

SFC  
Internet-Draft  
Intended status: Standards Track  
Expires: May 7, 2020

M. Boucadair  
Orange  
T. Reddy  
McAfee  
November 04, 2019

Integrity Protection for Network Service Header (NSH) and Encryption of  
Sensitive Metadata  
[draft-rebo-sfc-nsh-integrity-01](#)

Abstract

This specification adds integrity protection and optional encryption directly to Network Service Headers (NSH) used for Service Function Chaining (SFC).

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, 2020.

Copyright Notice

Copyright (c) 2019 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

<a href="#">1.</a>	Introduction . . . . .	<a href="#">2</a>
<a href="#">2.</a>	Terminology . . . . .	<a href="#">3</a>
<a href="#">3.</a>	Assumptions & Basic Requirements . . . . .	<a href="#">4</a>
<a href="#">4.</a>	Solution Overview . . . . .	<a href="#">6</a>
<a href="#">5.</a>	Mandatory to Implement AEAD Algorithms . . . . .	<a href="#">8</a>
<a href="#">6.</a>	New NSH Variable-Length Context Headers . . . . .	<a href="#">8</a>
<a href="#">6.1.</a>	Key Identifier Context Header . . . . .	<a href="#">9</a>
<a href="#">6.2.</a>	Sequence Number Context Header . . . . .	<a href="#">9</a>
<a href="#">6.3.</a>	MAC and Encrypted Metadata Context Header . . . . .	<a href="#">10</a>
<a href="#">6.3.1.</a>	MAC#1 Context Header . . . . .	<a href="#">11</a>
<a href="#">6.3.2.</a>	MAC#2 Context Header . . . . .	<a href="#">12</a>
<a href="#">7.</a>	Processing Rules . . . . .	<a href="#">13</a>
<a href="#">7.1.</a>	Generic Behavior . . . . .	<a href="#">13</a>
<a href="#">7.2.</a>	MAC NSH Data Generation . . . . .	<a href="#">14</a>
<a href="#">7.3.</a>	Encrypted NSH Metadata Generation . . . . .	<a href="#">14</a>
<a href="#">7.4.</a>	Sequence Number Validation for Replay Attack . . . . .	<a href="#">15</a>
<a href="#">7.5.</a>	NSH Data Validation . . . . .	<a href="#">15</a>
<a href="#">7.6.</a>	Decryption of NSH Metadata . . . . .	<a href="#">16</a>
<a href="#">8.</a>	Security Considerations . . . . .	<a href="#">16</a>
<a href="#">8.1.</a>	MAC#1 . . . . .	<a href="#">17</a>
<a href="#">8.2.</a>	MAC#2 . . . . .	<a href="#">17</a>
<a href="#">9.</a>	IANA Considerations . . . . .	<a href="#">17</a>
<a href="#">10.</a>	Acknowledgements . . . . .	<a href="#">18</a>
<a href="#">11.</a>	References . . . . .	<a href="#">18</a>
<a href="#">11.1.</a>	Normative References . . . . .	<a href="#">18</a>
<a href="#">11.2.</a>	Informative References . . . . .	<a href="#">19</a>
<a href="#">Appendix A.</a>	A Deployment Example with KMS . . . . .	<a href="#">19</a>
	Authors' Addresses . . . . .	<a href="#">21</a>

**[1.](#) Introduction**

Many advanced Service Functions (e.g., Performance Enhancement Proxies, NATs, firewalls, etc.) are invoked for the delivery of value-added services, particularly to meet various service objectives such as IP address sharing, avoiding covert channels, detecting Denial-of-Service (DoS) attacks and protecting network infrastructures against them, network slicing, etc. Because of the proliferation of such advanced SFs together with complex service deployment constraints that demand more agile service delivery procedures, operators need to rationalize their service delivery logics and master their complexity while optimising service activation time cycles. The overall problem space is described in [\[RFC7498\]](#).

[\[RFC7665\]](#) presents an architecture addressing the problematic aspects of existing service deployments, including topological dependence and



configuration complexity. It also describes an architecture for the specification, creation, and maintenance of Service Function Chains (SFC) within a network. That is, how to define an ordered set of SFs and ordering constraints that must be applied to packets/flows selected as a result of traffic classification. [\[RFC8300\]](#) specifies the SFC encapsulation: Network Service Header (NSH).

NSH data is unauthenticated and unencrypted [\[RFC8300\]](#), forcing a service topology that requires security and privacy to use a transport encapsulation that supports such features (e.g., IPsec). The lack of such capability was reported during the development of [\[RFC8300\]](#) and [\[RFC8459\]](#).

This specification fills that gap. Concretely, this document adds integrity protection and optional encryption directly to NSH ([Section 4](#)). Thus, NSH data do not have to rely upon an underlying transport encapsulation for security and confidentiality. Note that the payload encapsulated by NSH is not part of the NSH data.

This specification introduces new Variable-Length Context Headers to carry fields necessary for integrity protected and encrypted NSH data ([Section 6](#)), and is therefore only applicable to NSH MD Type 0x02, as defined in [Section 2.5 of \[RFC8300\]](#).

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14 \[RFC2119\]](#) [\[RFC8174\]](#) when, and only when, they appear in all capitals, as shown here.

This document makes use of the terms defined in [\[RFC7665\]](#) and [\[RFC8300\]](#).

The document defines the following terms:

- o SFC data plane functional element: Refers to SFC-aware Service Function, Service Function Forwarder (SFF), SFC proxy, or classifier as defined in the SFC data plane architecture [\[RFC7665\]](#).
- o SFC Control Element: A logical entity that instructs one or more SFC data plane functional elements on how to process NSH packets within an SFC-enabled domain.
- o Key Identifier (or Ticket): A key identifier or kerberos-like object used to identify and deliver keys to authorized entities.



- o NSH imposer: Refers to the SFC data plane element that is entitled to impose NSH with the Context headers defined in this document.

### 3. Assumptions & Basic Requirements

The NSH format is defined in [Section 2 of \[RFC8300\]](#); the NSH data can be spread over three headers:

- o Base Header: Provides information about the service header and the payload protocol.
- o Service Path Header: Provides path identification and location within a service path.
- o Context Header: Carries metadata (i.e., context data) along a service path.

NSH allows to share context information (a.k.a., metadata) with upstream SFC-aware data elements on a per SFC/SFP basis. To that aim:

- o The control plane is used to instruct the SFC classifier about the set of context information to be supplied in the context of a given chain.
- o The control plane is also used to instruct an SFC-aware SF about any metadata it needs to attach to packets for a given SFC. This instruction may occur any time during the validity lifetime of an SFC/SFP. The control plane may indicate, for a given service function chain, an order for consuming a set of contexts supplied in a packet.
- o An SFC-aware SF can also be instructed about the behavior it should adopt after consuming a context information that was supplied in the NSH header. For example, the context can be maintained, updated, or stripped.
- o An SFC proxy may be instructed about the behavior it should adopt to process the context information that was supplied in the NSH header on behalf of an SFC-unaware SF, e.g., the context can be maintained or stripped.
- o The SFC proxy may also be instructed to add some new context information into the NSH header on behalf of an SFC-unaware SF.

In reference to Figure 1,



- o Classifiers, SFC-aware SFs, and SFC proxies are entitled to update the context header: Only these elements must be able to encrypt and decrypt a supplied context header.
- o Only SFC-aware SFs and SFC proxies are entitled to update the Service Path header. The solution must provide integrity protection for this header.
- o SFFs are entitled to modify the Base Path header (TTL value, for example). The solution may provide integrity protection for the base header.

Discussion Note: Check the level of this requirement with the WG.

	Insert, remove, or replace the NSH			Update the NSH	
SFC Data Plane Element	Insert	Remove	Replace	Dec. Service Index	Update Context Header
Classifier	+		+		+
Service Function Forwarder (SFF)		+			
Service Function (SF)				+	+
SFC Proxy	+	+		+	+

Figure 1: NSH Actions

The solution described in this document does not make any assumption about the service function chains to be instantiated nor adds any constraint about how NSH can be used within a domain. For example, in reference to Figure 2, the solution accommodates deployment schemes such as:

- o No metadata is inserted by the Classifier: it only proceeds with integrity protection.





- o SF1 inserts two metadata M1 and M2 that it encrypts.
- o SF2 decrypts M1 and M2, strips M2, and then encrypts M1
- o SF3 decrypts M1 and then strips it.

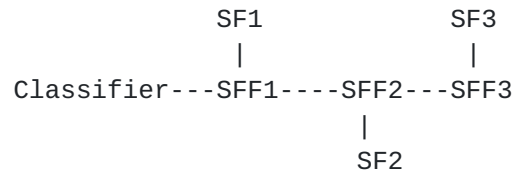


Figure 2: SFC-enabled Domain Example

#### 4. Solution Overview

The solution specified in the document allows for the following functions:

- o Encrypt all or a subset of metadata by Classifiers, SFC-aware SFs, and SFC proxies:

The control plane is assumed to instruct the Classifier, SFC-aware SFs, and SFC proxy with the set of context headers (privacy-sensitive metadata, typically) that must be encrypted.

The control plane may also indicate the set of SFC data plane elements that are entitled to supply a given context header (e.g., in reference to their identifiers as assigned within the SFC-enabled domain). It is out of the scope of this document to elaborate on how such instructions are provided to the appropriate SFC data plane elements, nor to detail the structure used to store the instructions.

The Service Path Header is not encrypted because SFFs use Service Index (SI) in conjunction with Service Path Identifier (SPI) for determining the next SF in the path.

- o Provide integrity protection for NSH data:

Two flavors are supported:

1. A first flavor where only the Service Path and Context Headers are integrity protected.

In this case, the NSH imposer may be a Classifier, an SFC-aware SF, or an SFC proxy.



2. A second flavor where the Base, Service Path, and Context Headers are integrity protected.

In this case, the NSH imposer may be a Classifier, an SFC-aware SF, an SFF, or an SFC proxy.

In order to avoid the overhead of multiple authentication tags and multiple keys, and to prevent SFFs from acquiring the secret key to decrypt the metadata, the recommendation is not to integrity protect the base header. Such approach does not require to recompute the MAC each time TTL is decremented by an SFF. As a reminder, an SFF is not supposed to act on the metadata or look into the content of the metadata.

The integrity scope is explicitly signaled in the NSH by means of a dedicated MD Type ([Section 6.3](#)).

In both schemes, the unencrypted metadata is subject to integrity protection.

The Authenticated Encryption with Associated Data (AEAD) algorithm [[RFC5116](#)] is used to provide NSH data integrity and to encrypt privacy-sensitive metadata.

The AEAD algorithm to be used by SFC data plane elements may be controlled using the control plane or other means. Mandatory to implement AEAD algorithms are listed in [Section 5](#).

AEAD algorithms take as input a single key (K), a Nonce, a plaintext (P) (which contains the data to be encrypted and authenticated), and "additional data" (A) (which contains the data to be authenticated, but not encrypted) as described in [Section 2.1 of \[RFC5116\]](#).

AEAD functions provide a unified encryption and authentication operation which turns plaintext into authenticated ciphertext and vice versa. When the length of plaintext is zero, the AEAD algorithm acts as a Message Authentication Code (MAC) on the "additional data" input. The length of the AEAD output will generally be larger than the plaintext, but by a magnitude that varies with the AEAD algorithm.

In order to decrypt and verify, the cipher takes as input the key (K), Nonce, additional data, and the ciphertext. The output is either the plaintext or an error indicating that the decryption failed as described in [Section 2.2 of \[RFC5116\]](#).

The procedure for the allocation/provisioning of the secret key (K) and AEAD algorithm is outside the scope of this specification. As



such, this specification does not mandate the support of any specific mechanism.

In order to accommodate deployments relying upon keying material per SFC/SFP and also the need to update keys after encrypting given amount of NSH data, this document uses key identifier (kid) to unambiguously identify the appropriate keying material. Doing so allows to address the problem of synchronization of keying material. A (non-normative) sample deployment case to illustrate how kids are assigned is provided in [Appendix A](#).

The NSH data is protected using K and optionally metadata is encrypted using K. New NSH Variable-Length Context Headers are defined in [Section 6](#) for NSH integrity protection and, optionally, metadata encryption. Concretely, an NSH imposer includes (1) the key identifier in NSH using the Key Identifier Context Header ([Section 6.1](#)), (2) a Sequence Number in Sequence Number Context Header to protect against replay attacks ([Section 7.4](#)), and (3) the Message Authentication Code (MAC) for the NSH data (depending on the integrity protection scope) calculated using the secret key (K) and optionally metadata encrypted K in 'MAC and Encrypted Metadata Context' Header ([Section 6.3](#)).

An NSH data plane element that needs to check the integrity of the NSH data uses the secret key (K) and AEAD algorithm for the key identifier being carried in the NSH.

An SFC-aware SF or SFC proxy that needs to decrypt the metadata uses the secret key (K) and the decryption algorithm for the key identifier being carried in the NSH.

[Section 7](#) specifies the detailed procedure.

## **5. Mandatory to Implement AEAD Algorithms**

Classifiers, SFC-aware SFs, and SFC proxies MUST implement the TLS\_AES\_128\_GCM\_SHA256 [[GCM](#)] cipher suite and SHOULD implement the TLS\_AES\_256\_GCM\_SHA384 [[GCM](#)] and TLS\_CHACHA20\_POLY1305\_SHA256 [[RFC8439](#)] cipher suites.

SFFs MAY implement the aforementioned cipher suites.

## **6. New NSH Variable-Length Context Headers**

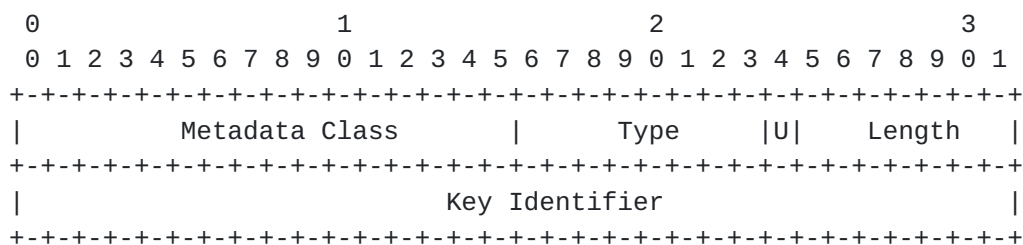
This section specifies the format of new Variable-Length Context headers that are used for NSH integrity protection and, optionally, metadata encryption.



### 6.1. Key Identifier Context Header

Key Identifier Context Header is a variable length Key Identifier object used to identify and deliver keys to SFC data plane elements. This is a mandatory TLV that MUST be present if an integrity protected and encrypted NSH solution is desired.

This Context Header is helpful to accommodate deployments relying upon keying material per SFC/SFP. Also, the key needs to be updated after the encryption of a certain amount of NSH data. The key identifier helps in resolving the problem of synchronization of keying material.



The description of the fields is as follows:

- o Metadata Class: MUST be set to 0x0 [[RFC8300](#)].
- o Type: TBD1 (See [Section 9](#))
- o Length: Variable.
- o Key Identifier: Carries the key identifier.

### 6.2. Sequence Number Context Header

Sequence Number Context Header conveys a 64-bit sequence number per key identifier. In this specification, a sequence number needs to be incremented every time NSH is included by the NSH imposer (for a given SFC/SFP). The sequence number SHOULD NOT be incremented if an existing NSH is being updated.

This is a mandatory TLV that MUST be present if an integrity protected and encrypted NSH solution is desired.









### 6.3.1. MAC#1 Context Header

MAC#1 Context Header is a variable-length TLV that carries the Message Authentication Code (MAC) for the Service Path and unencrypted Context headers calculated using key K and optionally metadata encrypted K. The scope of this TLV is depicted in Figure 5.

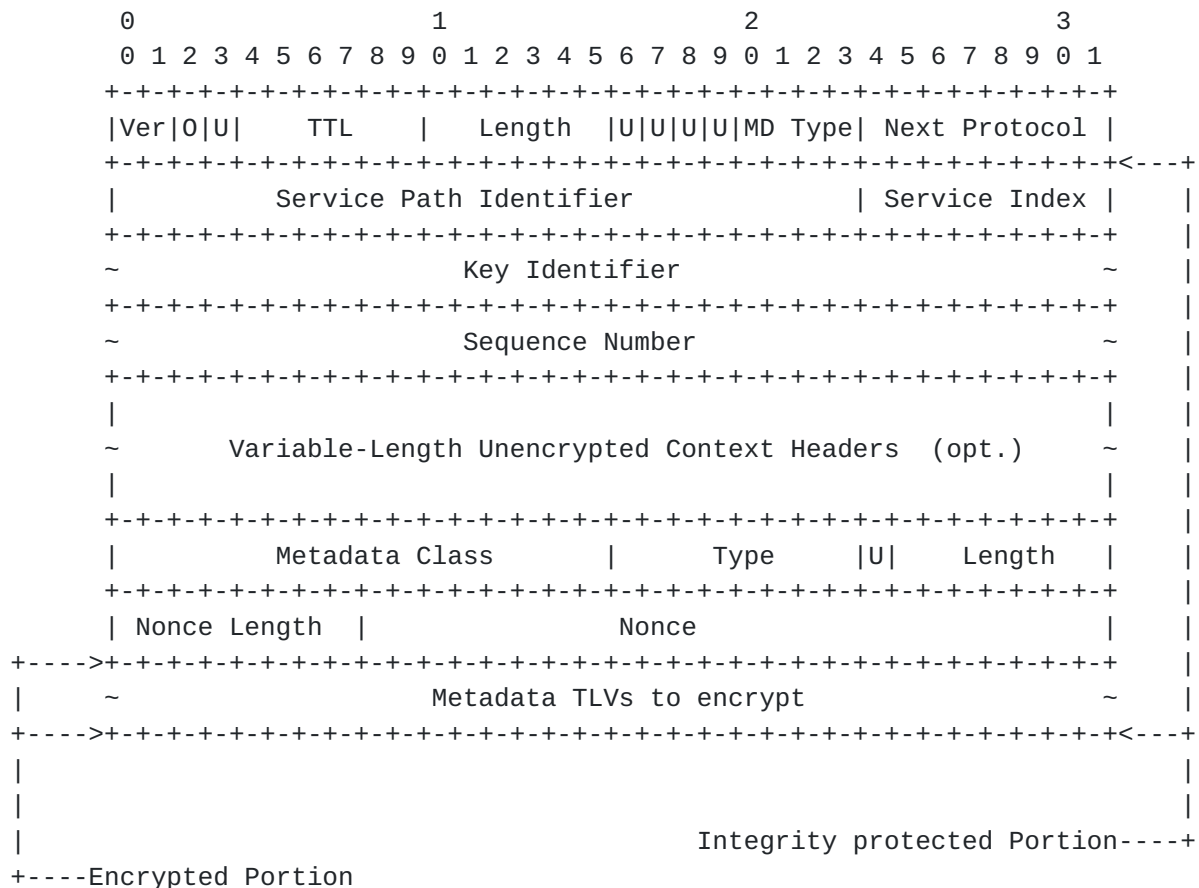


Figure 4: Scope of MAC#

In reference to Figure 3, the description of the fields is as follows:

- o Metadata Class: MUST be set to 0x0 [[RFC8300](#)].
- o Type: TBD3 (See [Section 9](#))
- o Nonce Length: Carries the length of the nonce ([Section 4 of \[RFC5116\]](#)).
- o Nonce: Carries the nonce for AEAD algorithms as discussed in [Section 3 of \[RFC5116\]](#). The associated data (defined in







- o Type: TBD4 (See [Section 9](#))
- o Nonce Length: Carries the length of the nonce ([Section 4 of \[RFC5116\]](#)).
- o Nonce: Carries the nonce for AEAD algorithms as discussed in [Section 3 of \[RFC5116\]](#). The associated data (defined in [\[RFC5116\]](#) as A) MUST be the entire NSH data excluding the metadata to be encrypted.
- o Message Authentication Code and optional Encrypted Metadata covering the entire NSH data excluding the metadata to be encrypted.

## **7. Processing Rules**

The following sub-sections describe the processing rules for integrity protected NSH and optionally encrypted metadata.

### **7.1. Generic Behavior**

This document adheres to the recommendations in [\[RFC8300\]](#) for handling the context headers at both ingress and egress SFC boundary nodes. That is, to strip such context headers.

Failures to inject or validate the Context Headers defined in this document SHOULD be logged locally while a notification alarm MAY be sent to an SFC Control Element. The details of sending notification alarms (i.e., the parameters affecting the transmission of the notification alarms depend on the information in the context header such as frequency, thresholds, and content in the alarm) SHOULD be configurable by the control plane.

SFC-aware SFs and SFC proxies MAY be instructed to strip some encrypted context headers from the packet or to pass the data to the next SF in the service function chain after processing the content of the context headers. If no instruction is provided, the default behavior for intermediary SFC-aware nodes is to maintain such context headers so that the information can be passed to next SFC-aware hops.

An SFC-aware SF or SFC proxy that receive an encrypted metadata, for which it is not allowed to decrypt the data, SHOULD keep that data unaltered when forwarding the packet upstream.

Notes: (1) add more text to handle multiple instances of the TLVs,  
(2) check which actual SFC element is doing what, ...





## 7.2. MAC NSH Data Generation

When the length of encrypted metadata is zero, the AEAD algorithm acts as a Message Authentication Code on the input A (defined in [RFC5116]). An NSH imposer inserts a "MAC and Encrypted Metadata" Context Header for integrity protection (Section 6.3).

The NSH imposer computes the message integrity for the target NSH data (depending on the integrity protection scope discussed in Section 6.3) using K, Nonce, and AEAD algorithm. It inserts the MAC in the "MAC and Encrypted Metadata" Context Header. The length of the MAC is decided by the AEAD algorithm adopted for the particular key identifier.

An entity in the service function path that intends to update NSH MUST follow the above behavior to maintain message integrity of the NSH for subsequent validations.

## 7.3. Encrypted NSH Metadata Generation

An NSH imposer can encrypt all NSH metadata or only a subset of metadata, i.e., encrypted and unencrypted metadata may be carried simultaneously (Figure 6).

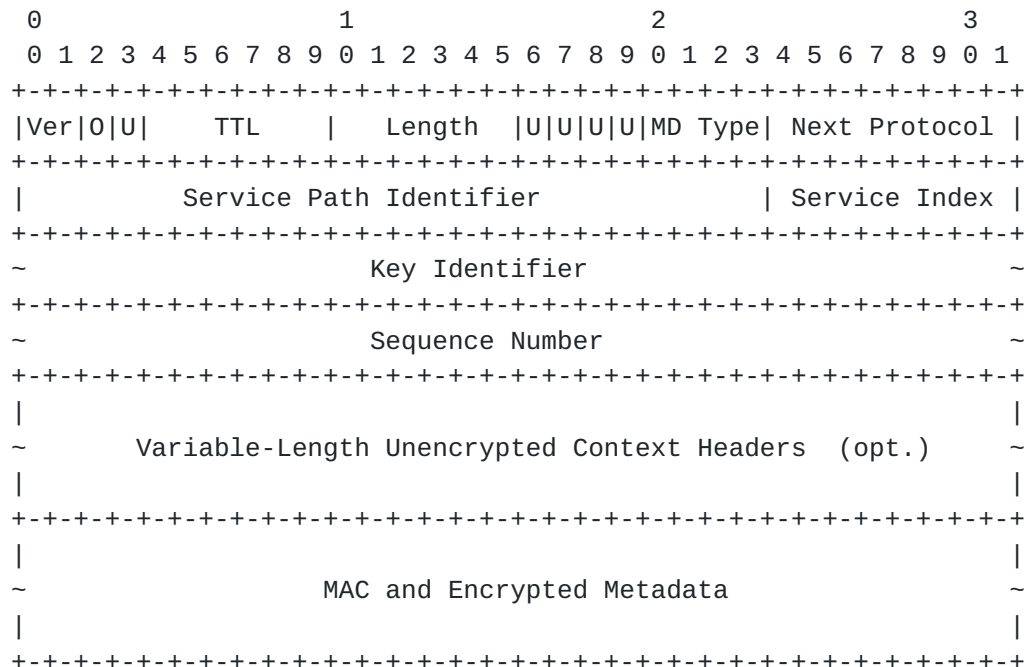


Figure 6: NSH with Encrypted and Unencrypted Metadata

In an SFC-enabled domain where pervasive monitoring [RFC7258] is possible, NSH metadata MUST be encrypted and MUST NOT reveal privacy



sensitive metadata to attackers. Privacy specific threats are discussed in [Section 5.2 of \[RFC6973\]](#).

Using K and AEAD algorithm, the NSH imposer encrypts the metadata (as set by the control plane [Section 3](#)) and inserts the resulting payload in the "MAC and Encrypted Metadata" Context Header ([Section 6.3](#)). The entire TLV carrying the privacy-sensitive metadata is encrypted (that is, including the MD Class, Type, Length, and associated metadata).

An authorized entity in the SFP that intends to update encrypted metadata MUST also follow the aforementioned behavior.

#### **7.4. Sequence Number Validation for Replay Attack**

A Sequence Number is an unsigned 64-bit counter value that increases by one for each NSH created and is sent from the NSH imposer, i.e., a per-key identifier packet sequence number. The information is mandatory and MUST always be present.

Processing of the Sequence Number field is at the discretion of the receiver, but all implementations MUST be capable of validating that the Sequence Number that does not duplicate the Sequence Number of any other NSH received during the life of the key identifier.

The NSH imposer's counter is initialized to '0' when a new key identifier is to be used. The sender increments the Sequence Number counter for this key identifier and inserts the 64-bit value into the Sequence Number Context Header ([Section 6.2](#)). Thus, the first NSH message (for a given service function chain) sent using a given key identifier will contain a Sequence Number of 1. The imposer checks to ensure that the counter has not cycled before inserting the new value in the Sequence Number Context Header. In other words, the sender MUST NOT send a packet with a key identifier if, by doing so, this would cause the Sequence Number to rollover.

Sequence Number counters of all participating nodes MUST be reset by establishing a new key identifier prior to the transmission of the 2<sup>64</sup>th packet of NSH for a particular key identifier.

#### **7.5. NSH Data Validation**

When an SFC data plane element receives an NSH, it MUST first ensure that all mandatory TLVs required for NSH data integrity are included. It MUST discard the message, if mandatory TLVs are absent or if the Sequence Number is invalid (described in [Section 7.4](#)). Otherwise, the SFC data plane element should then proceed with data validation. The SFC data plane element computes the message integrity for the



target NSH data (depending on the integrity protection scope discussed in [Section 6.3](#)) using K and AEAD algorithm for the key identifier being carried in NSH. If the value of the newly generated digest is identical to the one enclosed in NSH, the SFC data plane element is certain that the header has not been tampered and validation is therefore successful. Otherwise, the NSH message MUST be discarded.

#### **7.6. Decryption of NSH Metadata**

If entitled to consume a supplied encrypted metadata, an SFC-aware SF or SFC proxy decrypts metadata using K and decryption algorithm for the key identifier in NSH. AEAD algorithm has only a single output, either a plaintext or a special symbol FAIL that indicates that the inputs are not authentic ([Section 2.2 of \[RFC5116\]](#)).

There are cryptographic limits on the amount of plaintext which can be safely encrypted under a given set of keys. [\[AEAD-LIMITS\]](#) provides an analysis of these limits under the assumption that the underlying primitive (AES or ChaCha20) has no weaknesses. The NSH imposer SHOULD do a secret key update prior to reaching these limits.

### **8. Security Considerations**

NSH security considerations are discussed in [Section 8 of \[RFC8300\]](#).

The interaction between the SFC-aware data plane elements and a key management system MUST NOT be transmitted in clear since this would completely destroy the security benefits of the integrity protection scheme defined in this document.

NSH data are exposed to from to four primary attacks:

- o A man-in-the-middle attacker modifying NSH data.
- o Attacker spoofing NSH data.
- o Attacker capturing and replaying NSH data.
- o NSH metadata revealing privacy sensitive information to attackers.

In an SFC-enabled domain where the above attacks are possible, NSH data MUST be integrity- and replay-protected, and privacy-sensitive NSH metadata MUST be encrypted for confidentiality preservation purposes.

The Base header is not encrypted. An active attacker can potentially modify the Base header (e.g., decrement the TTL so the next SFF in



the SFP discards the NSH packet). In the meantime, an active attacker can also drop NSH packets. As such this is attack is not considered an attack against the security mechanism specified in the document.

The scope of the integrity protected data depends on the MAC flavor as discussed in the following sub-sections.

### **8.1. MAC#1**

No device other than the SFC-aware SFs in the SFC-enabled domain should be able to update the integrity protected NSH data. Similarly, no device other than the SFC-aware SFs and SFC proxies in the SFC-enabled domain be able to decrypt and update the metadata. In other words, if the SFC-aware SFs and SFC proxies in the SFC-enabled domain are considered fully trusted to act on the NSH data, only they can have access to privacy-sensitive NSH metadata and the keying material used to integrity protect NSH and encrypt metadata.

### **8.2. MAC#2**

SFFs can detect whether an illegitimate node has altered the content of the Base header. Such messages are discarded with appropriate logs and alarms generated.

This approach requires SFFs to have access to the shared key (which is used also for encrypting sensitive metadata). This may be undesired.

A dedicated key may be used for authenticating the Base header, while another key is used for the Service Path and Context Headers. Nevertheless, such approach will induce an overhead of multiple authentication tags.

## **9. IANA Considerations**

This document requests IANA to assign the following types from the "NSH IETF-Assigned Optional Variable-Length Metadata Types" (0x0000 IETF Base NSH MD Class) registry available at:

<https://www.iana.org/assignments/nsh/nsh.xhtml#optional-variable-length-metadata-types>.





Value	Description	Reference
TBD1	Key Identifier	[ThisDocument]
TBD2	Sequence Number	[ThisDocument]
TBD3	MAC and Encrypted Metadata#1	[ThisDocument]
TBD4	MAC and Encrypted Metadata#2	[ThisDocument]

## 10. Acknowledgements

This document was edited as a follow up to the discussion in IETF#104: <https://datatracker.ietf.org/meeting/104/materials/slides-104-sfc-sfc-chair-slides-01> (slide 7).

Thanks to Joel Halpern and Christian Jacquenet for the comments.

## 11. References

### 11.1. Normative References

- [GCM] Dworkin, M., "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC", NIST Special Publication 800-38D, November 2007.
- [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>>.
- [RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated Encryption", [RFC 5116](#), DOI 10.17487/RFC5116, January 2008, <<https://www.rfc-editor.org/info/rfc5116>>.
- [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>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", [RFC 8300](#), DOI 10.17487/RFC8300, January 2018, <<https://www.rfc-editor.org/info/rfc8300>>.



- [RFC8439] Nir, Y. and A. Langley, "ChaCha20 and Poly1305 for IETF Protocols", [RFC 8439](#), DOI 10.17487/RFC8439, June 2018, <<https://www.rfc-editor.org/info/rfc8439>>.

## **11.2. Informative References**

### **[AEAD-LIMITS]**

- Luykx, A. and K. Paterson, "Limits on Authenticated Encryption Use in TLS", 2016, <<http://www.isg.rhul.ac.uk/~kp/TLS-AEbounds.pdf>>.
- [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, <<https://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, <<https://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, <<https://www.rfc-editor.org/info/rfc7498>>.
- [RFC8459] Dolson, D., Homma, S., Lopez, D., and M. Boucadair, "Hierarchical Service Function Chaining (hSFC)", [RFC 8459](#), DOI 10.17487/RFC8459, September 2018, <<https://www.rfc-editor.org/info/rfc8459>>.

## **Appendix A. A Deployment Example with KMS**

SFC-aware SFs do not share any credentials; instead, they trust a third party, the KMS, with which they have or can establish shared credentials. These pre-established trust relations are used to establish a security association between SFC data plane elements within the context of a given service chain.

The NSH imposer requests a secret key and key identifier from the KMS. The request message also includes identities of the SFC data plane elements (including SFC-aware SFs and SFC proxies) authorized to receive the keying material associated with the key identifier. Each SFC-aware SF is referenced using an SF identifier that is unique within an SFC-enabled domain. If the request is authorized, then KMS generates the secret key (K), key identifier (kid), and returns them in a response message. The key identifier may be self-contained (key



encrypted in the key identifier) or just a handle to some internal data structure within the KMS.

The NSH imposer includes the key identifier in NSH data. The NSH data is protected using K and optionally metadata is encrypted using K. SFC-aware SFs and SFC proxies in the SFP forward the key identifier to the KMS and request the KMS to retrieve the keying material. If the SFC data plane element is authorized and the key identifier is valid, then the KMS retrieves the secret key and AEAD algorithm associated with the key identifier and conveys them to the SFC data plane element. The other alternative approach is that KMS implicitly pushes the keying material to, particularly, SFC-aware SFs and SFC proxies authorized by the NSH imposer.

If the NSH imposer requests a new key and a new key identifier from KMS, the request message from NSH imposer to KMS also includes identities of SFC-aware SFs and SFC proxies authorized to receive the keying material associated with the new key identifier. For subsequent packets, the new key identifier will be conveyed in the NSH data, NSH data will be integrity protected using the new secret key and optionally NSH metadata is encrypted using the new secret key.

Figure 7 shows an example of an NSH imposer requesting a secret key and key identifier from the KMS. The request message includes identifiers of SF1 and SF2 Service Functions authorized to receive keying material associated with the key identifier. KMS returns the secret key (K) and key identifier in the response message. The NSH imposer includes the key identifier in the NSH data. In this example, SF1 in the SFP forwards the key identifier to the KMS and requests the KMS for keying material associated with the key identifier (In key resolve request message). If SF1 is authorized and the key identifier is valid then KMS retrieves the key and AEAD algorithm associated with the key identifier and conveys them to the SF1 (In Resolve response message). Similarly, SF2 retrieves the keying material associated with the key identifier from KMS.

Note: Update the example with the SFF

The exchange with KMS is not required if the necessary information is pre-provisioned to the authorized SFC-aware SFs and SFC proxies.



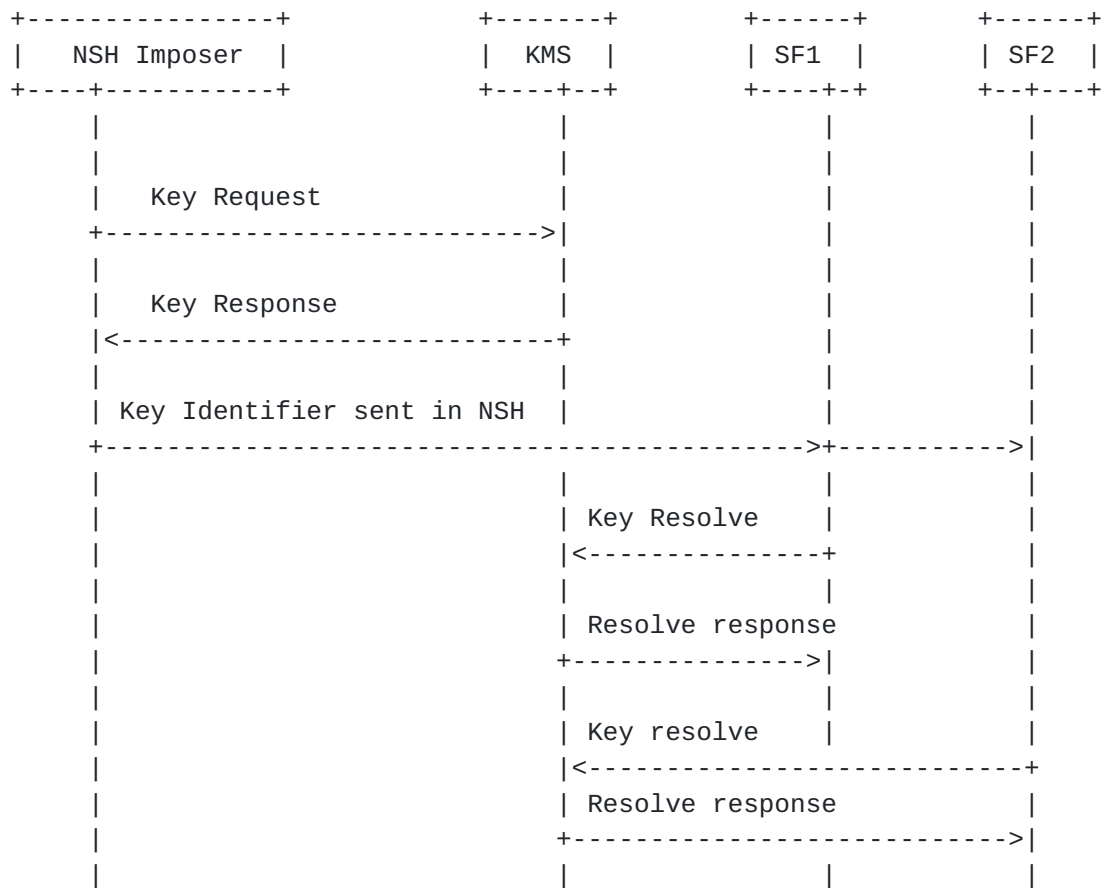


Figure 7: Example of Interactions with KMS

## Authors' Addresses

Mohamed Boucadair  
 Orange  
 Rennes 35000  
 France

Email: mohamed.boucadair@orange.com

Tirumaleswar Reddy  
 McAfee, Inc.  
 Embassy Golf Link Business Park  
 Bangalore, Karnataka 560071  
 India

Email: TirumaleswarReddy\_Konda@McAfee.com



