

Service Function Chaining
Internet-Draft
Intended status: Informational
Expires: December 29, 2016

D. Dolson
Sandvine
S. Homma
NTT
D. Lopez
Telefonica I+D
M. Boucadair
Orange
D. Liu
Alibaba Group
T. Ao
ZTE Corporation
V. Vu
SSU
June 27, 2016

Hierarchical Service Function Chaining (hSFC)
draft-dolson-sfc-hierarchical-06

Abstract

Hierarchical Service Function Chaining (hSFC) is a network architecture allowing an organization to compartmentalize a large-scale network into multiple domains of administration.

The goals of hSFC are to make a large-scale network easier to reason about, simpler to control and to able support independent functional groups within large operators.

Status of This Memo

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

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

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

This Internet-Draft will expire on December 29, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Hierarchical Service Function Chaining (hSFC)	4
2.1. Top Level	4
2.2. Lower Levels	5
3. Internal Boundary Node (IBN)	7
3.1. IBN Path Configuration	7
3.1.1. Flow-Stateful IBN	7
3.1.2. Encoding Upper-Level Paths in Metadata	9
3.1.3. Using Unique Paths per Upper-Level Path	9
3.1.4. Nesting Upper-Level NSH within Lower-Level NSH	10
3.1.5. Stateful / Metadata Hybrid	11
3.2. Gluing Levels Together	12
3.3. Decrementing Service Index	12
4. Sub-domain Classifier	13
5. Control Plane Elements	13
6. Extension for Adopting to NSH-Unaware Service Functions	14
6.1. Purpose	15
6.2. Requirements for IBN	16
7. Acknowledgements	16
8. IANA Considerations	17
9. Security Considerations	17
10. References	17
10.1. Normative References	17
10.2. Informative References	18
Appendix A. Examples of Hierarchical Service Function Chaining	18
A.1. Reducing the Number of Service Function Paths	18
A.2. Managing a Distributed Data-Center Network	20
Authors' Addresses	22

1. Introduction

Service Function Chaining (SFC) is a technique for prescribing differentiated traffic forwarding policies within an SFC-enabled domain. SFC is described in detail in the SFC architecture document [RFC7665], and is not repeated here.

In this document we consider the difficult problem of implementing SFC across a large, geographically dispersed network comprised of millions of hosts and thousands of network forwarding elements, involving multiple operational teams (with varying functional responsibilities). We expect asymmetrical routing is inherent in the network, while recognizing that some Service Functions (SFs) require bidirectional traffic for transport-layer sessions (e.g., NATs, firewalls). We assume that some Service Function Paths (SFPs) need to be selected on the basis of application-specific data visible to the network, with transport-layer coordinate (typically, 5-tuple) stickiness to specific SF instances.

Note: in this document, the notion of the "path" of a packet is the series of SF instances traversed by a packet. The means of delivering packets between SFs (the forwarding mechanisms enforced in the underlying network) is not relevant to the discussion.

Difficult problems are often made easier by decomposing them in a hierarchical (nested) manner. So instead of considering an omniscient SFC Control Plane that can manage (create, withdraw, supervise, etc.) complete SFPs from one end of the network to the other, we decompose the network into smaller sub-domains. Each sub-domain may support a subset of the network applications or a subset of the users. The criteria for determining decomposition into SFC-enabled sub-domains are beyond the scope of this document.

Note that decomposing a network into multiple SFC-enabled domains should permit end-to-end visibility of SFs and SFPs. Decomposition should also be implemented with special care to ease monitoring and troubleshooting of the network and services as a whole.

An example of simplifying a network by using multiple SF domains is further discussed in [I-D.ietf-sfc-dc-use-cases].

We assume the SFC-aware nodes use NSH [I-D.ietf-sfc-nsh] or a similar labeling mechanism.

The "domains" discussed in this document are assumed to be under control of a single organization, such that there is a strong trust relationship between the domains. The intention of creating multiple domains is to improve the ability to operate a network. It is

outside of the scope of the document to consider domains operated by different organizations.

2. Hierarchical Service Function Chaining (hSFC)

A hierarchy has multiple levels. The top-most level encompasses the entire network domain to be managed, and lower levels encompass portions of the network.

2.1. Top Level

Considering the example depicted in Figure 1, a top-level network domain includes SFC data plane components distributed over a wide area, including:

- o Classifiers (CFs),
- o Service Function Forwarders (SFFs) and
- o Sub-domains.

For the sake of clarity, components of the underlay network are not shown; an underlay network is assumed to provide connectivity between SFC data plane components.

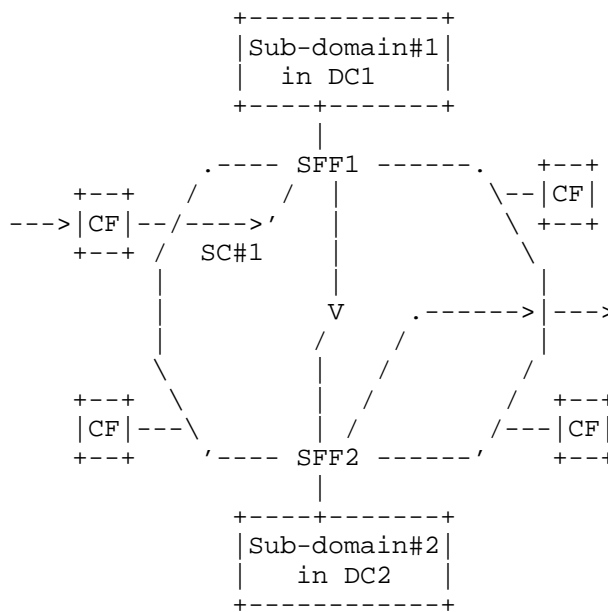
Top-level SFPs carry packets from classifiers through a series of SFFs and sub-domains, with the operations within sub-domains being opaque to the higher levels.

We expect the system to include a top-level control-plane having responsibility for configuring forwarding and classification (see [I-D.ietf-sfc-control-plane]). The top-level Service Chaining control-plane manages end-to-end service chains and associated service function paths from network edge points to sub-domains and configuring top-level classifiers at a coarse level (e.g., based on source or destination host) to forward traffic along paths that will transit appropriate sub-domains. Figure 1 shows one possible service chain passing from edge, through two sub-domains, to network egress. The top-level control plane does not configure classification or forwarding within the sub-domains.

At this network-wide level, the number of SFPs required is a linear function of the number of ways in which a packet is required to traverse different sub-domains and egress the network. Note that the various paths which may be taken within a sub-domain are not represented by distinct network-wide SFPs; specific policies at the ingress nodes of each sub-domain bind flows to sub-domain paths.

Packets are classified at the edge of the network to select the paths by which sub-domains are to be traversed. At the ingress of each sub-domain, paths are reclassified to select the paths by which SFs in the sub-domain are to be traversed. At the egress of each sub-domain, packets are returned to the top-level paths. Contrast this with an approach requiring the top-level classifier to select paths to specify all of the SFs in each sub-domain.

It should be assumed that some SFs require bidirectional symmetry of paths (see more in Section 4). Therefore the classifiers at the top level must be configured with policies ensuring outgoing packets take the reverse path of incoming packets through sub-domains.



One path is shown from edge classifier to SFF1 to Sub-domain#1 (residing in data-center1) to SFF1 to SFF2 (residing in data-center 2) to Sub-domain#2 to SFF2 to network egress.

Figure 1: Network-wide view of top level of hierarchy

2.2. Lower Levels

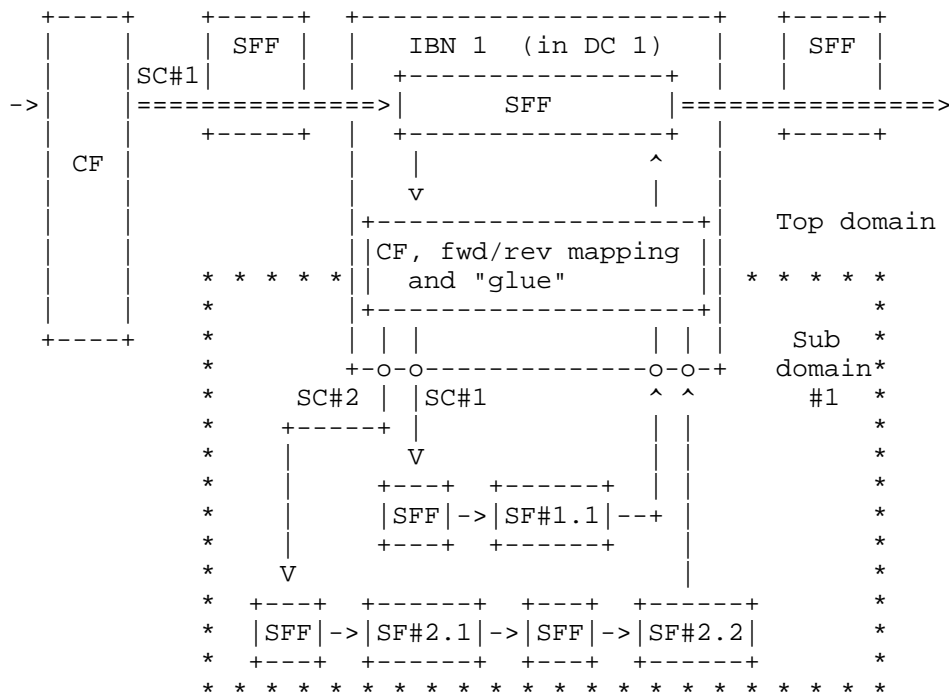
Each of the sub-domains in Figure 1 is an SFC-enabled domain.

Unlike the top level, data packets entering the sub-domain are already SFC-encapsulated. Figure 2 shows a sub-domain interfaced with a higher-level domain by means of an Internal Boundary Node

(IBN). It is the purpose of the IBN to apply classification rules and direct the packets to the selected local SFPs terminating at an egress IBN. The egress IBN finally restores packets to the original SFC shim and hands them off to SFFs.

Each sub-domain intersects a subset of the total paths that are possible in the higher-level domain. An IBN is concerned with higher-level paths, but only those traversing its sub-domain. A top-level control element may configure the IBN as an SF (i.e., the IBN plays the SF role in the top-level domain).

Each sub-domain is likely to have a control-plane that can operate independently of the top-level control-plane. The sub-domain control-plane configures the classification and forwarding rules in the sub-domain. The classification rules reside in the IBN, where SFC encapsulation of the top-level domain is converted to/from SFC encapsulation of the lower-level domain.



```
*** Sub-domain boundary; == top-level chain; --- low-level chain.
```

Figure 2: Sub-domain within a higher-level domain

If desired, the pattern can be applied recursively. For example, SF#1.1 in Figure 2 could be a sub-domain of the sub-domain.

3. Internal Boundary Node (IBN)

A network element termed "Internal Boundary Node" (IBN) bridges packets between domains. It behaves as an SF to the higher level, and looks like a classifier and end-of-chain to the lower level.

To achieve the benefits of hierarchy, the IBN should be applying more granular traffic classification rules at the lower level than the traffic passed to it. This means that the number of SFPs within the lower level is greater than the number of SFPs arriving to the IBN.

The IBN is also the termination of lower-level SFPs. This is because the packets exiting lower-level SF paths must be returned to the higher-level SF paths and forwarded to the next hop in the higher-level domain.

3.1. IBN Path Configuration

An operator of a lower-level domain may be aware of which high-level paths transit their domain, or they may wish to accept any paths.

When packets enter the sub-domain, the Service Path Identifier (SPI) and Service Index (SI) are re-marked according to the path selected by the classifier.

After exiting a path in the sub-domain, packets can be restored to an original upper-level SFP by these methods:

1. Saving SPI and SI in transport-layer flow state,
2. Pushing SPI and SI into metadata,
3. Using unique lower-level paths per upper-level path coordinates,
4. Nesting NSH headers, encapsulating the higher-level NSH headers within the lower-level NSH headers,
5. Saving upper-level by a flow ID and placing an hSFC flow ID into metadata,

3.1.1. Flow-Stateful IBN

An IBN can be flow-aware, returning packets to the correct higher-level SFP on the basis of the transport-layer coordinates (typically, a 5-tuple) of packets exiting the lower-level SFPs.

When packets are received by the IBN on a higher-level path, the encapsulated packets are parsed for IP and transport-layer (TCP, UDP, etc.) coordinates. State is created, indexed by these coordinates ({source-IP, destination-IP, source-port, destination-port and transport protocol} typically). The state contains at least critical fields of the encapsulating SFC header (or perhaps the entire header).

The simplest approach has the packets return to the same IBN at the end of the chain that classified the packet at the start of the chain. This is because the required transport-coordinates state is rapidly changing and most efficiently kept locally. If the packet is returned to a different IBN for egress, transport-coordinates state must be synchronized between the IBNs.

When a packet returns to the IBN at the end of a chain, the SFC header is removed, the packet is parsed for IP and transport-layer coordinates, and state is retrieved from them. The state contains the information required to forward the packet within the higher-level service chain.

State cannot be created by packets arriving from the lower-level chain; when state cannot be found for such packets, they must be dropped.

This stateful approach is limited to use with SFs that retain the transport coordinates of the packet. This approach cannot be used with SFs that modify those coordinates (e.g., NATs) or otherwise create packets for new coordinates other than those received (e.g., as an HTTP cache might do to retrieve content on behalf of the original flow). In both cases, the fundamental problem is the inability to forward packets when state cannot be found for the packet transport-layer coordinates.

In the stateful approach, there are issues caused by having state, such as how long the state should be maintained (it must time out eventually), as well as whether the state needs to be replicated to other devices to create a highly available network.

It is valid to consider the state to be disposable after failure, since it can be re-created by each new packet arriving from the higher-level domain. For example, if an IBN loses all flow state, the state is re-created by an end-point retransmitting a TCP packet.

If an SFC domain handles multiple network regions (e.g., multiple private networks), the coordinates may be augmented with additional parameters, perhaps using some metadata to identify the network region.

In this stateful approach, it is not necessary for the sub-domain's control-plane to modify paths when higher-level paths are changed. The complexity of the higher-level domain does not cause complexity in the lower-level domain.

Since it doesn't depend on NSH in the lower domain, this flow-stateful approach can be applied to translation methods of converting NSH to other forwarding techniques. (Refer to Section 6.)

3.1.2. Encoding Upper-Level Paths in Metadata

An IBN can push the upper-level Service Path Identifier (SPI) and Service Index (SI) (or encoding thereof) into a metadata field of the lower-level encapsulation (e.g., placing upper-level path information into a metadata field of NSH). When packets exit the lower-level path, the upper-level SPI and SI can be restored from the metadata retrieved from the packet.

This approach requires the SFs in the path to be capable of forwarding the metadata and appropriately attaching metadata to any packets injected for a flow.

Using new metadata may inflate packet size when variable-length metadata (type 2 from NSH [I-D.ietf-sfc-nsh]) is used.

It is conceivable that the MD-type 1 Mandatory Context Header fields of NSH [I-D.ietf-sfc-nsh] are not all relevant to the lower-level domain. In this case, one of the metadata slots of the Mandatory Context Header could be repurposed within the lower-level domain, and restored when leaving.

In this metadata approach, it is not necessary for the sub-domain's control element to modify paths when higher-level paths are changed. The complexity of the higher-level domain does not cause complexity in the lower-level domain.

3.1.3. Using Unique Paths per Upper-Level Path

In this approach, paths within the sub-domain are constrained so that a SPI (of the sub-domain) unambiguously indicates the egress SPI and SI (of the upper domain). This allows the original path information to be restored at sub-domain egress from a look-up table using the sub-domain SPI.

Whenever the upper-level domain provisions a path via the lower-level domain, the lower-level domain controller must provision corresponding paths to traverse the lower-level domain.

A down-side of this approach is that the number of paths in the lower-level domain is multiplied by the number of paths in the higher-level domain that traverse the lower-level domain. I.e., a sub-path must be created for each combination of upper SPI/SI and lower chain.

3.1.4. Nesting Upper-Level NSH within Lower-Level NSH

In this approach, when packets arrive at the IBN in the top-level domain, the classifier in the IBN determines the path for the lower-level domain and pushes the new NSH header in front of the original NSH header.

As shown in Figure 3 the Lower-NSH Header used to forward packets in the lower-level domain precedes the Upper-NSH Header from the top-level domain.

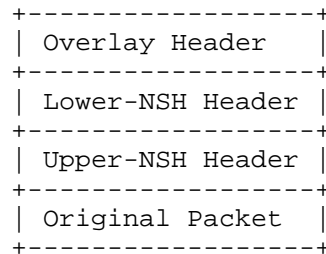


Figure 3: Encapsulation of NSH within NSH

The traffic with the above stack of two-layer-NSH header is to be forwarded according to the Lower-NSH header in the lower-level SFC domain. The Upper-NSH header is preserved in the packets but not used for forwarding. At the last SFF of the chain of the lower-level domain (which resides in the IBN), the Lower-NSH header is removed from the packet, and then the packet is forwarded by the IBN to an SFF of the upper-level domain, which will be forwarded according to the Upper-NSH header.

With such encapsulation, Upper-NSH information is carried along the extent of the lower-level chain without modification.

A benefit of this approach is that it does not require state in the IBN or configuration to encode fields in meta-data.

However, the down-side is it does require SFs in the lower-level domain to be able to parse multiple layers of NSH. If the SF injects

packets, it must also be able to deal with adding appropriate multiple layers of headers to injected packets.

3.1.5. Stateful / Metadata Hybrid

The basic idea of this approach is for the IBN to save upper domain encapsulation information such that it can be retrieved by a unique identifier, termed an "hSFC Flow ID". An example ID is shown in Table 1.

hSFC Flow ID	SPI	SI	Context1	Context2	Context3	Context4
1	45	254	100	2112	12345	7

Table 1: Example Mapping of an hSFC Flow ID to Upper-Level Header

The ID is placed in the metadata in NSH headers of the packet in the lower domain, as shown in Figure 4. When packets exit the lower domain, the IBN uses the ID to retrieve the appropriate NSH encapsulation for returning the packet to the upper domain.

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	R	Length	MD-type=0x1	Next Protocol																			
Service Path Identifier															Service Index																
hSFC Flow ID																															
Mandatory Context Header																															
Mandatory Context Header																															
Mandatory Context Header																															

Figure 4: Storing hSFC Flow ID in lower-level metadata

Advantages of this approach include:

- o Does not require state based on 5-tuple, so it works with functions that change the IP addresses or ports of a packet such as NATs,

- o Does not require all domains to have the same metadata scheme,
- o Can be used to restore any upper-domain information, not just service path,
- o The lower domain only requires a single item of metadata regardless of the number of items of metadata used in the upper domain. (For MD-Type 1, this leaves 3 slots for use in the lower domain.)
- o No special functionality is required of the SF, other than the usual ability to preserve metadata and to apply metadata to injected packets.

Disadvantages include those of other stateful approaches, including state timeout and replication mentioned in Section 3.1.1.

There may be a large number of unique NSH encapsulations to be stored, given that the hSFC Flow ID must represent all of the bits in the upper-level encapsulation. This might consume a lot of memory or create out-of-memory situations in which IDs cannot be created or old IDs are discarded while still in use.

3.2. Gluing Levels Together

The SPI or metadata on a packet received by the IBN may be used as input to reclassification and path selection within the lower-level domain.

In some cases the meanings of the various path IDs and metadata must be coordinated between domains.

One approach is to use well-known identifier values in metadata, communicated by some organizational registry.

Another approach is to use well-known labels for chain identifiers or metadata, as an indirection to the actual identifiers. The actual identifiers can be assigned by control-plane systems. For example, a sub-domain classifier could have a policy, "if pathID=classA then chain packet to path 1234"; the higher-level controller would be expected to configure the concrete higher-level pathID for classA.

3.3. Decrementing Service Index

Because the IBN acts as a Service Function to the higher-level domain, it must decrement the Service Index in the NSH headers of the higher-level path.

A good strategy seems to be to do this when the packet is first received by the IBN, before applying any of the strategies of Section 3.1, immediately prior to classification.

4. Sub-domain Classifier

Within the sub-domain (referring to Figure 2), after the IBN removes higher-level encapsulation from incoming packets, it sends the packets to the classifier, which selects the encapsulation for the packet within the sub-domain.

One of the goals of the hierarchical approach is to make it easy to have transport-flow-aware service chaining with bidirectional paths. For example, it is desired that for each TCP flow, the client-to-server packets traverse the same SFs as the server-to-client packets, but in the opposite sequence. We call this bidirectional symmetry. If bidirectional symmetry is required, it is the responsibility of the control-plane to be aware of symmetric paths and configure the classifier to chain the traffic in a symmetric manner.

Another goal of the hierarchical approach is to simplify the mechanisms of scaling in and scaling out service functions. All of the complexities of load-balancing among multiple SFs can be handled within a sub-domain, under control of the classifier, allowing the higher-level domain to be oblivious to the existence of multiple SF instances.

Considering the requirements of bidirectional symmetry and load-balancing, it is useful to have all packets entering a sub-domain to be received by the same classifier or a coordinated cluster of classifiers. There are both stateful and stateless approaches to ensuring bidirectional symmetry.

5. Control Plane Elements

Although control protocols have not yet been standardized, from the point of view of hierarchical service function chaining we have these expectations:

- o Each control-plane instance manages a single level of hierarchy of a single domain.
- o Each control-plane is agnostic about other levels of hierarchy. This aspect allows humans to reason about the system within a single domain and allows control-plane algorithms to use only domain-local inputs. Top-level control does not need visibility to sub-domain policies, nor does sub-domain control need visibility to higher-level policies.

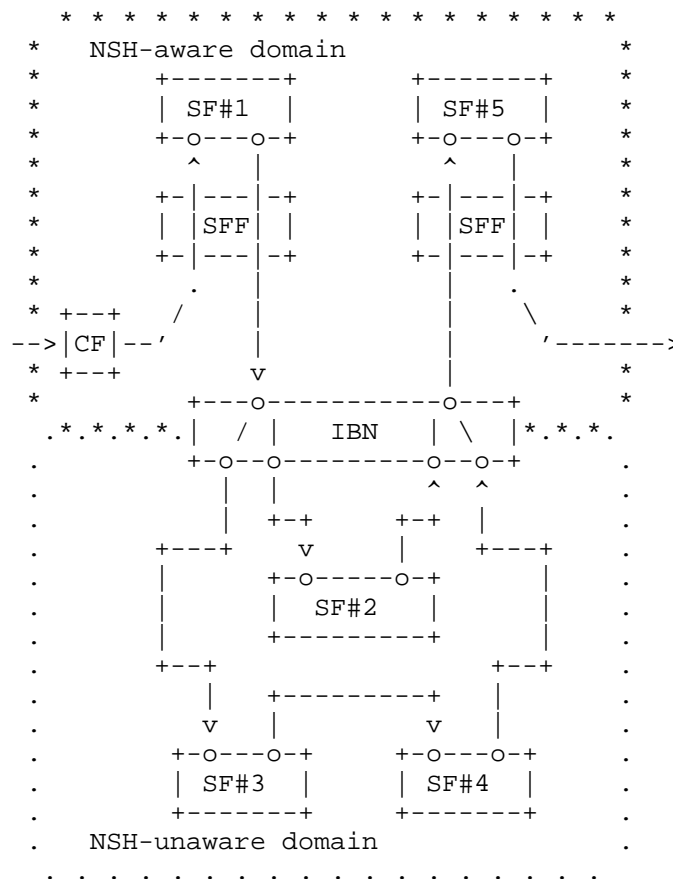
- o Sub-domain control-planes are agnostic about control-planes of other sub-domains. This allows both humans and machines to manipulate sub-domain policy without considering policies of other domains.

Recall that the IBN acts as an SF in the higher-level domain (receiving SF instructions from the higher-level control-plane) and as a classifier in the lower-level domain (receiving classification rules from the sub-domain control-plane). In this view, it is the IBN that glues the layers together.

The above expectations are not intended to prohibit network-wide control. A control hierarchy can be envisaged to distribute information and instructions to multiple domains and sub-domains. Control hierarchy is outside the scope of this document.

6. Extension for Adopting to NSH-Unaware Service Functions

The hierarchical approach can be used for dividing networks into NSH-aware and NSH-unaware domains by converting NSH encapsulation to other forwarding techniques (e.g., 5-tuple-based routing with OpenFlow), as shown in Figure 5.



SF#1 and SF#5 are NSH-aware and SF#2, SF#3 and SF#4 are NSH-unaware. In the NSH-unaware domain, packets are conveyed in a format supported by SFs which are deployed there.

Figure 5: Dividing NSH-aware and NSH-unaware domains

6.1. Purpose

This approach is expected to facilitate service chaining in networks in which NSH-aware and NSH-unaware SFs coexist. Some examples of such situations are:

- o In a period of transition from legacy SFs to NSH-aware SFs and
- o Supporting multi-tenancy.

6.2. Requirements for IBN

In this usage, an IBN classifier is required to have an NSH conversion table for applying packets to appropriate lower-level paths and returning packets to the correct higher-level paths. For example, the following methods would be used for saving/restoring upper-level path information:

- o Saving SPI and SI in transport-layer flow state (refer to Section 3.1.1) and
- o Using unique lower-level paths per upper-level NSH coordinates (refer to Section 3.1.3).

Especially, the use of unique paths approach would be good for translating NSH to a different forwarding technique in the lower level. A single path in the upper level may be branched to multiple paths in the lower level such that any lower-level path is only used by one upper-level path. This allows unambiguous restoration to the upper-level path.

In addition, an IBN might be required to convert metadata contained in NSH to the format appropriate to the packet in the lower-level path. For example, some legacy SFs identify subscriber based on information of network topology, such as VID, and IBN would be required to create VLAN to packets from metadata if subscriber identifier is conveyed as metadata in higher-level domains.

Other fundamental functions required as IBN (e.g., maintaining metadata of upper level or decrementing Service Index) are same as normal usage.

7. Acknowledgements

The concept of Hierarchical Service Path Domains was introduced in [I-D.homma-sfc-forwarding-methods-analysis] as a means to improve scalability of service chaining in large networks.

The concept of nested NSH headers was introduced in [I-D.ao-sfc-for-dc-interconnect] as a means of creating hierarchical SFC in a data center.

The authors would like to thank the following individuals for providing valuable feedback:

Ron Parker

Christian Jacquenet

Jie Cao

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

Hierarchical service function chaining makes use of service chaining architecture, and hence inherits the security considerations described in the architecture document.

Furthermore, hierarchical service function chaining inherits security considerations of the data-plane protocols (e.g., NSH) and control-plane protocols used to realize the solution.

The systems described in this document bear responsibility for forwarding internet traffic. In some cases the systems are responsible for maintaining separation of traffic in private networks.

This document describes systems within different domains of administration that must have consistent configurations in order to properly forward traffic and to maintain private network separation. Any protocol designed to distribute the configurations must be secure from tampering.

All of the systems and protocols must be secure from modification by untrusted agents.

Security considerations related to the control plane are discussed in [I-D.ietf-sfc-control-plane].

10. References

10.1. Normative References

[I-D.ietf-sfc-control-plane]

Boucadair, M., "Service Function Chaining (SFC) Control Plane Components & Requirements", draft-ietf-sfc-control-plane-06 (work in progress), May 2016.

[I-D.ietf-sfc-nsh]

Quinn, P. and U. Elzur, "Network Service Header", draft-ietf-sfc-nsh-05 (work in progress), May 2016.

[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>>.

10.2. Informative References

[I-D.ao-sfc-for-dc-interconnect]
Ao, T. and W. Bo, "Hierarchical SFC for DC Interconnection", draft-ao-sfc-for-dc-interconnect-01 (work in progress), October 2015.

[I-D.homma-sfc-forwarding-methods-analysis]
Homma, S., Naito, K., Lopez, D., Stiemerling, M., Dolson, D., Gorbunov, A., Leymann, N., Bottorff, P., and d. don.fedyk@hpe.com, "Analysis on Forwarding Methods for Service Chaining", draft-homma-sfc-forwarding-methods-analysis-05 (work in progress), January 2016.

[I-D.ietf-sfc-dc-use-cases]
Surendra, S., Tufail, M., Majee, S., Captari, C., and S. Homma, "Service Function Chaining Use Cases In Data Centers", draft-ietf-sfc-dc-use-cases-02 (work in progress), January 2015.

Appendix A. Examples of Hierarchical Service Function Chaining

The advantage of hierarchical service function chaining compared with normal or flat service function chaining is that it can reduce the management complexity significantly. This section discusses examples that show those advantages.

A.1. Reducing the Number of Service Function Paths

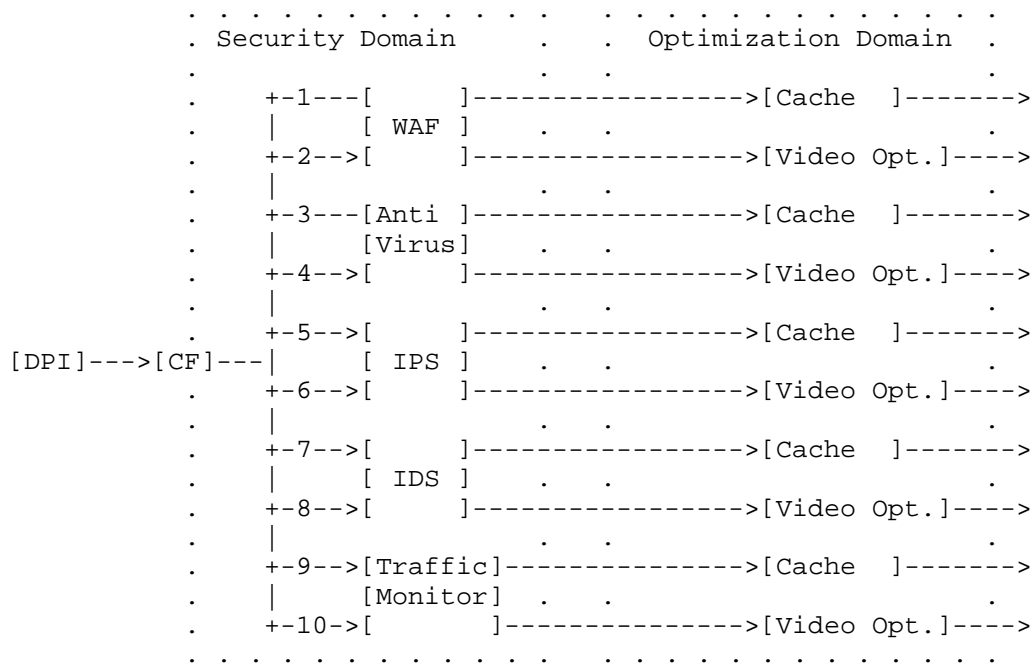
In this case, hierarchical service function chaining is used to simplify service function chaining management by reducing the number of Service Function Paths.

As shown in Figure 6, there are two domains, each with different concerns: a Security Domain that selects Service Functions based on network conditions and an Optimization Domain that selects Service Functions based on traffic protocol.

In this example there are five security functions deployed in the Security Domain. The Security Domain operator wants to enforce the five different security policies, and the Optimization Domain operator wants to apply different optimizations (either cache or video optimization) to each of these two types of traffic. If we use

flat SFC (normal branching), 10 SFPs are needed in each domain. In contrast, if we use hierarchical SFC, only 5 SFPs in Security Domain and 2 SFPs in Optimization Domain will be required, as shown in Figure 7.

In the flat model, the number of SFPs is the product of the number of functions in all of the domains. In the hSFC model, the number of SFPs is the sum of the number of functions. For example, adding a "bypass" path in the Optimization Domain would cause the flat model to require 15 paths (5 more), but cause the hSFC model to require one more path in the Optimization Domain.



The classifier must select paths that determine the combination of Security and Optimization concerns. 1:WAF+Cache, 2:WAF+VideoOpt, 3:AntiVirus+Cache, 4:AntiVirus+VideoOpt, 5: IPS+Cache, 6:IPS+VideoOpt, 7:IDS+Cache, 8:IDS+VideoOpt, 9:TrafficMonitor+Cache, 10:TrafficMonitor+VideoOpt

Figure 6: Flat SFC (normal branching)

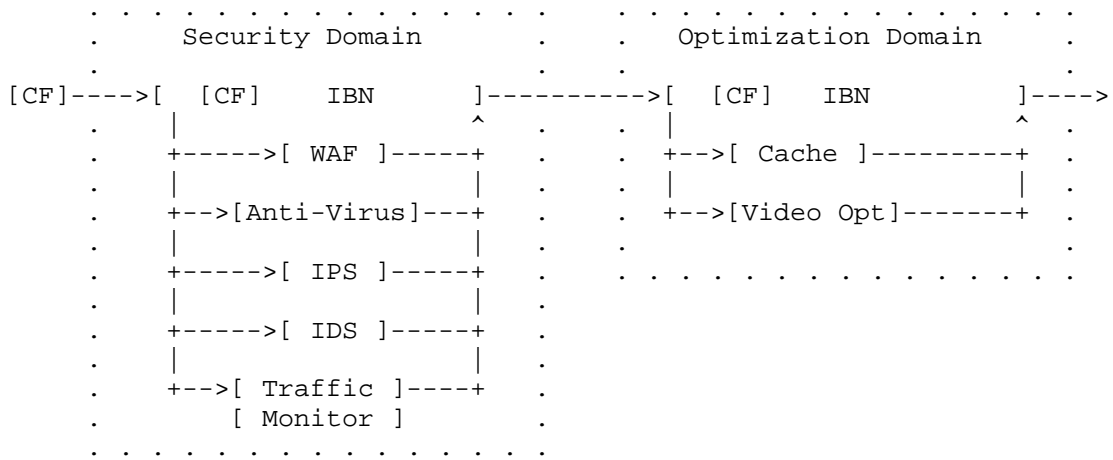


Figure 7: Simplified path management with Hierarchical SFC

A.2. Managing a Distributed Data-Center Network

Hierarchical service function chaining can be used to simplify inter-data-center SFC management. In the example of Figure 8, shown below, there is a central data center (Central DC) and multiple local data centers (Local DC#1, #2, #3) that are deployed in a geographically distributed manner. All of the data centers are under a single administrative domain.

The central DC may have some service functions that the local DC needs, such that the local DC needs to chain traffic via the central DC. This could be because:

- o Some service functions are deployed as dedicated hardware appliances, and there is a desire to lower the cost (both CAPEX and OPEX) of deploying such service functions in all data centers.
- o Some service functions are being trialed, introduced or otherwise handle a relatively small amount of traffic. It may be cheaper to manage these service functions in a single central data center and steer packets to the central data center than to manage these service functions in all data centers.

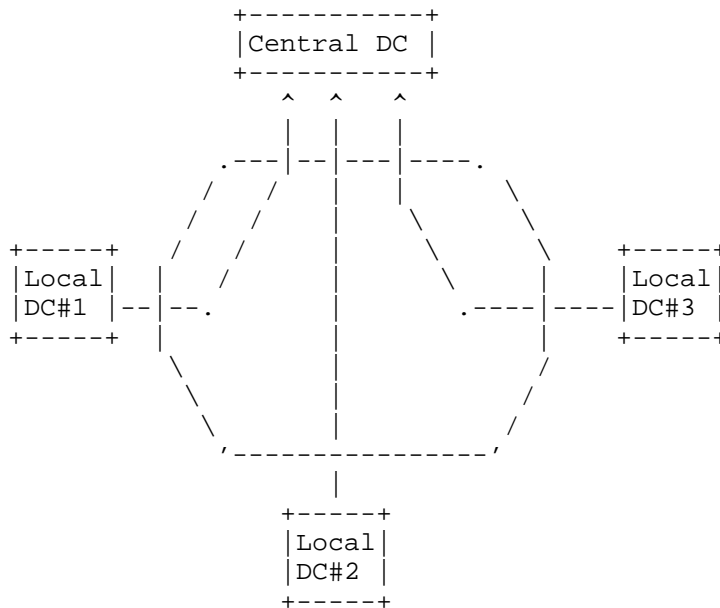


Figure 8: Simplify inter-DC SFC management

For large data center operators, one local DC may have tens of thousands of servers and hundred of thousands of virtual machines. SFC can be used to manage user traffic. For example, SFC can be used to classify user traffic based on service type, DDoS state etc.

In such large scale data center, using flat SFC is very complex, requiring a super-controller to configure all data centers. For example, any changes to Service Functions or Service Function Paths in the central DC (e.g., deploying a new SF) would require updates to all of the Service Function Paths in the local DCs accordingly. Furthermore, requirements for symmetric paths add additional complexity when flat SFC is used in this scenario.

Conversely, if using hierarchical SFC, each data center can be managed independently to significantly reduce management complexity. Service Function Paths between data centers can represent abstract notions without regard to details within data centers. Independent controllers can be used for the top level (getting packets to pass the correct data centers) and local levels (getting packets to specific SF instances).

Authors' Addresses

David Dolson
Sandvine
408 Albert Street
Waterloo, ON N2L 3V3
Canada

Phone: +1 519 880 2400
Email: ddolson@sandvine.com

Shunsuke Homma
NTT, Corp.
3-9-11, Midori-cho
Musashino-shi, Tokyo 180-8585
Japan

Email: homma.shunsuke@lab.ntt.co.jp

Diego R. Lopez
Telefonica I+D
Don Ramon de la Cruz, 82
Madrid 28006
Spain

Phone: +34 913 129 041
Email: diego.r.lopez@telefonica.com

Mohamed Boucadair
Orange
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Dapeng Liu
Alibaba Group
Beijing 100022
China

Email: max.ldap@alibaba-inc.com

Ting Ao
ZTE Corporation
No.889,Bibo Rd.,Zhangjiang Hi-tech Park
Shanghai 201203
China

Phone: +86-21-688976442
Email: ao.ting@zte.com.cn

Vu Anh Vu
Soongsil University
369 Sangdo-ro
Seoul, Dongjak-gu 06978
Korea

Email: vuva@dcn.ssu.ac.kr

Service Function Chaining
Internet-Draft
Intended status: Standards Track
Expires: May 7, 2018

P. Quinn, Ed.
Cisco
U. Elzur, Ed.
Intel
C. Pignataro, Ed.
Cisco
November 3, 2017

Network Service Header (NSH)
draft-ietf-sfc-nsh-28

Abstract

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

Status of This Memo

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

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

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

This Internet-Draft will expire on May 7, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Requirements Language	4
1.2. Applicability	4
1.3. Definition of Terms	5
1.4. Problem Space	6
1.5. NSH-based Service Chaining	6
2. Network Service Header	7
2.1. Network Service Header Format	7
2.2. NSH Base Header	8
2.3. Service Path Header	11
2.4. NSH MD Type 1	12
2.5. NSH MD Type 2	12
2.5.1. Optional Variable Length Metadata	13
3. NSH Actions	14
4. NSH Transport Encapsulation	16
5. Fragmentation Considerations	17
6. Service Path Forwarding with NSH	17
6.1. SFFs and Overlay Selection	17
6.2. Mapping the NSH to Network Topology	21
6.3. Service Plane Visibility	21
6.4. Service Graphs	22
7. Policy Enforcement with NSH	22
7.1. NSH Metadata and Policy Enforcement	22
7.2. Updating/Augmenting Metadata	24
7.3. Service Path Identifier and Metadata	25
8. Security Considerations	26
8.1. NSH Security Considerations from Operators' Environments	27
8.2. NSH Security Considerations from the SFC Architecture	28
8.2.1. Integrity	29
8.2.2. Confidentiality	31
9. Contributors	32
10. Acknowledgments	33
11. IANA Considerations	33
11.1. Network Service Header (NSH) Parameters	34
11.1.1. NSH Base Header Bits	34
11.1.2. NSH Version	34
11.1.3. MD Type Registry	34
11.1.4. MD Class Registry	35
11.1.5. New IETF Assigned Optional Variable Length Metadata Type Registry	36
11.1.6. NSH Base Header Next Protocol	36

12. NSH-Related Codepoints	37
12.1. NSH EtherType	37
13. References	37
13.1. Normative References	37
13.2. Informative References	38
Authors' Addresses	40

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, and so forth.

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

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

1. The movement of service functions and application workloads in the network.
2. The ability to easily bind service policy to granular information, such as per-subscriber state.
3. The capability to steer traffic to the requisite service function(s).

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

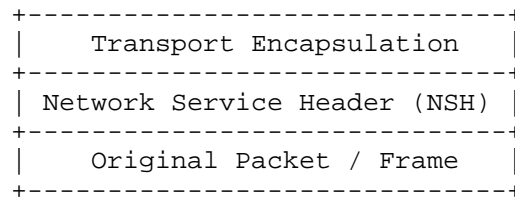


Figure 1: Network Service Header Encapsulation

The NSH is composed of the following elements:

1. Service Function Path identification.
2. Indication of location within a Service Function Path.
3. Optional, per packet metadata (fixed length or variable).

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

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

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

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

1.3. Definition of Terms

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

Classification: Defined in [RFC7665].

Classifier: Defined in [RFC7665].

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

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

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

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

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

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

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

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

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

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

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

SFC Proxy: Defined in [RFC7665].

1.4. Problem Space

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

1.5. NSH-based Service Chaining

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

1. Topological Independence: Service forwarding occurs within the service plane, so the underlying network topology does not require modification. The NSH provides an identifier used to select the network overlay for network forwarding.
2. Service Chaining: The NSH enables service chaining per [RFC7665]. The NSH contains path identification information needed to realize a service path. Furthermore, the NSH provides the ability to monitor and troubleshoot a service chain, end-to-end via service-specific 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. The 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. The NSH offers a common and standards-based header for service chaining to all network and service nodes.
5. Transport Encapsulation Agnostic: The NSH is transport encapsulation-independent, meaning it can be transported by a

variety of encapsulation protocols. An appropriate (for a given deployment) encapsulation protocol can be used to carry NSH-encapsulated traffic. This transport encapsulation may form an overlay network and if an existing overlay topology provides the required service path connectivity, that existing overlay may be used.

2. Network Service Header

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

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

2.1. Network Service Header Format

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

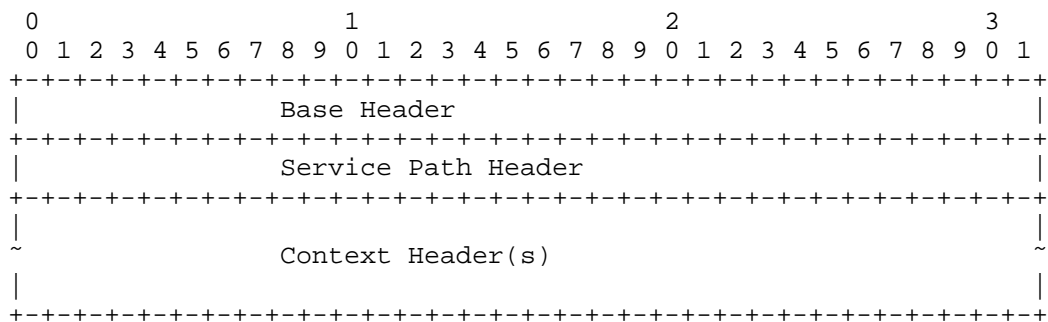


Figure 2: Network Service Header

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

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

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

2.2. NSH Base Header

Figure 3 depicts the NSH base header:

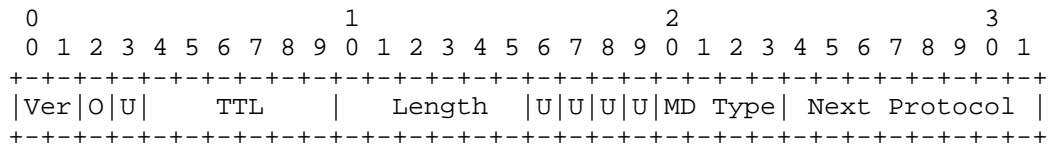


Figure 3: NSH Base Header

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 the NSH. If a packet presumed to carry an NSH header is received at an SFF, and the SFF does not understand the version of the protocol as indicated in the base header, the packet MUST be discarded, and the event SHOULD be logged. Given the widespread implementation of existing hardware that uses the first nibble after an MPLS label stack for equal-cost multipath (ECMP) decision processing, this document reserves version 01b. This value MUST NOT be used in future versions of the protocol. Please see [RFC7325] for further discussion of MPLS-related forwarding requirements.

O bit: Setting this bit indicates an Operations, Administration, and Maintenance (OAM, see [RFC6291]) 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. This field is used for service plane loop detection. The initial TTL value SHOULD be configurable via the control plane; the configured initial value can be specific to one or more SFPs. If no initial value is explicitly provided, the default initial TTL value of 63 MUST be used. Each SFF involved in forwarding an NSH packet MUST decrement the TTL value by 1 prior to NSH forwarding lookup. Decrementing by 1 from an incoming value of 0 shall result in a TTL value of 63. The packet MUST NOT be forwarded if TTL is, after decrement, 0.

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

Length: The total length, in 4-byte words, of the NSH including the Base Header, the Service Path Header, the Fixed Length Context Header or Variable Length Context Header(s). The length MUST be 0x6 for MD Type equal to 0x1, and MUST be 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.

Unassigned bits: All other flag fields, marked U, are unassigned and available for future use, see Section 11.1.1. Unassigned bits MUST be set to zero upon origination, and MUST be ignored and preserved unmodified by other NSH supporting elements. At reception, all elements MUST NOT modify their actions based on these unknown bits.

Metadata (MD) Type: Indicates the format of the NSH beyond the mandatory Base Header and the Service Path Header. MD Type defines the format of the metadata being carried. Please see the IANA Considerations Section 11.1.3.

This document specifies the following four MD Type values:

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

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

0x2 - This does not mandate any headers beyond the Base Header and Service Path Header, but may contain optional variable length Context Header(s). With MD Type 0x2, a Length of 0x2 implies there are no Context Headers. The semantics of the variable length Context

Header(s) are not defined in this document. The format of the optional variable length Context Headers is provided in Section 2.5.1.

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

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

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

Next Protocol: indicates the protocol type of the encapsulated data. The NSH does not alter the inner payload, and the semantics on the inner protocol remain unchanged due to NSH service function chaining. Please see the IANA Considerations section below, Section 11.1.6.

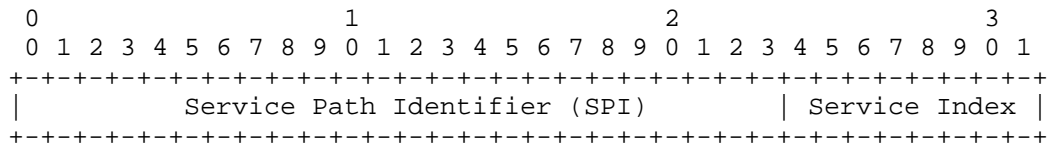
This document defines the following Next Protocol values:

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

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

2.3. Service Path Header

Figure 4 shows the format of the Service Path Header:



Service Path Identifier (SPI): 24 bits

Service Index (SI): 8 bits

Figure 4: NSH Service Path Header

The meaning of these fields is as follows:

Service Path Identifier (SPI): Uniquely identifies a service function path. Participating nodes **MUST** use this identifier for Service Function Path selection (SFP). 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). The 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 packet's NSH. The initial Classifier **MUST** send the packet to the first SFF in the identified SFP for forwarding along an SFP. If re-classification occurs, and that re-classification results in a new SPI, the (re)classifier is, in effect, the initial classifier for the resultant SPI.

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

2.4. NSH MD Type 1

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

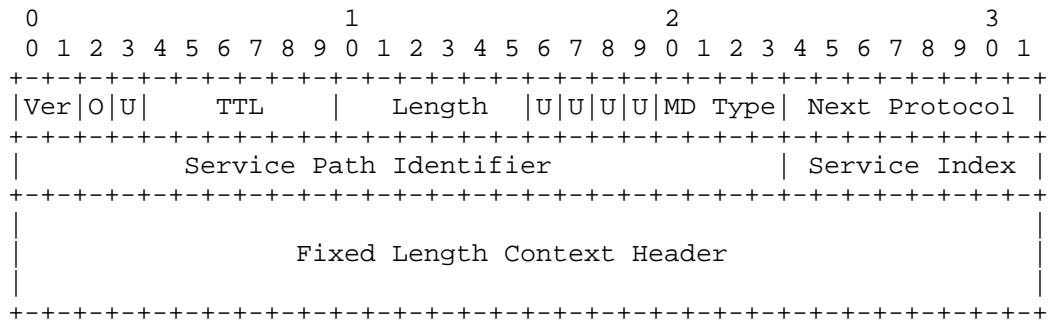


Figure 5: NSH MD Type=0x1

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

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

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

[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 6). 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.

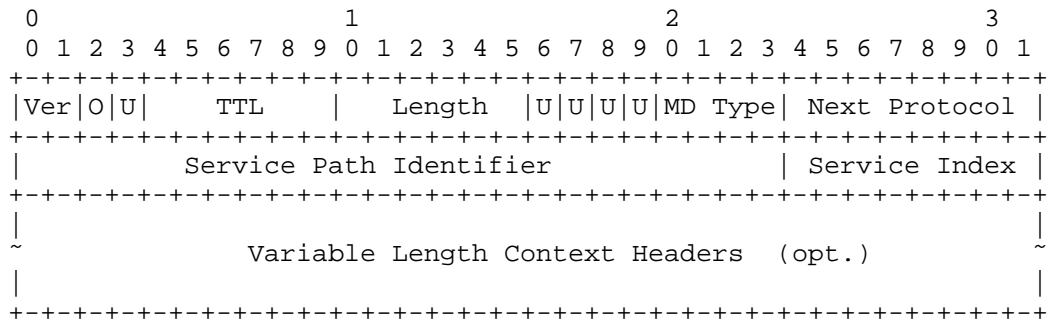


Figure 6: NSH MD Type=0x2

2.5.1. Optional Variable Length Metadata

The format of the optional variable length Context Headers, is as depicted in Figure 7.

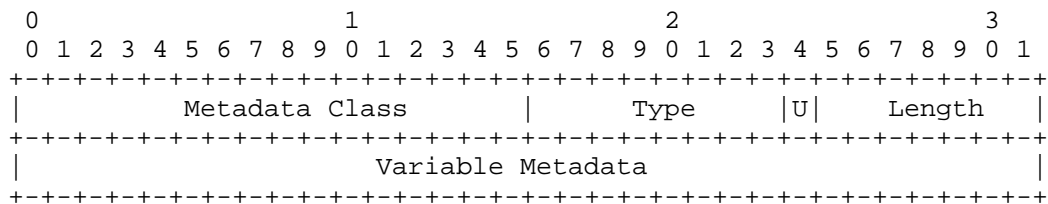


Figure 7: Variable Context Headers

Metadata Class (MD Class): Defines the scope of the 'Type' field to provide a hierarchical namespace. The IANA Considerations Section 11.1.4 defines how the MD Class values can be allocated to standards bodies, vendors, and others.

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

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

Length: Indicates the length of the variable metadata, in bytes. In case the metadata length is not an integer number of 4-byte words, the sender MUST add pad bytes immediately following the last metadata byte to extend the metadata to an integer number of 4-byte words. The receiver MUST round 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 bytes) and MUST ignore the remaining bytes up to the nearest 4-byte word boundary. The Length may be 0 or greater.

A value of 0 denotes a Context Header without a Variable 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 its scoping (see e.g., Section 3.3.3 of [I-D.ietf-sfc-control-plane]).

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

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

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

3. NSH Actions

NSH-aware nodes, which include service classifiers, SFFs, SFs and SFC proxies, may alter the contents of the NSH headers. These nodes have several possible NSH-related actions:

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

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

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

2. Select service path: The Service Path Header provides service path information and is used by SFFs to determine correct service path selection. SFFs MUST use the Service Path Header for selecting the next SF or SFF in the service path.
3. Update the NSH: SFs MUST decrement the service index by one. If an SFF receives a packet with an SPI and SI that do not correspond to a valid next hop in a valid 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 an 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 the NSH before forwarding it to an NSH-unaware SF. When the SFC Proxy receives a packet back from an NSH-unaware SF, it MUST re-encapsulate it with the correct NSH, and MUST decrement the Service Index by one.

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

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

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

Figure 8: NSH Action and Role Mapping

4. NSH Transport Encapsulation

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

1. Creates a topologically independent services plane. Packets are forwarded to the required services without changing the underlying network topology.
2. Transit network nodes simply forward the encapsulated packets without modification.

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

5. Fragmentation Considerations

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

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

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

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

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

6. Service Path Forwarding with NSH

6.1. SFFs and Overlay Selection

As described above, the NSH contains a Service Path Identifier (SPI) and a Service Index (SI). The SPI is, as per its name, an identifier. The SPI alone cannot be used to forward packets along a service path. Rather the SPI provides a level of indirection between the service path/topology and the network transport encapsulation.

Furthermore, there is no requirement, 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 paths. In particular, an SI value of zero indicates that forwarding is incorrect and the packet must be discarded.

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

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

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

Table 1: SFF NSH Mapping Example

Additionally, further indirection is possible: the resolution of the required SF network locator may be a localized resolution on an SFF, rather than 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] [RFC7676], respectively.

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

Table 2: NSH to SF Mapping Example

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

Table 3: SF Locator Mapping Example

Since the SPI is a representation of the service path, the lookup may return more than one possible next-hop within a service path for a given SF, essentially a series of weighted (equally or otherwise) paths to be used (for load distribution, redundancy, or policy), see Table 4. The metric depicted in Table 4 is an example to help 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

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

6.2. Mapping the NSH to Network Topology

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

Examples of mapping for a topology:

1. Next SF is located at SFFb with locator 2001:db8::1
SFFa mapping: SPI=10 --> VXLAN-gpe, dst-ip: 2001:db8::1
2. Next SF is located at SFFc with multiple network locators for load distribution purposes:
SFFb mapping: SPI=10 --> VXLAN-gpe, dst_ip:203.0.113.1, 203.0.113.2, 203.0.113.3, equal cost
3. Next SF is located at SFFd with two paths from SFFc, one for redundancy:
SFFc mapping: SPI=10 --> VXLAN-gpe, dst_ip:192.0.2.10 cost=10, 203.0.113.10, cost=20

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

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

6.3. Service Plane Visibility

The SPI and SI serve an important function for visibility into the service topology. An operator can determine what service path a packet is "on", and its location within that path simply by viewing NSH information (packet capture, 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, of course, also 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 2, NSH provides the ability to carry metadata along a service path. This metadata may be derived from several sources. Common examples include:

Network nodes/devices: Information provided by network nodes can indicate network-centric information (such as 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.

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

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

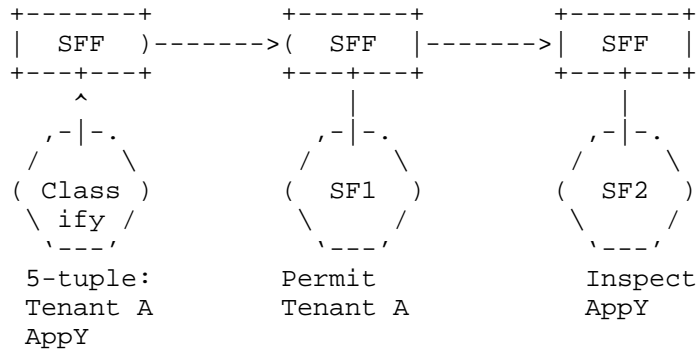


Figure 9: Metadata and Policy

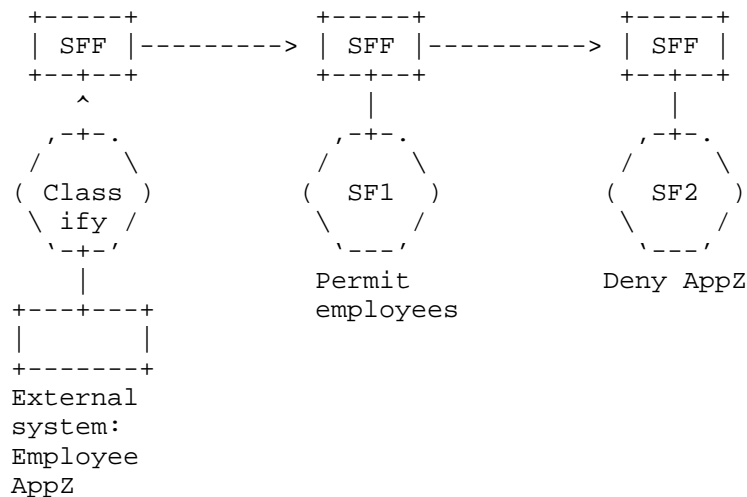


Figure 10: External Metadata and Policy

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

Depending on the information carried in the metadata, data privacy impact needs to be considered. For example, if the metadata conveys tenant information, that information may need to be authenticated

and/or encrypted between the originator and the intended recipients (which may include intended SFs only); one approach to an optional capability to do this is explored in [I-D.reddy-sfc-nsh-encrypt]. The NSH itself does not provide privacy functions, rather it relies on the transport encapsulation/overlay. An operator can select the appropriate set of transport encapsulation protocols to ensure confidentiality (and other security) considerations are met. Metadata privacy and security considerations are a matter for the documents that define metadata format.

7.2. Updating/Augmenting Metadata

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

1. Metadata Augmentation: Information may be added to the NSH's existing metadata, as depicted in Figure 11. For example, if the initial classification returns the tenant information, a secondary classification (perhaps co-resident with 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 12 illustrates an example of updating metadata.

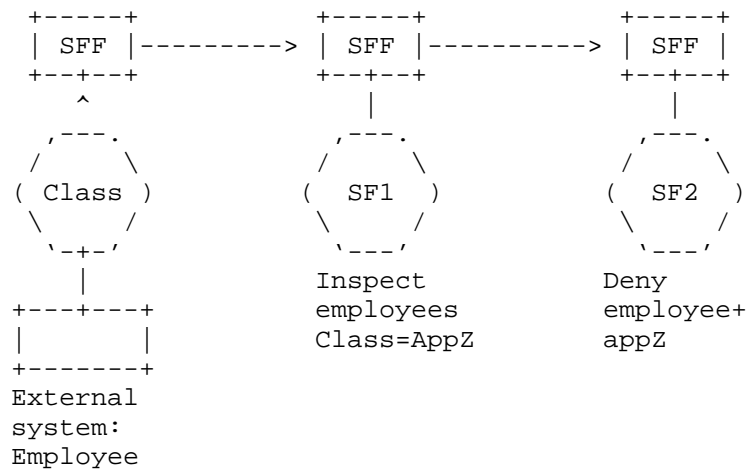


Figure 11: Metadata Augmentation

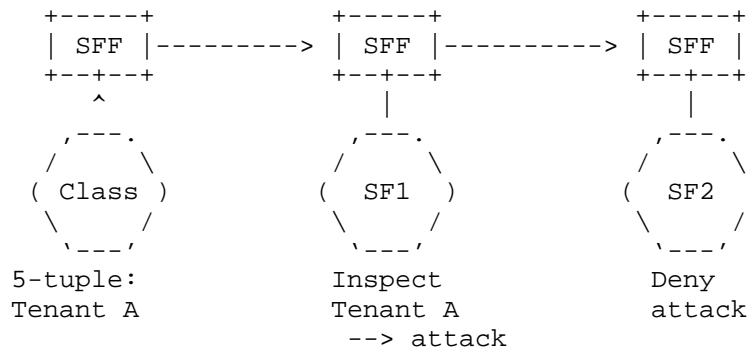
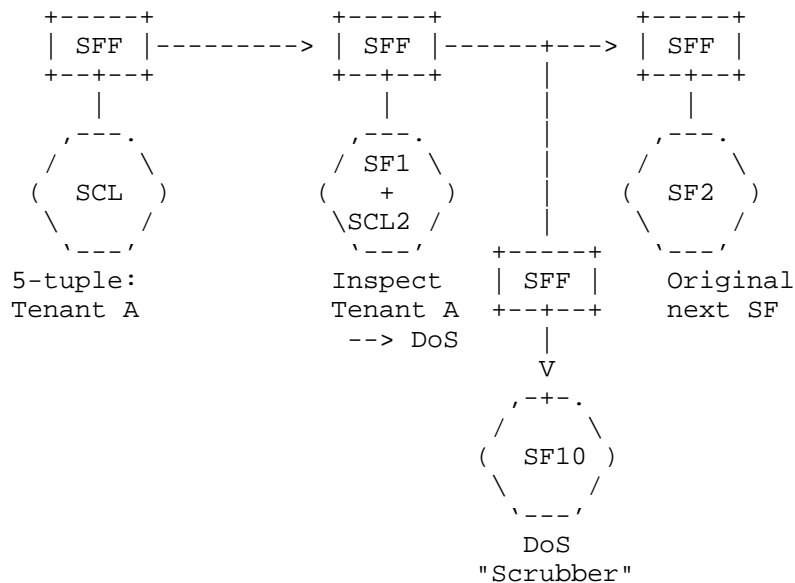


Figure 12: Metadata Update

7.3. Service Path Identifier and Metadata

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

This relationship provides the ability to create a dynamic service plane based on complex classification without requiring each node to be capable of such classification, or requiring a coupling to the network topology. This yields service graph functionality as



8.1. NSH Security Considerations from Operators' Environments

Trusted Devices

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

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

Single Domain Boundary

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

i) Clear NSH boundaries

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

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

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

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

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

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

ii) Mitigation of external threats

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

8.2. NSH Security Considerations from the SFC Architecture

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

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

8.2.1. Integrity

SFC devices

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

NSH Base and Service Path Headers

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

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

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

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

1. SFF/SF NSH verification

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

2. Transport Security

NSH is always encapsulated by an outer transport encapsulation as detailed in Section 4 of this specification, and as depicted in Figure 1. If an operator deems cryptographic integrity protection necessary due to their risk analysis,

then an outer transport encapsulation that provides such protection [RFC6071], such as IPsec, MUST be used.

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

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

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

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

3. NSH Variable Header-based Integrity

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

NSH metadata

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

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

8.2.2. Confidentiality

SFC devices

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

NSH base and service path headers

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

NSH metadata

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

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

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

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

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

For those situations where obfuscation is either inapplicable or judged to be insufficient, an operator can also encrypt the metadata. An approach to an optional capability to do this was explored in [I-D.reddy-sfc-nsh-encrypt]. For other situations where greater assurance is desired, optional mechanisms such as [I-D.brockners-proof-of-transit] can be used.

9. Contributors

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

- o Jim Guichard, Cisco Systems, Inc.
- o Surendra Kumar, Cisco Systems, Inc.
- o Michael Smith, Cisco Systems, Inc.
- o Wim Henderickx, Alcatel-Lucent
- o Tom Nadeau, Brocade
- o Puneet Agarwal
- o Rajeev Manur, Broadcom
- o Abhishek Chauhan, Citrix
- o Joel Halpern, Ericsson
- o Sumandra Majee, F5
- o David Melman, Marvell
- o Pankaj Garg, Microsoft
- o Brad McConnell, Rackspace
- o Chris Wright, Red Hat, Inc.
- o Kevin Glavin, Riverbed
- o Hong (Cathy) Zhang, Huawei US R&D

- o Louis Fourie, Huawei US R&D
- o Ron Parker, Affirmed Networks
- o Myo Zarny, Goldman Sachs
- o Andrew Dolganow, Alcatel-Lucent
- o Rex Fernando, Cisco Systems, Inc.
- o Praveen Muley, Alcatel-Lucent
- o Navindra Yadav, Cisco Systems, Inc.

10. Acknowledgments

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

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

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

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

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

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

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

11. IANA Considerations

11.1. 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.

11.1.1. NSH Base Header Bits

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

Bit 2 - O (OAM) bit

Bit 3 - Unassigned

Bits 16-19 - Unassigned

11.1.2. NSH Version

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

Version 00b: Protocol as defined by this document.

Version 01b: Reserved. This document.

Version 10b: Unassigned.

Version 11b: Unassigned.

11.1.3. MD Type Registry

IANA is requested to set up a registry of "MD Types". These are 4-bit values. MD Type values 0x0, 0x1, 0x2, and 0xF 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
0x0	Reserved	This document
0x1	NSH MD Type 1	This document
0x2	NSH MD Type 2	This document
0x3..0xE	Unassigned	
0xF	Experimentation	This document

Table 5: MD Type Values

11.1.4. 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

A registry for Types for the MD Class of 0x0000 is defined in Section 11.1.5.

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

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

11.1.5. New IETF Assigned Optional Variable Length Metadata Type Registry

The Type values within the IETF Base NSH MD Class, i.e., when the MD Class is set to 0x0000 (see Section 11.1.4), are the Types owned by the IETF. This document requests IANA to create a registry for the Type values for the IETF Base NSH MD Class called the "IETF Assigned Optional Variable Length Metadata Type Registry", as specified in Section 2.5.1.

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

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

11.1.6. 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	
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

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

12. NSH-Related Codepoints

12.1. NSH EtherType

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

13. References

13.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

- [RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <<https://www.rfc-editor.org/info/rfc7665>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.

13.2. Informative References

- [I-D.brockners-proof-of-transit]
Brockners, F., Bhandari, S., Dara, S., Pignataro, C., Leddy, J., Youell, S., Mozes, D., and T. Mizrahi, "Proof of Transit", draft-brockners-proof-of-transit-04 (work in progress), October 2017.
- [I-D.guichard-sfc-nsh-dc-allocation]
Guichard, J., Smith, M., Kumar, S., Majee, S., Agarwal, P., Glavin, K., Laribi, Y., and T. Mizrahi, "Network Service Header (NSH) MD Type 1: Context Header Allocation (Data Center)", draft-guichard-sfc-nsh-dc-allocation-07 (work in progress), August 2017.
- [I-D.ietf-nvo3-vxlan-gpe]
Maino, F., Kreeger, L., and U. Elzur, "Generic Protocol Extension for VXLAN", draft-ietf-nvo3-vxlan-gpe-05 (work in progress), October 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 (SFC) Operation, Administration and Maintenance (OAM) Framework", draft-ietf-sfc-oam-framework-03 (work in progress), September 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-03 (work in progress), July 2017.
- [I-D.reddy-sfc-nsh-encrypt]
Reddy, T., Patil, P., Fluhrer, S., and P. Quinn, "Authenticated and encrypted NSH service chains", draft-reddy-sfc-nsh-encrypt-00 (work in progress), April 2015.
- [RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, DOI 10.17487/RFC2784, March 2000, <<https://www.rfc-editor.org/info/rfc2784>>.
- [RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <<https://www.rfc-editor.org/info/rfc3552>>.
- [RFC3692] Narten, T., "Assigning Experimental and Testing Numbers Considered Useful", BCP 82, RFC 3692, DOI 10.17487/RFC3692, January 2004, <<https://www.rfc-editor.org/info/rfc3692>>.
- [RFC6071] Frankel, S. and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap", RFC 6071, DOI 10.17487/RFC6071, February 2011, <<https://www.rfc-editor.org/info/rfc6071>>.
- [RFC6291] Andersson, L., van Helvoort, H., Bonica, R., Romascanu, D., and S. Mansfield, "Guidelines for the Use of the "OAM" Acronym in the IETF", BCP 161, RFC 6291, DOI 10.17487/RFC6291, June 2011, <<https://www.rfc-editor.org/info/rfc6291>>.
- [RFC7325] Villamizar, C., Ed., Kompella, K., Amante, S., Malis, A., and C. Pignataro, "MPLS Forwarding Compliance and Performance Requirements", RFC 7325, DOI 10.17487/RFC7325, August 2014, <<https://www.rfc-editor.org/info/rfc7325>>.
- [RFC7498] Quinn, P., Ed. and T. Nadeau, Ed., "Problem Statement for Service Function Chaining", RFC 7498, DOI 10.17487/RFC7498, April 2015, <<https://www.rfc-editor.org/info/rfc7498>>.

- [RFC7676] Pignataro, C., Bonica, R., and S. Krishnan, "IPv6 Support for Generic Routing Encapsulation (GRE)", RFC 7676, DOI 10.17487/RFC7676, October 2015, <<https://www.rfc-editor.org/info/rfc7676>>.
- [RFC8165] Hardie, T., "Design Considerations for Metadata Insertion", RFC 8165, DOI 10.17487/RFC8165, May 2017, <<https://www.rfc-editor.org/info/rfc8165>>.
- [RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <<https://www.rfc-editor.org/info/rfc8201>>.

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

SFC Working Group
Internet-Draft
Intended status: Informational
Expires: May 1, 2017

D. Migault, Ed.
Ericsson
C. Pignataro
T. Reddy
Cisco
C. Inacio
CERT/SEI/CMU
October 28, 2016

SFC environment Security requirements
draft-mglt-sfc-security-environment-req-02.txt

Abstract

This document provides environment security requirements for the SFC architecture. Environment security requirements are independent of the protocols used for SFC - such as NSH for example. As a result, the requirements provided in this document are intended to provide good security practices so SFC can be securely deployed and operated. These security requirements are designated as environment security requirements as opposed to the protocol security requirements.

Status of This Memo

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

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

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

This Internet-Draft will expire on May 1, 2017.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Requirements notation	2
2. Introduction	2
3. Terminology and Acronyms	3
4. SFC Environment Overview	3
4.1. Deployment of SFC Architecture	6
5. Threat Analysis	7
5.1. Attacks performed from the SFC Control Plane	8
5.2. Attacks performed from the SFC Management Plane	9
5.3. Attacks performed from the Tenant's Users Plane	9
5.4. Attacks performed from the SFC Data Plane	11
6. Security Requirements	14
6.1. Plane Isolation Requirements	15
6.1.1. SFC Control Plane Isolation	16
6.1.2. SFC Management Plane Isolation	17
6.1.3. Tenant's Users Data Plane Isolation	18
6.2. SFC Data Plane Requirements	19
6.3. Additional Requirements	22
7. Security Considerations	22
8. Privacy Considerations	23
9. IANA Considerations	23
10. Acknowledgments	23
11. References	23
11.1. Normative References	23
11.2. Informative References	24
Authors' Addresses	24

1. Requirements notation

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 [RFC2119].

2. Introduction

This document provides environment security requirements for the SFC architecture [I-D.ietf-sfc-architecture]. Environment security requirements are independent of the protocols used for SFC - such as NSH [I-D.ietf-sfc-nsh]. As a result, the requirements provided in this document are intended to provide good security practice so SFC

can be securely deployed and operated. These security requirements are designated as environment security requirements as opposed to the protocol security requirements. This document is built as follows. Section 4 provides an overall description of the SFC environment with the introduction of the different planes (SFC Control Plane, the SFC Management Plane, the Tenant's user Plane and the SFC Data Plane). Section 6 lists environment security requirements for the SFC. These requirements are intended to prevent attacks, as well as network and SFC misconfigurations. When such events happens, the security recommendations also aim at detecting and identifying the threats or misconfiguration as well as limiting their impact. Recommendations also may apply differently depending on the infrastructure. For example trusted environment may enforce lighter security recommendations than public and open SFC infrastructures. However, one should also consider future evolution of their infrastructure, and consider the requirements as a way to maintain the SFC architecture stable during its complete life cycle. For each requirement this document attempts to provide further guidance on the reasons to enforce it as well as what should be considered while enforcing it or the associated risks of not enforcing it.

This document assumes the reader is familiar with the SFC architecture defined in [I-D.ietf-sfc-architecture] as well as the Internet Security Glossary [RFC4949]

3. Terminology and Acronyms

In addition to the terminology defined in [I-D.ietf-sfc-architecture], the document defines the following terminology:

- Tenant: A tenant is one organization that is using SFC. A tenant may use SFC on one's own private infrastructure or on a shared infrastructure.
- Tenant's User Data Plane: The tenant may be using SFC to provide service to its customers or users. The communication of these users is designated as Tenant's user Data Plane and includes all communications involving the tenant's users. As a result, if a user is communicating with a server or a user from another domain, the communication with that tenant's user is part of the Tenant's Users Data Plane.

4. SFC Environment Overview

This section provides an overview of SFC. It is not in the scope to this document to provide an explicit description of SFC. Instead, the reader is expected to read [RFC7498],

[I-D.ietf-sfc-architecture], [I-D.ietf-sfc-control-plane] and other SFC related documents.

Service Function Chaining (SFC) architecture is defined in [I-D.ietf-sfc-architecture]. This section briefly illustrates the main concepts of the SFC architecture and positions the architecture within an environment.

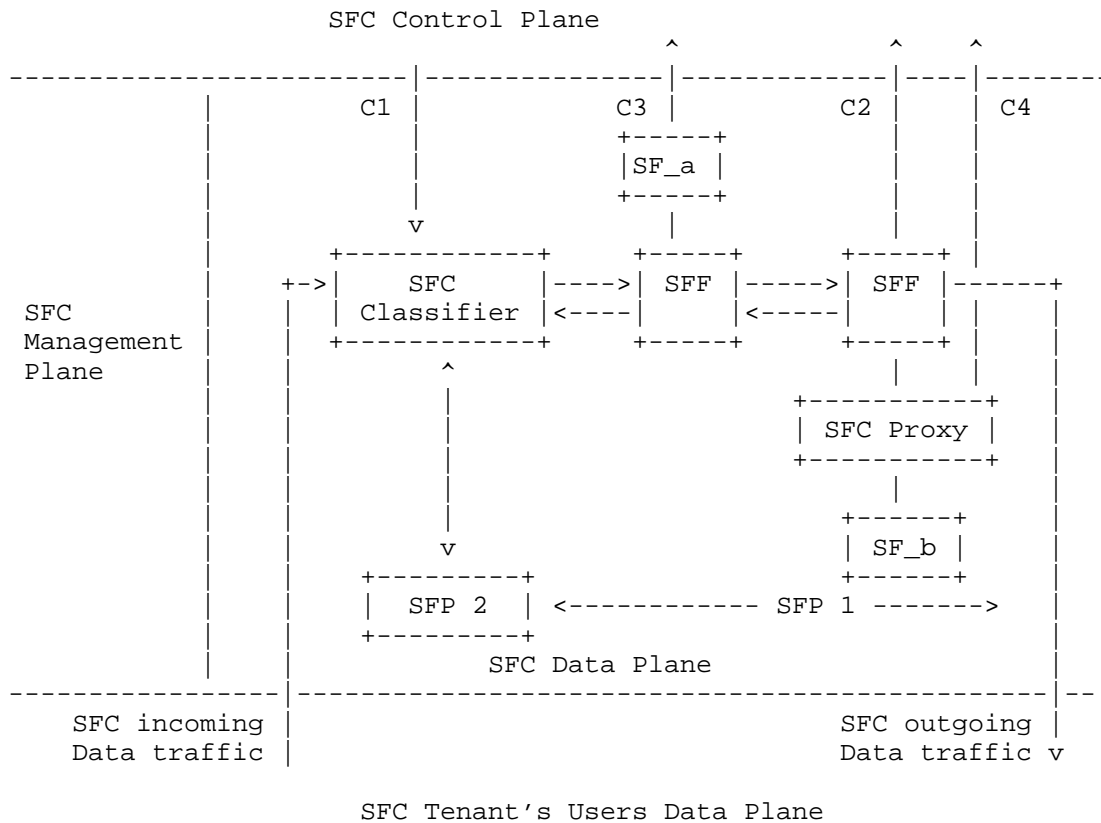


Figure 1: SFC Environment Overview

SFC defined a Service Function Path (SFP) which is an ordered set of Service Functions (SF) applied to part of the packets. The figure above represents two SFP: SFP1 and SFP2. SFP2 is not detailed but SFP1 defines a path that goes through SF_a and SF_b. SFP is defined at the SF level, which means the path does not consider the specific instance of an SF for example. A SF may be performed by different instances of SF located at different positions. As a result, a specific packet may pass through different instances of SFC. The

ordered set of SF instances a packet goes through is called the Rendered SF Path (RSFP).

Upon the receipt of an incoming packet from the tenant's user, the SFC Classifier determines, according to Classifiers, which SFP is associated to that packet. The packet is forwarded from Service Function Forwarders (SFF) to SFF. SFF are then in charge of forwarding the packet to the next SFF or to a SF. Forwarding decisions may be performed using SFP information provided by the SFC Encapsulation. As described in [I-D.ietf-sfc-nsh] the SFC Encapsulation contains SFP information such as the SFP ID and Service Index and eventually (especially for the MD-2 in NSH) some additional metadata. SF may be SFC aware or not. In the case the SFC functions are not SFC aware, a SFC Proxy performs the SFC Decapsulation (resp. SFC Encapsulation) before forwarding the packet to the SF (resp. after receiving the packet from the SF).

The environment associated to SFC may be separated into the four main planes:

- SFC Management Plane and Control Plane are defined in [I-D.ietf-sfc-control-plane]. The SFC Management Plane can be assimilated to the cloud infrastructure provider allocating various resource to the various SF and eventually active the various SF components. Typically management operations would consist in setting the number of CPU, memory bandwidth associated to the various SFs as well as specific configuration parameters of the SFC components. It is expected that the interface between the various SFC components configuration will be vendor specific. These configurations may be provided by the Cloud infrastructure provider or in the case of multitenancy by the administrator of the virtual network, or by each administrator of the SFC components. The SFC control plane controls and configure the SFC related components. The Control Plane differs from the Management Plane as it only concerns a subset of the parameters and facilities associated to the SF. In general, these parameters are expected to only modify the internal states of the different elements. This aspect confers programmability properties to the Control Plane that are usually not provide to the Management Plane. It is also expected that the SFP are elaborated in this plane before being pushed into the SFC Data Plane, and more generally, the SFP state in the SFF is expected to come from control rather than management.
- SFC Data Plane consists in all SF components as well as the data exchanged between the SF components. Communications between SF components includes the packet themselves, their

associated metadata, the routing logic - similar to RIB - or SF logic, i.e. what they returned values are for example. In other words, the SFC Data Plane can also be seen as all the elements that interact with a packet provided by an end user. Of course the end user is not expected to configure any of these element through the SFC Data Plane. Instead it is expected to apply the policies and configurations put in place by the SFC Tenant.

- SFC Tenant's Users Data Plane consists in the traffic data provided by the different users of the tenants. When a user is communicating with a server or another user -eventually from another administrative domain - , the communication belongs to the SFC Tenant's Users Data Plane whenever packets are provided by the server or by the user.

4.1. Deployment of SFC Architecture

This section illustrates a deployment of SFC we consider in this document.

A Cloud Provider provides an infrastructure that is shared by multiple SFC Tenants. The Cloud Provider may also provide some servers or hardware that have a dedicated function. Such hardware may be provided to the SFC Tenants under the form of a SF. It may thus be shared by multiple SFC Tenants. Such SF are designated as third party SF. Another case of SF may also consider a local SF proxying the traffic to a remote site or domain. The SF proxy transparently to the SFC elements may forward the traffic out of the boundaries of the Tenant. In some case this may be needed, but in some other case this may be done unbeknownst to the Tenant's.

Each SFC Tenant is responsible of its domain, that is to administrate or provision the necessary resource and control all its SFC elements which include defining SFC Paths, configuring the elements... Typically the coordination of the SFC elements is likely to be performed by a SDN controller.

Protecting the deployed SFC architecture from attacker is one goal of the security requirements. Some could easily argue that such requirements are not needed for example in a private SFC deployment where SFC components may be considered in a trusted environment and administrated by a single entity. However, even in a single administrative domain, inside attacks are possible. (e.g. inside attacker sniffing the SFC metadata, sending spoofed packets etc.). Then, the trusted domain assumption may not remain valid over time. Suppose, for example, that the SFC architecture is now interconnected with some third party SF or SFF. Such SFC component is now outside the initial trusted domain which has several security implications.

Similarly, a single trusted domain with one tenant may evolve over time and become multitenants and share a SFC platform. These tenants, may be trusted as in the case for example where each tenant represents a different department of a single company. Authentication is not sufficient, and relying only on a access control presents some risks. If the tenants are not strongly isolated - with physical or logical networks isolation, they may share a common SFF and one tenant may update the SFP of the other tenant. Such misconfiguration has similar impact as a redirecting attack. This document provide guidance that result in limiting such risks and improve detection for further mitigation.

5. Threat Analysis

The SFC environment is composed of the following plans: SFC Management Plane, SFC Control Plane, SFC Data Plane and SFC Tenant's User Data Plane. The purpose of these planes is to group a given set of functions while limiting the interactions between these planes. Interactions between planes are only limited - in most cases controlled - but these interactions still exist and so may be used by an attacker. As a result, for each plane, the threat analysis is performed by analysis the vulnerabilities present within each plane as well as those performed via the other planes.

Threat analysis of the Management Plane and the Control Plane have been described in [I-D.ietf-sfc-control-plane]. The SFC Tenant's User Plane is out of the boundaries of the SFC administrator. As a result attacks performed on SFC Tenant's User Plan are not considered in this section and this section limits its analysis on teh SFC Data Plan.

This section describes potential threats the SFC Data Plane may be exposed. The list of threats is not expected to be complete. More especially, the threats mentioned are provided to illustrate some security requirements for the SFC architecture. For simplicity, this document mostly considers that security breaches are performed by an attacker. However, such breaches may also be non-intentional and may result from misconfiguration for example.

Attacks may be performed from inside the SFC Data Plane or from outside the SFC Data plane, in which case, the attacker is in at least one of the following planes: SFC Control Plane, SFC Management Plane or SFC Tenants' Users Plane. Some most sophisticated attacks may involve a coordination of attackers in multiple planes.

5.1. Attacks performed from the SFC Control Plane

Attacks related to the control plane have been detailed in section 5 of [I-D.ietf-sfc-control-plane].

The different interfaces between the SFC Control Plane and the SFC Data Plane are exposed in [draft-ietf-sfc-control-plane]. It includes:

- Updating the classification rule of the SFC Classifier (also referred as interface C1).
- Updating the forwarding decision of the SFF (also referred as interface C2). This interface is also used to provide the SFC Control Plane some information for example on the system load, network load or the latency so appropriated SFP may be computed.
- Updating SF's internal states (interface C3). This interface is also used to provide the SFC Control Plane some information for example on the system load, network load or the latency so appropriated SFP may be computed.
- Updating SFC Proxy's internal states (interface C4). This interface is also used to provide the SFC Control Plane some information for example on the system load, network load or the latency so appropriated SFP may be computed.

An attacker may change the SFC Classifier classification and completely modify the services provided by the SFC. Such privileges may be used to avoid some control over the tenant's traffic (like firewalling service). An attacker may also modify the filtering or classification rules to overload heavy processing functions with traffic. In a pay-what-you-use model, this could result in extra cost for the tenant or to trigger a DoS attack on the tenant SFC Data Plane.

Attack performed on the SFC Control Plane mostly consists in tenant impersonation or communication hijacking. This would enable an attacker to control the SFC components associated to the tenant. Similarly an attacker may also collect system or network load information in order to better orchestrate a DoS attack for example. An attacker may also inject instructions in order to perform a DoS attack on a given SFC component or to prevent the tenant to control other SFC components.

5.2. Attacks performed from the SFC Management Plane

Attacks performed on the SFC Management Plane are similar to those performed from the SFC Control Plane. The main difference is that the SFC Management Plan provides usually a greater control of the SFC component than the SFC Control Plane.

In addition, the actions performed by the SFC Management Plane have fewer restrictions, which means it may be harder to enforce strong control access policies.

5.3. Attacks performed from the Tenant's Users Plane

The SFC Tenant's User Plane is not expected to have fine access control policies on the packets sent or received by users. Unless they are filtered, all packets are good candidate to the SFC Classifier. This provides the user some opportunities to test the behavior of the SFC.

In addition, the Tenant's Users Plane is not controlled by the SFC Tenant, and users may initiate communications where both ends - the client and the server- are under the control of the same user. Such communications may be seen as user controlled communications (UCC).

UCC may enable any user to monitor and measure the health of the SFC. This may be an useful information to infer information on the tenant's activity or to define when a DoS attack may cause more damage. One way to measure the health or load of the tenant's SFC is to regularly send a packet and measure the time it takes to be received, in order to estimate the processing time within the SFC.

UCC may enable any user to test the consistency of the SFC. One example of inconsistency could be that SFC decapsulation is not performed - or inconsistently performed - before leaving the SFC, which could leak some metadata with private information. For example, a user may send spoofed packet. Suppose for example, that a request HTTP GET video.example.com/movie is received with some extra header information such as CLIENT_ID: 1234567890, or CLIENT_EMAIL: client@example.foo. If these pieces of information are derived from the source IP address, the attacker may collect them by changing the IP address for example. In this case, the spoofed packets as used to collect private and confidential information of the tenant's users. Note that such threat is not specific to SFC, and results from the combination of spoofed IP and non-authenticated IP address are used to identify a user. What is specific to SFC is that metadata are likely to carry multiple pieces of information - potentially non-authenticated - associated to the user. In the case above, meta-data is carried over the HTTP header. Inserting the metadata in the HTTP

header may be performed by a SF that takes its input from the SFC encapsulation. In addition, SFC encapsulation may also leak this information directly to a malicious node if that node belongs to the SFC plane. In this later case, the user builds on the top of and intrusion to the SFC Data Plane that is detailed later.

In some case, spoofed packet may impersonate other's tenants. Suppose for example that the same infrastructure is used by multi tenants, and which are identified by the IP address of their users. In this case, spoofing an IP address associated to another tenant may be sufficient to collect the information confidential and private information. The best current practice to prevent such leaks are usually ingress filtering for example, which prevents illegitimate flows to enter the network. Note that ingress filtering may also be performed at higher layers such as at application layers to prevent unexpected applications to enter the network. When possible, the cost needs to be balanced with the risk by the SFC tenants.

Similarly, UCC may enable any user to infer packet has been dropped or is in a loop. Suppose a user send a spoofed packet and receives no response. The attacker may infer that the packet has been dropped or is in a loop. A loop is expect to load the system and sending a "well known packet" over the UCC and measuring the response time may determine whether the packet has been dropped or is in a loop.

Correlation of time measurement and spoofed packet over a UCC may provide various type information that could be used by an attacker.

- The attacker may correlate spoofed packet and time measurement in order discover the SFC topology or the logic of the SFC Classifier. Typically, it may infer when new SFs are placed in the SFC for example. In addition, as metadata are placed in band, the time response may also provide an indication of the size of the metadata associated to the packet. The combination of these pieces of information may help an attacker to orchestrate a future attack on a specific SF either to maximize the damages or to collect some metadata - like identification credentials.
- The attacker may also define the type of packets that require the SFC the more processing. Additional processing may be due a large set of additional metadata that require fragmentation, some packets that are not treated in a coherent and consistent manner within the SFC. Such information may be used for example to optimize a DoS attack. In addition, it could also be used in order to artificially increase the necessary resource of the Tenant in order to increase the cost of operation for running its service.

Time measurement and spoofed packet in combination with variable query rate over a UCC may provide information on the orchestration of the SFC itself. For example, the user may be able to detect when elasticity mechanisms are triggered. Such attack is not SFC specific, and may have occurred with traditional cloud mechanisms. However, the main difference between SFC and traditional cloud mechanisms is that SFC is a standard way to interconnect SF. In that sense, the use of SFC provides more details to the attack as non standard mechanisms.

An attacker may be able to leverage the knowledge that SFC is in use by specific carriers to effect the processing of data using the SFC system as a processor in the attack. This leads to a number of potential weaknesses in the Internet ecosystem.

An attacker may be able to characterize the type of client platforms using a web site by carefully crafting data streams that will be modified by the SFC system versus client systems that would view web data unmodified. For example, leveraging SFC and carefully crafted data, a malicious web site operator may be able to create a particularly formatted common file that when modified by a cellular operator for bandwidth savings creates a file that may crash, (creating a DoS attack) on a select set of clients. Clients not accessing that web site using the same RSFP would not experience any issues. Additionally, external examination of the malicious site would not demonstrate any malicious content, relying on the SF to modify the content.

A well crafted site could potentially leverage the variances of functionality from different RSFPs in order to GEO locate a user. An example would be creating an image file which when recompressed creates image artifacts rendering the image unusable, but allowing the user to respond to such an event, thereby letting the web site operator know the user has potentially moved from a higher to lower bandwidth network location within the area of a specific network operator.

5.4. Attacks performed from the SFC Data Plane

This section considers an attacker has been able to take control of an SFC component. As a result, the attacker may become able to modify the traffic and perform, on-path attacks, it may also be able to generate traffic, or redirect traffic to perform some kind of Man-in-the-middle attacks. This is clearly a fault, and security policies should be set to avoid this situation. This section analyses in case this intrusion occurs, the potential consequences on the SFC. As mentioned earlier, this section assumes all these actions are performed by an attacker. However, what is designated by

an attack may also result from misconfigurations at various layer. A SF or a SFF may become inadvertently configured or programmed which may result in similar outcomes as an attack. Whatever result in what we designate as an attack, the purpose of security requirements will be to detect, to analyse and mitigate such security breaches.

The traffic within the SFC Data Plane is composed of multiple layers. The traffic is composed of communications between SFC components. The transport between the SFC component is the transport protocol and is not considered in the SFC. It can typically be a L2 transport layer, or an L3 transport layer using various encapsulation techniques (vLAN, VxLAN, GRE, IPsec tunnels for example). Each of these transport layer adds or remove attack vectors. The transport layer carries SFC Encapsulated that are composed of an SFC Encapsulation envelope that carries metadata and a SFC payload that is the actual packet exchanged between the two end points.

As a result, attacker may use the traffic to perform attacks at various layers. More specifically, attacks may be performed at the transport layer, the SFC Encapsulation layer or the SFC payload layer.

- Attacks performed at the transport layer may be related to SFC in the sense that illegitimate SFC traffic could be provided to the SF. Typically, a malicious node that is not expected to communicate with that SF may inject packets into the SFC, such malicious node may eventually spoof the IP address of legitimate SF, so the receiving SF may not be able to detect the packet is not legitimate. Threats related to IP spoofing are described in [RFC6959] and may be addressed by authenticated traffic (e.g. using IPsec). Such threats are not related to SFC even though they may impact a given SF.
- the SFC Encapsulation as well as the SFC payload are usually considered as input by a SF. As such they may represent efficient vector of attacks for the SF. Attacks performed through SFC payload are similar as the ones described in the Tenant's Users Data Plane section. As a result, such attacks are not considered in this section, and this section mostly considers attacks based on the SFC Encapsulation and malicious metadata.

When an attacker is within the SFC Data Plane, it may have a full or partial control of one SF component in which case, the attacker is likely to compromise the associated SFCs. It could for example, modify the expected operation of the SFC. Note that in this case, the SFC may be appropriately provisioned and set, however, the SFC

does not operate as expected this may only be detected by monitoring and auditing the SFC Data Plane.

Although traffic authentication may be performed at various layers L2 L3 or at the SFC Encapsulation layer, this section considers the SFC traffic. As a result, the SFC traffic is authenticated if the SF is able to authenticate the incoming SFC packet.

When SFC traffic is not authenticated, an attacker may inject spoofed packet in any SFC component. The attacker may use spoofed packet to discover the logic of the SFC. On the other hand, the attacker may also inject packet in order to perform DoS attack via reflection. In fact, as NSH provides the ability to add metadata, some deployment may end up with payloads carrying large metadata. Addition of such overhead presents a vector for amplification within the SFC Data Plane and thus either load the network or the next SF. Note that amplification may be generated by metadata, the SFC payload, and the attacker may replay packets or completely craft new packets. In addition, the attacker may choose a spoofed packet to increase the CPU load on the SFC components. For example, it could insert additional metadata to generate fragmentation. Similarly, it may also insert unnecessary metadata that may need to be decapsulated and analyzed even though they may not be considered for further actions. Spoofed packet may not only be generated to attack the SFC component at the SFC layer. In fact spoofed packet may also target applications of the SF. For example an attacker may also forge packet for HTTP based application - like a L7 firewall - in order to perform a slowloris [SLOWLORIS] like attack. Note that in this case, such attacks are addressed in the Tenant's Users Data Plane section. The specificity here is that the attacker has a more advanced understanding of the processing of the SFC, and can thus be more efficient.

When SFC traffic is not authenticated, an attacker may also modify on-path the packet. By changing some metadata contained in the SFC Encapsulation, the attacker may test and discover the logic of the SFF. Similarly, when the attacker is aware of the logic of a SFC component, the attacker may modify some metadata in order to modify the expected operation of the SFC. Such example includes for example redirection to a SF which could result in overloading the SF and overall affect the complete SFC. Similarly, the attacker may also create loops within the SFC. Note that redirection may not occur only in a given SFC. In fact, the attacker may use SFC branching to affect other SFC. Another example would also include a redirection to a node owned by the attacker and which is completely outside the SFC. Motivation for such redirection would be that the attacker has full administrator privileges on that node, whereas it only has limited capabilities on the corrupted node. Such attack is a man-in-

the-middle attack. The important thing to note is that in this case the traffic is brought outside the legitimate SFC domain. In fact, performing a man-in-the-middle attack as described above means that the SFC domain has been extended. This can be easily performed in case all node of the data center or the tenant's virtual network is likely to host a SFC component. A similar scenario may also consider that the traffic could be redirected outside the data center or the tenant's virtual network if the routing of firewall rule enables such policies.

A direct consequence is that a corrupted SFC component may affect the whole SFC. This also means that the trust of a given SFC decreases with the number of SF involved as each SF presents a surface of attack.

An attacker may also perform passive attacks by listening to traffic exchanged throughout the SFC Data Plane. Such attacks are described in [RFC7258]. Metadata are associated to each packet. These metadata are additional pieces of information not carried in the packet and necessary for each SF to operate. As a result, metadata may contain private information such as identifiers or credentials. In addition, observing the traffic may provide information on the tenant's activity. Note that encryption only may not prevent such attacks, as activity may be inferred by the traffic load.

6. Security Requirements

This section aims at providing environment security requirements. These requirements are derived from the generalization of the threat analysis described in Section 5. More specifically, the threat analysis section was mostly illustrative, and its generalization leads us to the following requirements.

Although the security requirements are derived from described threats, the scope of security should be understood in a much broader way than addressing threats. In fact the primary purpose of the security requirements is to ensure the deployment of the SFC architecture can remain robust and stable.

The goal of this section is to provide some security requirements that should be checked against any evolution of the deployment of SFC architecture. The requirements should be understood and the risks of not following them should be evaluated with the current deployment as well as the foreseen evolutions.

Similarly, the document provides means to evaluate the consequences of a security breach, as well as means to detect them.

The motivations for the security requirements are:

- a) Preventing attacks
- b) Preventing misconfigurations - as far as stability and security of the SFC architecture is concerned.
- c) Providing means to evaluate the consequences of a security breach
- d) Making possible to audit, and detect any misbehavior that may affect stability and security of the SFC.

6.1. Plane Isolation Requirements

Plane Isolation consists in limiting the surface of attack of the SFC Data Plane by controlling the interfaces between the SFC Data Plane and the other planes.

Complete isolation of the planes is not possible, as there are still some communications that must be enabled in order to benefit from the benefits of SFC. Typically the SFC Control Plane configures the SFC elements used by the SFC Data Plane. Similarly, access to the SFC Control Plane may be performed remotely, in which case interaction between the SFC Tenant's User and the SFC Control Plane may be considered. As a result, isolation should be understood as enabling communications between planes in a controlled way.

This section lists the recommendations so communication between planes can be controlled. This involves controlling communications between planes as well as controlling communication within a plane.

The requirements listed below applies to all planes, whereas the following subsection are more specific to each plane, providing recommendations on the interface with the SFC Data Plane.

REQ1: In order to increase isolation every plane that communicates with another plane SHOULD use a dedicated interface. In our case, the SFC Management Plane, the SFC Control Plane and the SFC Data Plane SHOULD use dedicated networks and dedicated interfaces. Isolation of inter-plane communication may be enforced using different ways. How isolation is enforced depends on the type of traffic, the network environment for example, and within a given SFC architecture different techniques may be used for the different planes. One way to isolate communications is to use completely different network on dedicated NICS. On the other hand, depending on the required level of isolation, a logical isolation may be performed using different IP addresses or ports with network

logically isolated - that is using for example different VxLAN, or GRE tunnels. In this case, isolation relies on the trust associated to the different switches and router. In case of a lack of trust on the on-path elements, authenticated encryption may be used to provide a logical isolation. With authenticated encryption, trust is placed on the end points. Note also that encryption can also be used in combination of other isolation mechanisms in order to increase the level of isolation.

REQ2: Activity between planes SHOULD be monitored and regularly audited. At least operations performed between the planes as well as the source and destination should be logged. When possible the identity of the identities should also be logged. Activity may be performed independently by the different planes as well as by different actors such as the SFC Tenants, the infrastructure provider. The level of information available may also differ between planes and actors.

REQ3: Traffic and communications between planes SHOULD be filtered traffic or rate-limited. Filtering and rate-limiting policies may be finer grained and may apply for a subset of traffic.

The above requirements mostly corresponds to the architecture best current practice. Isolation is mostly motivated to avoid the planes to interact on each other. For example the load on the SFC Data Plane should not affect the SFC Control Plane and SFC Management Plane communications. Such requirements are also current best practices.

Such recommendations are thus strongly recommended even in the case the two planes are considered to belong to trusted environments.

6.1.1.1. SFC Control Plane Isolation

In order to limit the risks of an attack from the SFC Control Plane, effort should be made in order to restrict the capabilities and the information provided by the SFC Data Plane to the SFC Control Plane to the authorized tenants only. In this case the authorized tenants are the users or organizations responsible for the SFC domain.

REQ4: Tenants of the SFC Control Plane SHOULD authenticate in order to prevent tenant's usurpation or communication hijacking.

REQ5: Communications between SFC Control Plane and the SFC Data Plane MUST be authenticated and encrypted in order to preserve privacy. The purpose of encryption in this case prevents an attacker to be aware of the action performed by the SFC

Control Plane. Such information may be used to orchestrate an attack - especially when SFC component report their CPU/network load.

REQ6: Strong access control policies SHOULD be enforced. Control SHOULD be performed on the engaged resource (e.g. CPU, memory, disk access for example) and SHOULD be associated explicitly to authorized tenants. By default, a tenant SHOULD be denied any access to resource, and access SHOULD be explicit.

Given the SFC Control Plane traffic load that is expected to be light - at least compared to the SFC Tenant's Users Data Plane or the SFC Data Plane. As a result, encryption is not expected to impact the performances of the SFC architecture. Given the effort to migrate from an non authenticated (and non protected) communications to a protected communication, we recommend these requirements to be considered even in trusted environments. By protecting these communications by design, the deployed SFC architecture is also ready for future expansion of the Control Plane outside the initial trusted domain. This could typically include the evolution to multiple tenants as well as the inclusion of tenants that remotely access the SFC Control Plane.

Access Control policies can be enforced in various ways. One way could be to consider the systems of the SF to limit the resources associated to each tenants. Other ways include the use of API in order to limit the scope of possible interactions between the SFC Control Plane and the SFC Data Plane. This is one way to limit the possibilities of the tenants. In addition, each of these actions should be associated an authorized tenant, as well as authorized parameters. The use of API belongs to best practices and so is strongly recommended even in trusted environments.

REQ7: Audit SHOULD be performed regularly to check access control policies are still up-to-date and prevent non-authorized users to control the SFC Data Plane.

The purpose of audits is to provide evidences when something went wrong. As a result, audit facilities are expected to be provided even in trusted environments.

6.1.2. SFC Management Plane Isolation

The requirements for the SFC Control Plane and SFC Management Plane are similar. The main difference of the interfaces between the SFC Management Plane and the SFC Control Plane is that it is less likely that APIs could be used to configure the different SFC components.

As a result, users of the SFC Management Plane are likely to have a broader and wider control over the SFC component.

REQ8: it is RECOMMENDED to enforce stronger authentication mechanisms (for example relying on hardware tokens or keys) and to limit the scope of administrative roles on a per component basis.

REQ9: SFC Control Plane and SFC Management Plane may present some overlap. Each SFC component MUST have clear policies in case these two planes enter in conflict.

6.1.3. Tenant's Users Data Plane Isolation

The Tenant's Users Data Plane is supposed to have less restricted access control than the other SFC Management Plane and SFC Control Planes. A typical use case could be that each tenant are controlling and managing the SFC in order to provide services to their associated users. The number of users interacting with the SFC Data Plane is expected to be larger than the number of tenants interacting with the SFC Control and SFC Management Planes. In addition, the scope of communications initiated or terminating at the user end points is likely to be unlimited compared to the scope of communications between the tenants and the SFC Control Plane or SFC Management Plane. In such cases, the tenant may be provided two roles. One to grant access to the SFC, and another one to control and manage the SFC. These two roles should be able to interact and communicate.

REQ10: Users SHOULD be authenticated, and only being granted access to the SFC if authorized. Authorization may be provided by the SFC itself or outside the SFC.

REQ11: Filtering policies SHOULD prevent access to a user, or traffic when a malicious behavior is noticed. A malicious activity may be noticed once a given behavioral pattern is detected or when unexpected load is monitored in the SFC Data Plane.

REQ12: Tenant's User Plane SHOULD be monitored, in order to detect malicious behaviors.

REQ13: When SFC is used by multiple tenants, each tenant's traffic SHOULD be isolated based on authenticated information. More specifically, the use of a Classifier that can easily be spoofed like an IP address SHOULD NOT be used.

REQ14: It is RECOMMENDED that user's access authorization be performed outside the SFC. In fact granting access and treating the traffic are two different functions, and we

RECOMMEND they remain separated. Then, splitting these two functions makes it possible for a tester to perform tests of an potential attacker, without any contextual information. More specifically, having a traffic identified as associated to test by the SFC reduces the scope of the tests simply because an attacker will not be considered as a tester. For that reason, we RECOMMEND authorization is performed outside the SFC, and SFC deployment may not be designed to authenticate end users.

The remaining requirements are associated to monitoring the network and providing interactions between the access and the SFC. This interaction may be provided outside SFC itself.

6.2. SFC Data Plane Requirements

This section provides requirements and recommendation for the SFC Data Plane.

- REQ15: Communications within the SFC Data Plane SHOULD be authenticated in order to prevent the traffic to be modified or injected by an attacker. As a result, authentication includes the SFC Encapsulation as well as the SFC payload.
- REQ16: Communication MUST NOT reveal privacy sensitive metadata.
- REQ17: The metadata provided in the communication MUST be limited in in term of volume as to limit the amplification factor as well as fragmentation.
- REQ18: Metadata SHOULD NOT be considered by the SFF for forwarding decision. In fact, the inputs considered for switching the packet to the next SFF or a SF should involve a minimum processing operation to be read. More specifically, these inputs are expected fixed length value fields in the SFC Encapsulation header rather than any TLV format.
- REQ19: When multiple tenants share a given infrastructure, the traffic associated to each tenant MUST be authenticated and respective Tenant's Users Planes MUST remain isolated. More specifically, if for example, a SFC Classifier is shared between multiple tenants. The Classifier used to associate the SFC MUST be authenticated. This is to limit the use of spoofed Classifiers. In any case, the SFC component that receives traffic from multiple tenants is assumed to be trusted.

REQ20: Being a member of a SFC domain SHOULD be explicitly mentioned by the node and means should be provided so the SFC domain the node belongs to may be checked. Such requirement intends to prevent a packet to go outside a SFC domain, for example in the case of a man-in-the-middle attacks, where a redirection occurs outside the SFC domain. It is expected that most deployment will rely on border / port mechanisms that prevent outsider users from injecting packets with spoofed metadata. Although such mechanisms are strongly recommended to deploy, in case of failure, they do not prevent man-in-the-middle attack outside the SFC domain.

Authentication of the traffic within the SFC Data Plane is particularly recommended in an open environment where third party SF or SFF are involved. It can also be recommended when a strong isolation of the traffic is crucial for the infrastructure or to meet some level of certification. In addition, authentication may also be performed using various techniques. The whole packet may be authenticated or limited to some parts (like the flow ID). Authenticating the traffic and how or what to authenticate is a trade off between the risk associated and the cost of encryption. When possible we recommend to authenticate, but we also consider that the price may be too high in controlled and small trusted environment.

Metadata is an important part of the SFC architecture, and their impact on security should be closely evaluated. It is the responsibility of the SFC administrator to evaluate the privacy associated by the metadata - section 5.2.2 of [RFC6973] - and according to this evaluation to consider appropriated mechanisms to prevent the privacy leakage. Mechanisms should be provided even though they may not be activated.

As a general guidance exposing privacy sensitive metadata in any communications between two any SFC component should be avoided. [One way, for example to avoid exposing privacy sensitive metadata is to include a reference to the metadata instead of the metadata itself. Another way could be to encrypt the metadata itself - but that is part of the solution space.] Applying this principle prevents any private oriented data to be leaked. This requirement is mandatory when the SFC is not deployed in a trusted environment.

When exposition of the privacy sensitive metadata cannot be avoided and you are in a trusted domain, then exposing privacy sensitive metadata may be considered as long as they do not leak outside the boundaries of the trusted environment. In this case, the security is delegated to the security policies of the trusted environment boundaries, that may be outside the scope of SFC. More especially, the security policies may be for example enforced by a firewall. In

this specific case, the trusted environment MUST prevent leakage of the metadata out of the trusted environment and MUST ensure that untrusted node cannot access in any way the communications within the trusted environment.

The reason this requirement is set to MUST is to specify that if one does not follow the requirement it is at its own risk and must provide the necessary means to prevent any leak - in our case enforcing the necessary security policies that your environment / deployment needs.

Similarly, it is the responsibility of the administrator to define what an acceptable size for metadata is. Even in trusted environment, we recommend the SFC administrator be able to define and change this level.

Processing metadata by the SFF seems also expensive, and it is the responsibility of the SFC administrator to evaluate whether processing metadata by the SFF may impact the SFC architecture. In addition, metadata are expected to be associated to SF as opposed to the forwarding information that are associated to the SFF. These inputs have different functions, are associated to different processing rules, and may be administrated by different parties. It is thus part of the general good practise to split these functionalities. Optimization may require to combine the analysis of metadata and forwarding information, but this should be handled cautiously.

Assertion of belonging to a security domain, is especially recommended in open environments. This may also partly be addressed by node authenticating.

In addition, the following operational requirements have been identified:

REQ21: SFC components SHOULD be uniquely identified and have their own cryptographic material. In other words the use of a shared secret for all nodes SHOULD NOT be considered as one corrupted node would be able to impersonate any node of the SFC Data Plane. This is especially useful for audit.

REQ22: Activity in the SFC Data Plane MUST be monitored and audited regularly. Audit and log analysis is especially useful to check that SFC architecture assessments. They can be useful to detect a security breach for example before it is being discovered and exploited by a malicious user. Monitoring the system is also complementary in order to provide alarms when a suspicious activity is detected. Monitoring enables the

system to react to unexpected behaviors in a dynamic way. Both activities are complementary as monitoring enables to counter suspicious behavior and audit may detect misconfiguration or deep causes of a malicious behavior. For these reasons, audit and monitoring facilities are expected even in trusted environment.

REQ23: Isolate the Plane with border and firewall to restrict access of SFC components to legitimate traffic. More specifically, SFC components are supposed to be accessed only via dedicated interfaces. Outside these interfaces, inbound or outbound traffic SHOULD be rejected.

6.3. Additional Requirements

REQ24: SFC Encapsulation SHOULD carry some identification so it can be associated to the appropriated SFP as well as its position within the SFC or SFP. Indicating the SFP ID may be sufficient as long as a SFP can uniquely be associated to a single SFC. Otherwise, the SFC should be also somehow indicated. This is especially useful for audit and to avoid traffic coming from one SFC to mix with another SFC. Authentication of the SFP ID is one way to enforce SFP ID uniqueness. This may not be mandatory, but large deployment or deployment that are involving multiple parties are expected enforce this. In fact assuming SFP ID will have no collision is an hypothesis that may be hard to fulfill over time.

REQ25: Although this requirement is implementation specific, it is RECOMMENDED that SFF and SF keep separate roles. SFF should be focused on SF forwarding. As a result, they are expected to access a limited information from the packet - mostly fixed size information. SF on the other hand are service oriented, and are likely to access all SFC information which includes metadata for example. The reasons to keep these functions are clearly different and may involve different entities. For example, SF management or SF configuration may involve different administrators as those orchestrating the SFC.

REQ26: SFC Encapsulation SHOULD be integrity protected to prevent attackers from modifying the SFP ID. See Data Plane communication Requirements and considerations)

7. Security Considerations

8. Privacy Considerations

9. IANA Considerations

10. Acknowledgments

The authors would like to thank Joel Halpern, Mohamed Boucadair and Linda Dunbar for their valuable comments. Similarly the authors would also like to thank Martin Stiernerling for its careful review as well as its recommendations.

11. References

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007, <<http://www.rfc-editor.org/info/rfc4949>>.
- [RFC6959] McPherson, D., Baker, F., and J. Halpern, "Source Address Validation Improvement (SAVI) Threat Scope", RFC 6959, DOI 10.17487/RFC6959, May 2013, <<http://www.rfc-editor.org/info/rfc6959>>.
- [RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <<http://www.rfc-editor.org/info/rfc6973>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<http://www.rfc-editor.org/info/rfc7258>>.
- [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>>.

11.2. Informative References

[I-D.ietf-sfc-nsh]

Quinn, P. and U. Elzur, "Network Service Header", draft-ietf-sfc-nsh-01 (work in progress), July 2015.

[I-D.ietf-sfc-architecture]

Halpern, J. and C. Pignataro, "Service Function Chaining (SFC) Architecture", draft-ietf-sfc-architecture-11 (work in progress), July 2015.

[I-D.ietf-sfc-control-plane]

Li, H., Wu, Q., Huang, O., Boucadair, M., Jacquenet, C., Haeffner, W., Lee, S., Parker, R., Dunbar, L., Malis, A., Halpern, J., Reddy, T., and P. Patil, "Service Function Chaining (SFC) Control Plane Components & Requirements", draft-ietf-sfc-control-plane-00 (work in progress), August 2015.

[SLOWLORIS]

Wikipedia, "Slowloris", <https://en.wikipedia.org/wiki/Slowloris_%28software%29>.

Authors' Addresses

Daniel Migault (editor)
Ericsson
8400 boulevard Decarie
Montreal, QC H4P 2N2
Canada

Phone: +1 514-452-2160
Email: daniel.migault@ericsson.com

Carlos Pignataro
Cisco Systems, Inc.
7200-12 Kit Creek Road
Research Triangle Park, NC 27709
USA

Phone: +1 919-392-7428
Email: cpignata@cisco.com

Tirumaleswar Reddy
Cisco Systems, Inc.
Cessna Business Park, Varthur Hobli
Bangalore, Karnataka 560103
India

Phone: +91 9886
Email: tireddy@cisco.com

Christopher Inacio
CERT, Software Engineering Institute, Carnegie Mellon University
4500 5th Ave
Pittsburgh, PA 15213
USA

Phone: +1 412-268-3098
Email: inacio@cert.org