other flag fields are reserved for future use. Reserved bits
MUST be set to zero upon origination and MUST be preserved unmodified
by other NSH supporting elements. Elements which do not understand
the meaning of any of these bits MUST NOT modify their actions based
on those unknown bits.

Length: The total length, in 4-byte words, of NSH including the Base
Header, the Service Path Header, the Fixed Length Context Header or
Variable Length Context Header(s). 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 length of the NSH header MUST be an integer

multiple of 4 bytes, thus variable length metadata is always padded
out to a multiple of 4 bytes.

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 the IANA Considerations
Section 11.2.3.

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 - which indicates that the format of the header includes a fixed
length Context Header (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
Header(s). 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 0x15.

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 MD Type = 0x2
(where the length is of value 0x2). NSH implementations SHOULD
support MD Type 0x2 with length > 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. 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.

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 the IANA Considerations section below, Section 11.2.5.

This document defines the following Next Protocol values:

0x0: Unassigned
0x1: IPv4
0x2: IPv6
0x3: Ethernet
0x4: NSH
0x5: MPLS
0xFE: Experiment 1
0xFF: Experiment 2

Packets with Next Protocol values not supported by an implementation
SHOULD be silently dropped. 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 3 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 3: NSH Service Path Header

The meaning of these fields is as follows:

Service Path Identifier (SPI): identifies a service path.
Participating nodes MUST use this identifier for Service Function
Path 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 SI as appropriate
(i.e., taking into account the length of the service function path).
Service Index MUST be decremented by a value of 1 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.

The SI is used in conjunction with Service Path Identifier for Service Function Path Selection and for determining the next SFF/SF in the path. The SI is also valuable when troubleshooting/ reporting service paths. In addition to indicating the location within a Service Function Path, SI can be used for service plane loop detection.

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 4. 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|R|    TTL    |   Length  |R|R|R|R|MD Type| Next Protocol |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |          Service Path Identifier              | Service Index |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                                                               |
  |                Fixed Length Context Header                    |
  |                                                               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
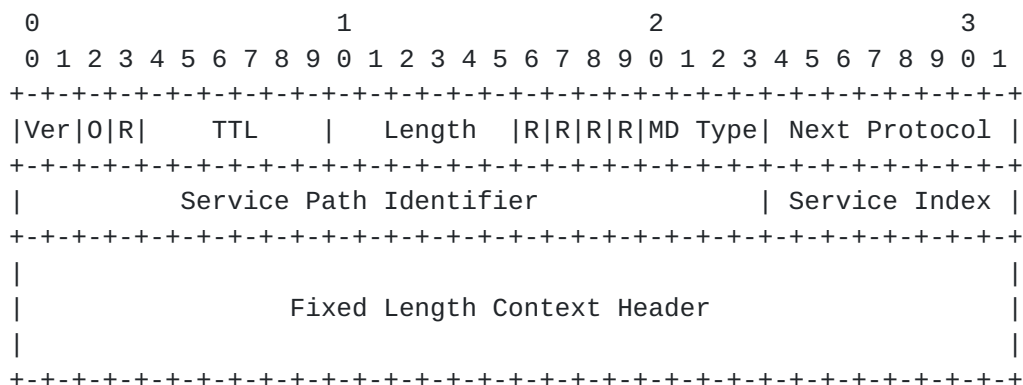
Figure 4: NSH MD Type=0x1

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 does not describe the structure or meaning of the included metadata.

An SFC-aware SF MUST receive the data semantics first in order to process the data placed in the mandatory context field. The data semantics include both the allocation schema and the meaning of the included data. How an SFC-aware SF gets the data 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).

[I-D.guichard-sfc-nsh-dc-allocation] and
[I-D.napper-sfc-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 5). 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
 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|R|    TTL    |   Length  |R|R|R|R|MD Type| Next Protocol |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Service Path Identifier              | Service Index |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~             Variable Length Context Headers  (opt.)          ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
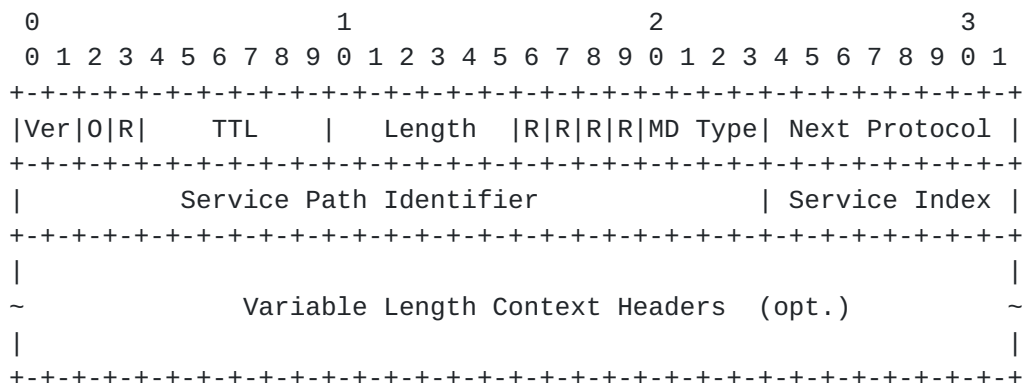
Figure 5: NSH MD Type=0x2

2.5.1. Optional Variable Length Metadata

The format of the optional variable length Context Headers, is as
depicted in Figure 6.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Metadata Class       |     Type      |R|     Len     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Variable Metadata                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
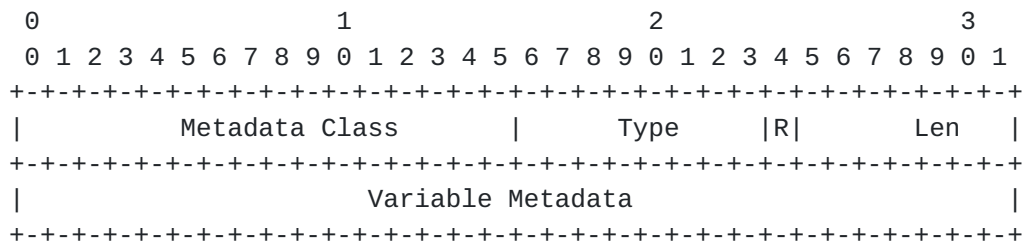
Figure 6: Variable Context Headers

Metadata Class (MD Class): defines the scope of the 'Type' field to
provide a hierarchical namespace. The IANA Considerations

Section 11.2.4 defines how the MD Class values can be allocated to
standards bodies, vendors, and others.

Type: indicates the explicit type of metadata being carried and is
the responsibility of the MD Class owner.

Reserved bit: one reserved bit is present for future use. The
reserved bits MUST be set to 0x0.

Length: indicates the 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. The Length may be 0 or greater.

A value of 0x0 denotes a Context Header without a Variable 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 context header together with their scoping (see
Section 3.3.3 of [I-D.ietf-sfc-control-plane]).

Upon receipt of a packet that belong 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 at least once per the SPI
for which a 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, the SFC-aware SF MUST process these Context Headers in the
order their 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 first instance and ignore
subsequent instances.

3. NSH Actions

   NSH-aware nodes are the only nodes that may alter the content of NSH
   headers. NSH-aware nodes include: service classifiers, SFF, SF and
   SFC proxies. These nodes have several possible NSH-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, an SFF, MUST be
       the last node operating on the service header and MUST remove NSH
       before forwarding or delivering the un-encapsulated packet

       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 a different 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
       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 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 Service Function Path, 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 a NSH unaware
       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 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-encapsulates it
       with the correct NSH, and 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 NSH. Metadata shared in NSH can provide a range of service-
    relevant information such as traffic classification.

Figure 7 maps each of the four actions above to the components in the
SFC architecture that can perform it.

```
+----------------+--------------+-------+----------------+---------+
|                | Insert       |Forward|    Update      |Service  |
|                | or remove NSH |NSH    |     NSH        |policy   |
|                |              |Packets|                |selection|
| Component      +-------+-------+       +----------------+         |
|                |       |       |       | Dec.  |Update  |         |
|                |Insert |Remove |       |Service |Context|         |
|                |       |       |       | Index  |Header |         |
+----------------+-------+-------+-------+--------+-------+---------+
|                |   +   |   +   |       |        |   +   |         |
|Classifier      |       |       |       |        |       |         |
+--------------- +-------+-------+-------+--------+-------+---------+
|Service Function|       |   +   | +     |        |       |         |
|Forwarder(SFF)  |       |       |       |        |       |         |
+--------------- +-------+-------+-------+--------+-------+---------+
|Service         |       |       |       |   +    |   +   |   +     |
|Function  (SF)  |       |       |       |        |       |         |
+--------------- +-------+-------+-------+--------+-------+---------+
|SFC Proxy       |   +   |   +   |       |   +    |   +   |         |
+----------------+-------+-------+-------+--------+-------+---------+
```

                Figure 7: NSH Action and Role Mapping

4. NSH Transport 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.

5. Fragmentation Considerations

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

   As discussed in [I-D.ietf-rtgwg-dt-encap], 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 [I-D.ietf-rtgwg-dt-encap] provides guidance for those scenarios.

6. Service Path Forwarding with NSH

6.1. 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 provides 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 serves as a mechanism for detecting invalid service function path.
   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 over or underlay and in some case

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 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. Table 1 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             |
+------+------+--------------------+------------------+
```

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 a service function chain control plane responsibility, as
per Table 2 and Table 3 below.

Please note: VXLAN-gpe and GRE in the above table refer to
[I-D.ietf-nvo3-vxlan-gpe] and [RFC2784], 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   |
+------+------------------+-------------+
| 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
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)

Table 4: NSH Weighted Service Path

6.2. 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 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, 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 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
   reclassification). These internal reclassifiers 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.

7. Policy Enforcement with NSH

7.1. 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. Figure 8 and Figure 9
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 8: Metadata and Policy

```
        +-----+            +-----+            +-----+
        | SFF |---------> | SFF |----------> | SFF |
        +--+--+            +--+--+            +--+--+
           ^                  |                  |
         ,-+-.              ,-+-.              ,-+-.
        /     \            /     \            /     \
       ( Class )          (  SF1  )          (  SF2  )
        \ ify /            \     /            \     /
         `-+-'              `---'              `---'
           |              Permit            Deny AppZ
       +---+---+          employees
       |       |
       +-------+
       external
       system:
       Employee
       AppZ
```
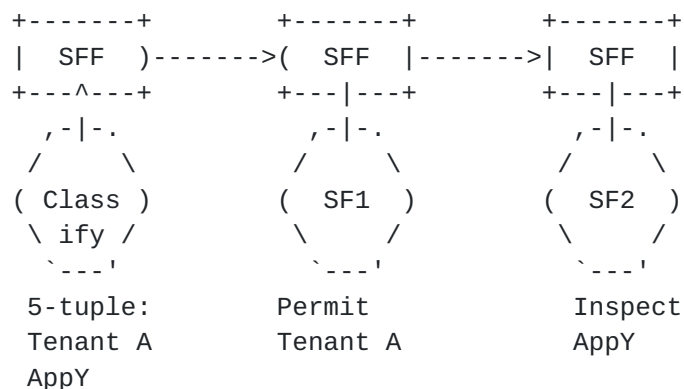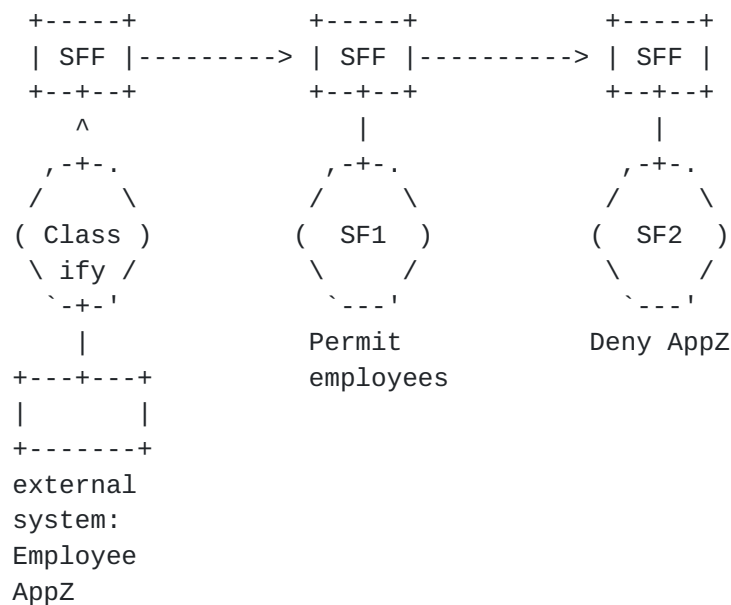
               Figure 9: External Metadata and Policy

   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. 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), metadata may be augmented or updated:

   1. Metadata Augmentation: Information may be added to NSH's existing
       metadata, as depicted in Figure 10. 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 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 11 illustrates an example of updating metadata.

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

                Figure 10: Metadata Augmentation

```
        +-----+             +-----+             +-----+
        | SFF |---------> | SFF |----------> | SFF |
        +--+--+             +--+--+             +--+--+
           ^                   |                   |
         ,---.               ,---.               ,---.
        /     \             /     \             /     \
       ( Class )           (  SF1  )           (  SF2  )
        \     /             \     /             \     /
         `---'               `---'               `---'
        5-tuple:            Inspect             Deny
        Tenant A            Tenant A            attack
                            --> attack
```
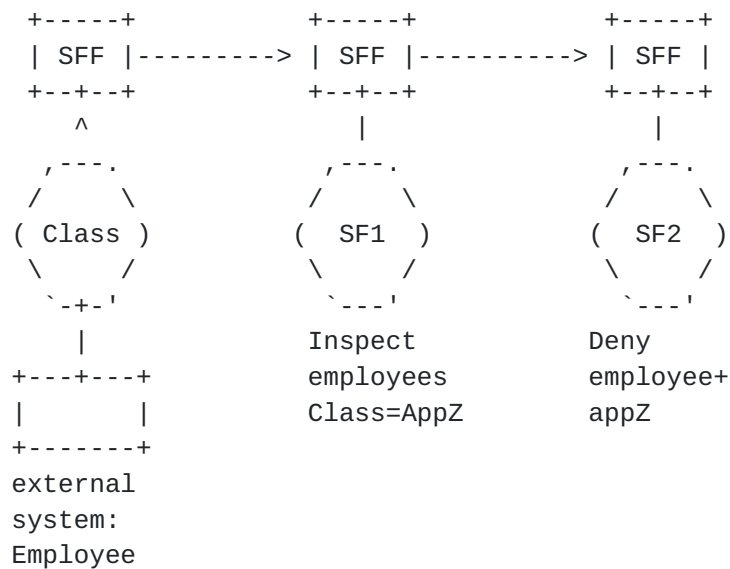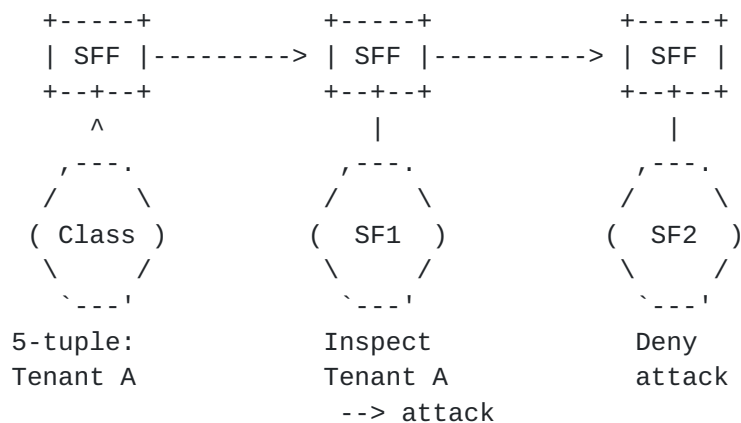
                 Figure 11: Metadata Update

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 7.4. Figure 12 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"
```
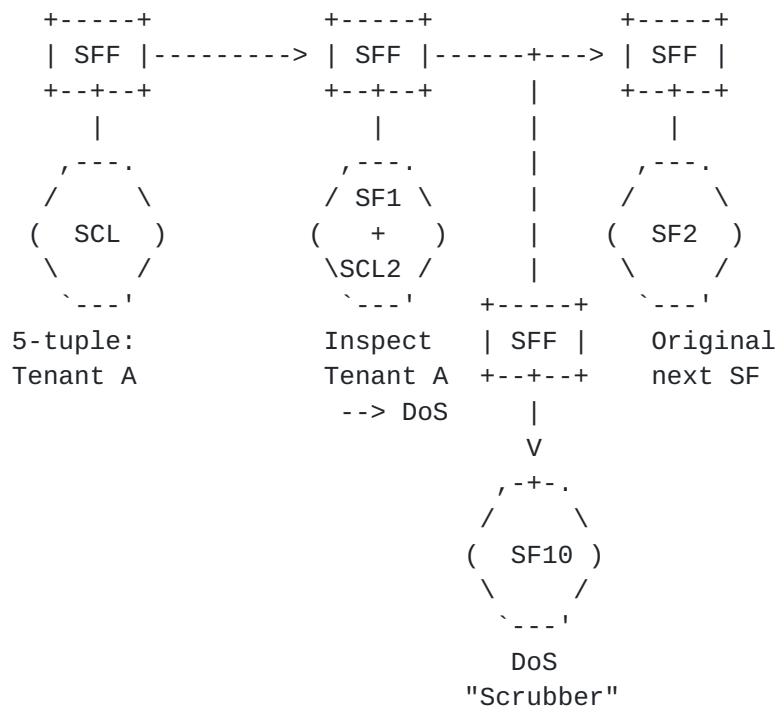
                     Figure 12: Path ID and Metadata

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

8. Security Considerations

   As with many other protocols, NSH data can be spoofed or otherwise
   modified. 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.

NSH is always encapsulated in a transport protocol and therefore, when required, existing security protocols that provide authenticity (e.g., [RFC6071]) can be used. 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". [I-D.ietf-rtgwg-dt-encap] 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 described in Section 8.1, operators can and should use indirect identification for personally identifying information, thus significantly mitigating the risk of privacy violation. Means to prevent leaking privacy-related information outside an administrative domain are natively supported by NSH given that the last SFF of a path will systematically remove the NSH header before forwarding a packet upstream.

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.

9. 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

Huawei
james.n.guichard@huawei.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

[10]. Acknowledgments

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 a comprehensive review and respective suggestions by Med Boucadair.

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

11. IANA Considerations

11.1. NSH EtherType

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

11.2. 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.

11.2.1. NSH Base Header Reserved Bits

There are five reserved bits in the NSH Base Header. New bits are assigned via Standards Action [RFC8126].
Bit 3 - Reserved
Bits 16-19 - Reserved

11.2.2. NSH Version

IANA is requested to setup a registry of "NSH Version". New values are assigned via Standards Action [RFC8126].

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

11.2.3. MD Type Registry

   IANA is requested to set up a registry of "MD Types". These are
   4-bit values. MD Type values 0, 1, 2, and 15 are specified in this
   document, see Table 5. Registry entries are assigned by using the
   "IETF Review" policy defined in RFC 8126 [RFC8126].

```
           +---------+----------------+---------------+
           | MD Type | Description    | Reference     |
           +---------+----------------+---------------+
           | 0       | Reserved       | This document |
           |         |                |               |
           | 1       | NSH MD Type 1  | This document |
           |         |                |               |
           | 2       | NSH MD Type 2  | This document |
           |         |                |               |
           | 3..14   | Unassigned     |               |
           |         |                |               |
           | 15      | Experimentation | This document |
           +---------+----------------+---------------+
```

                       Table 5: MD Type Values

11.2.4. MD Class Registry

   IANA is requested to set up a registry of "MD Class". These are 16-
   bit values. New allocations are to be made according to the
   following policies:

   0x0000 to 0x01ff: IETF Review
   0x0200 to 0xfff5: Expert Review
   0xfff6 to 0xfffe: Experimental
   0xffff: Reserved

   IANA is requested to assign the values as per Table 6::

```
       +-----------+----------------------------+------------+
       | MD Class  | Meaning                    | Reference  |
       +-----------+----------------------------+------------+
       | 0x0000    | IETF Base NSH MD Class     | This.I-D   |
       +-----------+----------------------------+------------+
```

                       Table 6: MD Class Value

   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.

11.2.5. 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 document (see Table 7. New values are assigned via "Expert
Reviews" as per [RFC8126].

```
+---------------+--------------+---------------+
| Next Protocol | Description  | Reference     |
+---------------+--------------+---------------+
| 0x0           | Unassigned   | This document |
|               |              |               |
| 0x1           | IPv4         | This document |
|               |              |               |
| 0x2           | IPv6         | This document |
|               |              |               |
| 0x3           | Ethernet     | This document |
|               |              |               |
| 0x4           | NSH          | This document |
|               |              |               |
| 0x5           | MPLS         | This document |
|               |              |               |
| 0x6..0xFD     | Unassigned   |               |
|               |              |               |
| 0xFE          | Experiment 1 | This document |
|               |              |               |
| 0xFF          | Experiment 2 | This document |
+---------------+--------------+---------------+
```

              Table 7: NSH Base Header Next Protocol Values

11.2.6. New IETF assigned MD Type Registry

This document requests IANA to create a registry for the type values
owned by the IETF (i.e., MD Class set to 0x0000) called the "IETF
Assigned MD Type Registry."

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

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

12. References

12.1. 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>.

[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>.

[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,
           <http://www.rfc-editor.org/info/rfc8126>.

12.2. Informative References

[BCP38]   Ferguson, P. and D. Senie, "Network Ingress Filtering:
           Defeating Denial of Service Attacks which employ IP Source
           Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827,
           May 2000, <http://www.rfc-editor.org/info/rfc2827>.

[I-D.guichard-sfc-nsh-dc-allocation]
           Guichard, J., Smith, M., Surendra, S., Majee, S., Agarwal,
           P., Glavin, K., and Y. Laribi, "Network Service Header
           (NSH) Context Header Allocation (Data Center)", draft-
           guichard-sfc-nsh-dc-allocation-05 (work in progress),
           August 2016.

[I-D.ietf-nvo3-vxlan-gpe]
           Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol
           Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe-04 (work
           in progress), April 2017.

[I-D.ietf-rtgwg-dt-encap]
           Nordmark, E., Tian, A., Gross, J., Hudson, J., Kreeger,
           L., Garg, P., Thaler, P., and T. Herbert, "Encapsulation
           Considerations", draft-ietf-rtgwg-dt-encap-02 (work in
           progress), October 2016.

[I-D.ietf-sfc-control-plane]
           Boucadair, M., "Service Function Chaining (SFC) Control
           Plane Components & Requirements", draft-ietf-sfc-control-
           plane-08 (work in progress), October 2016.

   [I-D.ietf-sfc-oam-framework]
             Aldrin, S., Pignataro, C., Kumar, N., Akiya, N., Krishnan,
             R., and A. Ghanwani, "Service Function Chaining Operation,
             Administration and Maintenance Framework", draft-ietf-sfc-
             oam-framework-02 (work in progress), July 2017.

   [I-D.napper-sfc-nsh-broadband-allocation]
             Napper, J., Kumar, S., Muley, P., Henderickx, W., and M.
             Boucadair, "NSH Context Header Allocation -- Broadband",
             draft-napper-sfc-nsh-broadband-allocation-02 (work in
             progress), January 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,
             <http://www.rfc-editor.org/info/rfc2784>.

   [RFC3692] Narten, T., "Assigning Experimental and Testing Numbers
             Considered Useful", BCP 82, RFC 3692,
             DOI 10.17487/RFC3692, January 2004,
             <http://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,
             <http://www.rfc-editor.org/info/rfc6071>.

   [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>.

   [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>.

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