Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 22, 2016
March 21, 2016

P. Quinn, Ed.
Cisco Systems, Inc.
U. Elzur, Ed.
Intel

Network Service Header
draft-ietf-sfc-nsh-04.txt

Abstract

This draft describes a Network Service Header (NSH) inserted onto encapsulated 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 as per SFC Architecture [SFC-arch]

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].

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.

The current service function deployment models are 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 dataplane 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 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.

The SFC Architecture document [SFC-arch] 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 defined in that draft.

2.1. Definition of Terms

Classification: Locally instantiated matching of traffic flows against policy for subsequent application of the required set of network service functions. The policy may be customer/network/ service specific.

Service Function Forwarder (SFF): A service function forwarder is responsible for forwarding traffic to one or more connected service functions according to information carried in the NSH, as well as handling traffic coming back from the SF. Additionally, a service function forwarder is responsible for transporting traffic to another SFF (in the same or different type of overlay), and terminating the SFP.

Service Function (SF): A function that is responsible for specific treatment of received packets. A Service Function can act at various layers of a protocol stack (e.g., at the network layer or other OSI layers). As a logical component, a Service Function can be realized as a virtual element or be embedded in a physical network element. One or more Service Functions can be embedded in the same network element. Multiple occurrences of the Service Function can exist in the same administrative domain.

One or more Service Functions can be involved in the delivery of added-value services. A non-exhaustive list of abstract Service Functions includes: firewalls, WAN and application acceleration, Deep Packet Inspection (DPI), LI (Lawful Intercept), server load balancing, NAT44 [RFC3022], NAT64 [RFC6146], NPTv6 [RFC6296], HOST_ID injection, HTTP Header Enrichment functions, TCP optimizer.

An SF may be NSH-aware, that is it receives and acts on information in the NSH. The SF may also be NSH-unaware in which case data forwarded to the SF does not contain NSH.

Service Function Chain (SFC): A service function chain defines an ordered set of abstract service functions (SFs) and ordering constraints that must be applied to packets and/or frames and/or flows selected as a result of classification. An example of an abstract service function is "a firewall". The implied order may not be a linear progression as the architecture allows for SFCs that copy to more than one branch, and also allows for cases where there is flexibility in the order in which service functions need to be applied. The term service chain is often used as shorthand for service function chain.

Service Function Path (SFP): The Service Function Path is a constrained specification of where packets assigned to a certain service function path must go. While it may be so constrained as to identify the exact locations, it can also be less specific. The SFP provides a level of indirection between the fully abstract notion of service chain as a sequence of abstract service functions to be delivered, and the fully specified notion of exactly which SFF/SFs the packet will visit when it actually traverses the network. By allowing the control components to specify this level of indirection, the operator may control the degree of SFF/SF selection authority that is delegated to the network.

Network Node/Element: Device that forwards packets or frames based on outer header information.

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

Network Service Header: provides SFP identification, and is used by the NSH-aware functions, such as the Classifier, SFF and NSH-aware SFs. In addition to SFP identification, the NSH may carry data plane metadata.

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.

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

NSH Proxy: Removes and inserts NSH on behalf of an NSH-unaware service function. The proxy node removes the NSH header and delivers the original packet/frame via a local attachment circuit to the service function. Examples of a local attachment circuit include, but are not limited to: VLANs, IP in IP, GRE, VXLAN. When complete, the Service Function returns the packet to the NSH proxy via the same or different attachment circuit. The NSH Proxy, in turn, re-imposes NSH on the returned packets. Often, an SFF will act as an NSH-proxy when required.

2.2. Problem Space

Network Service Header (NSH) addresses several limitations associated with service function deployments today (i.e. prior to use of NSH). A short reference is included below, RFC 7498 [RFC7498], provides a more comprehensive review of the SFC Problem Statement.

1. Topological Dependencies: Network service deployments are often coupled to network topology. Such a dependency imposes constraints on the service delivery, potentially inhibiting the network operator from optimally utilizing service resources, and reduces the flexibility. This limits scale, capacity, and redundancy across network resources.

2. Service Chain Construction: Service function chains today are most typically built through manual configuration processes. These are slow and error prone. With the advent of newer dynamic service deployment models, the control/management planes provide not only connectivity state, but will also be increasingly utilized for the creation of network services. Such a control/ management planes could be centralized, or be distributed.

3. Application of Service Policy: Service functions rely on topology information such as VLANs or packet (re) classification to determine service policy selection, i.e. the service function specific action taken. Topology information is increasingly less viable due to scaling, tenancy and complexity reasons. The topological information is often stale, providing the operator with inaccurate service Function (SF) placement that can result in suboptimal resource utilization. Furthermore topology-centric information often does not convey adequate information to the service functions, forcing functions to individually perform more granular classification.

4. Per-Service (re)Classification: Classification occurs at each service function independent from previously applied service functions. More importantly, the classification functionality often differs per service function and service functions may not leverage the results from other service functions.

5. Common Header Format: Various proprietary methods are used to share metadata and create service paths. A standardized protocol provides a common format for all network and service devices.

6. Limited End-to-End Service Visibility: Troubleshooting service related issues is a complex process that involve both network-specific and service-specific expertise. This is especially the case, when service function chains span multiple DCs, or across

administrative boundaries. Furthermore, physical and virtual environments (network and service) can be highly divergent in terms of topology and that topological variance adds to these challenges.

7. Transport Dependence: Service functions can and will be deployed in networks with a range of transports requiring service functions to support and participate in many transports (and associated control planes) or for a transport gateway function to be present.

2.3. NSH-based Service Chaining

The NSH creates a dedicated service plane, that addresses many of the limitations highlighted in Section 2.2. More specifically, NSH enables:

1. Topological Independence: Service forwarding occurs within the service plane, via a network overlay, 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 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 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 network devices and service function, and between service functions. The semantics of the shared metadata is communicated via a control plane 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 carried in an overlay, over existing underlays. 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. The original packets preceded by NSH, are then encapsulated in an outer header for transport.

NSH is added by a Service Classifier. The NSH header is removed by the last SFF in the chain or by a SF that consumes the packet.

3.1. Network Service Header Format

A 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
 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 path.

Context headers: carry opaque metadata and variable length encoded information.

3.2. 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|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.

O bit: when set to 0x1 indicates that this packet is an operations and management (OAM) packet. The receiving SFF and SFs nodes MUST examine the payload and take appropriate action (e.g. return status information).

OAM message specifics and handling details are outside the scope of this document.

C bit: Indicates that a critical metadata TLV is present (see Section 3.4.2). 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. The C bit MUST be set to 0x0 when MD Type= 0x1 and MAY be used with MD Type = 0x2 and MUST be set to 0x1 if one or more critical TLVs are present.

All other flag fields are reserved for future use. Reserved bits MUST be set to zero 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 optional variable TLVs. The Length MUST be of value 0x6 for MD Type = 0x1 and MUST be of value 0x2 or higher for MD Type = 0x2. The NSH header length MUST be an integer number of 4 bytes. The length field MUST be used to determine 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. A new registry will be requested from IANA for the MD Type.

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, and 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 1 (as per the MUST) metadata, yet participate in the a network with MD-Type 2 metadata packets. In that case, the type-1 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 original packet. A new IANA registry will be created for protocol type.

This draft defines the following Next Protocol values:

0x1 : IPv4
0x2 : IPv6
0x3 : Ethernet
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 ID                      | Service Index |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Service path ID (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 first Classifier (i.e. at the boundary of the NSH domain)in the NSH Service Function Path, 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). A Classifier MUST send the packet to the first SFF in the chain. Service index MUST be decremented by service functions or proxy nodes after performing required services and the new decremented SI value MUST be

reflected in the egress NSH packet. SI MAY be used in conjunction with Service Path ID for Service Function Path selection. Service Index (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 loop detection.

3.4. NSH MD-type 1

When the Base Header specifies MD Type = 0x1, four Context Header, 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|R|   Length  |  MD-type=0x1  | Next Protocol |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Service Path ID                      | Service Index |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Mandatory Context Header                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Mandatory Context Header                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Mandatory Context Header                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Mandatory Context Header                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: NSH MD-type=0x1

Draft-dc [dcalloc] and draft-mobility [moballoc] 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|R|   Length  |  MD-type=0x2  | Next Protocol |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Service Path ID                      | 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          TLV Class            |C|    Type     |R|R|R|   Len   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Variable Metadata                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: Variable Context Headers

TLV Class: describes the scope of the "Type" field. In some cases, the TLV Class will identify a specific vendor, in others, the TLV Class will identify specific standards body allocated types. A new IANA registry will be created for TLV Class type.

Type: the specific type of information being carried, within the scope of a given TLV Class. Value allocation is the responsibility of the TLV Class owner.

Encoding the criticality of the TLV within the Type field is consistent with IPv6 option types: the most significant bit of the Type field indicates whether the TLV is mandatory for the receiver to understand/process. This effectively allocates Type values 0 to 127 for non-critical options and Type values 128 to 255 for critical options. 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 a receiver 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 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 4-byte words. 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 NSH 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. A service index = 0x0 indicates that a packet MUST be dropped by the SFF.

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

   If an NSH proxy (see Section 7) 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 NSH proxy receives an NSH-encapsulated packet, it MUST remove the NSH headers before forwarding it to an NSH unaware SF. When the NSH Proxy receives a packet back from an NSH unaware SF, it MUST re-encapsulate it with the correct NSH, and MUST also 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 component 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)				+		+
NSH 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 the ensure proper forwarding of NSH data, several options for handling fragmentation and re-assembly exist:

1. Jumbo Frames, when supported, enable the transport of NSH and associated transport packets without requiring fragmentation.

2. Path MTU Discovery [RFC1191]"describes a technique for dynamically discovering the maximum transmission unit (MTU) of an arbitrary internet path" and can be utilized to ensure the the required packet size is used.

3. [RFC6830] describes two schemes for fragmentation and re-assembly in section 5.4.

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 may 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 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 a simple, single next-hop SPI/SI to network overlay network locator mapping.

SPI	SI	NH	Transport
10	255	1.1.1.1	VXLAN-gpe
10	254	2.2.2.2	nvGRE
10	251	10.1.2.3	GRE
40	251	10.1.2.3	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.

SPI	SI	NH
10	3	SF2
245	12	SF34
40	9	SF9

Figure 10: NSH to SF Mapping Example

SF	NH	Transport
SF2	10.1.1.1	VXLAN-gpe
SF34	192.168.1.1	UDP
SF9	1.1.1.1	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) overlay links 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	10.1.1.1	1
		10.1.1.2	1
20	12	192.168.1.1	1
		10.2.2.2	1
30	7	10.2.2.3	10
		10.3.3.3	5

(encap type omitted for formatting)

Figure 12: NSH Weighted Service Path

7.2. Mapping NSH to Network Overlay

As described above, the mapping of SPI to network topology may result in a single overlay 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 10.1.1.1
   SFFa mapping: SPI=10 --> VXLAN-gpe, dst-ip: 10.1.1.1

2. Next SF is located at SFFc with multiple network locators for load distribution purposes:
   SFFb mapping: SPI=10 --> VXLAN-gpe, dst_ip:10.2.2.1, 10.2.2.2, 10.2.2.3, equal cost

3. Next SF is located at SFFd with two paths to SFFc, one for redundancy:
   SFFc mapping: SPI=10 --> VXLAN-gpe, dst_ip:10.1.1.1 cost=10, 10.1.1.2, 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 overlay paths as required. For example, the overlay path between service functions forwarders 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 function 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

In some cases, a service path is exactly that -- a linear list of service functions that must be traversed. However, the "path" is actually a directed graph. Furthermore, within a given service topology several directed graphs may exist with packets moving between graphs based on non-initial classification (in Figure 13, co-located with the SFs).

```
                     ,---.          ,---.            ,---.
                   /     \        /     \          /     \
                  (  SF2  +------+  SF7  +--------+  SF3  )
            ,------\     /        \     /       /-+      /
           ;        |---'          `---'\      /    `-+-'
           |        :                    \    /
           |         \                /---:---
         ,-+-.        `.     ,---.   /     :
        /     \         '---+     \/        \
       (  SF1  )           (  SF6  )        \
        \     /             \     +--.       :
         `---'               `---'    `-.  ,-+-.
                                         `+     \
                                         (  SF4  )
                                          \     /
                                           `---'
```

Figure 13: Service Graph Example

The SPI/SI combination provides a simple representation of a directed graph, the SPI represents a graph ID; and the SI a node ID. The service topology formed by SPI/SI support cycles, weighting, and alternate topology selection, all within the service plane. The realization of the network topology occurs as described above: SPI/ID mapping to an appropriate transport and associated next network hops.

NSH-aware services receive the entire header, including the SPI/SI. An non-initial logical classifier (in many deployment, this classifier will be co-resident with a SF) can now, based on local policy, alter the SPI, which in turn effects both the service graph, and in turn the selection of overlay at the SFF. The figure below depicts the policy associated with the graph in Figure 13 above. Note: this illustrates multiple graphs and their representation; it does not depict the use of metadata within a single service function graph.

```
SF1:
    SPI: 10
        NH: SF2
SF2:
    Class: Bad
            SPI: 20
            NH: SF6
     Class: Good
            SPI: 30
            NH: SF7
SF6:
     Class: Employee
            SPI: 21
            NH: SF4
     Class: Guest
            SPI: 22
            NH: SF3
SF7:
     Class: Employee
            SPI: 31
            NH: SF4
     Class: Guest
               SPI: 32
           NH: SF3
```

Figure 14: Service Graphs Using SPI

This example above does not show the mapping of the service topology to the network overlay topology. As discussed in the sections above, the overlay selection occurs as per network policy.

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 15: Metadata and Policy

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

Figure 16: 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.

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 18 illustrates an example of updating metadata.

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

Figure 17: Metadata Augmentation

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

Figure 18: Metadata Update

8.3. Service Path ID and Metadata

Metadata information may influence the service path selection since the Service Path Identifier can represent the result of classification. A given SPI can represent all or some of the metadata, and be updated based on metadata classification results. This relationship provides the ability to create a dynamic services 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 19 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 19: Path ID and Metadata

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

9. NSH Encapsulation Examples

9.1. GRE + NSH

```
    IPv4 Packet:
   +----------+--------------------+--------------------+
   |L2 header | L3 header, proto=47|GRE header,PT=0x894F|
   +----------+--------------------+--------------------+
   --------------+----------------+
   NSH, NP=0x1   |original packet |
   --------------+----------------+



    L2 Frame:
    +----------+--------------------+--------------------+
    |L2 header | L3 header, proto=47|GRE header,PT=0x894F|
    +----------+--------------------+--------------------+
    ---------------+---------------+
    NSH, NP=0x3    |original frame |
    ---------------+---------------+
```

Figure 20: GRE + NSH

9.2. VXLAN-gpe + NSH

```
    IPv4 Packet:
    +----------+------------------------+---------------------+
    |L2 header | IP + UDP dst port=4790 |VXLAN-gpe NP=0x4(NSH)|
    +----------+------------------------+---------------------+
    --------------+----------------+
    NSH, NP=0x1   |original packet |
    --------------+----------------+


    L2 Frame:
    +----------+------------------------+---------------------+
    |L2 header | IP + UDP dst port=4790 |VXLAN-gpe NP=0x4(NSH)|
    +----------+------------------------+---------------------+
    ---------------+---------------+
    NSH,NP=0x3     |original frame |
    ---------------+---------------+
```

Figure 21: VXLAN-gpe + NSH

9.3. Ethernet + NSH

```
IPv4 Packet:
+-------------------------------+---------------+--------------------+
|Outer Ethernet, ET=0x894F      | NSH, NP = 0x1 | original IP Packet |
+-------------------------------+---------------+--------------------+


L2 Frame:
+-------------------------------+---------------+----------------+
|Outer Ethernet, ET=0x894F      | NSH, NP = 0x3 | original frame |
+-------------------------------+---------------+----------------+
```

Figure 22: Ethernet + NSH

10. 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. RFC 2119 [RFC6071]) can be used.

Similarly if confidentiality is required, existing encryption protocols can be used in conjunction with encapsulated NSH.

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

11. Open Items for WG Discussion

1. MD type 1 metadata semantics specifics

2. Bypass bit in NSH.

3. Rendered Service Path ID (RSPID).

12. 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
Garg.Pankaj@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

13. Acknowledgments

The authors would like to thank 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 draft.

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

14. IANA Considerations

14.1. NSH EtherType

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

14.2. 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.

14.2.1. 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)
Bits 2-9 - Reserved

14.2.2. 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

14.2.3. TLV Class Registry

IANA is requested to set up a registry of "TLV Types". These are 16-bit values. Registry entries are assigned by using the "IETF Review" policy defined in RFC 5226 [RFC5226].

14.2.4. 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 and 3 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..253	Unassigned	
254	Experiment 1	This document
255	Experiment 2	This document

Table 2

15. References

15.1. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <http://www.rfc-editor.org/info/rfc791>.

[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>.

15.2. 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>.

[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>.

[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>.

[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>.

[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>.

[SFC-arch] Quinn, P., Ed. and J. Halpern, Ed., "Service Function Chaining (SFC) Architecture", 2014, <http://datatracker.ietf.org/doc/draft-quinn-sfc-arch>.

[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/>.

[dcalloc] Guichard, J., Smith, M., and S. Kumar, "Network Service Header (NSH) Context Header Allocation (Data Center)", 2014, <https://datatracker.ietf.org/doc/draft-guichard-sfc-nsh-dc-allocation/>.

[moballoc]
Napper, J. and S. Kumar, "NSH Context Header Allocation -- Mobility", 2014, <https://datatracker.ietf.org/doc/draft-napper-sfc-nsh-mobility-allocation/>.

[nsh-sec] Reddy, T., Migault, D., Pignataro, C., Quinn, P., and C. Inacio, "NSH Security and Privacy requirements", 2016, <https://datatracker.ietf.org/doc/draft-reddy-sfc-nsh-security-req/>.

Authors' Addresses

Paul Quinn (editor)
Cisco Systems, Inc.

Email: paulq@cisco.com

Uri Elzur (editor)
Intel

Email: uri.elzur@intel.com