

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: January 3, 2015

S. Aldrin
Huawei Technologies
C. Pignataro
N. Akiya
Cisco Systems
July 2, 2014

Service Function Chaining
Operations, Administration and Maintenance Framework
draft-aldrin-sfc-oam-framework-00

Abstract

This document provides reference framework for Operations, Administration and Maintenance (OAM) of 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 <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 30, 2014.

Copyright Notice

Copyright (c) 2014 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	2
1.1. Document Scope	3
2. SFC Layering Model	3
3. SFC OAM Components	4
3.1. Service Function Component	5
3.1.1. Service Function Availability	5
3.1.2. Service Function Performance Measurement	6
3.2. Service Function Chain Component	6
3.2.1. Service Function Chain Availability	6
3.2.2. Service Function Chain Performance Measurement	6
3.3. Classifier Component	7
4. SFC OAM Functions	7
4.1. Connectivity Functions	7
4.2. Continuity Functions	8
4.3. Trace Functions	8
4.4. Performance Measurement Function	8
5. Gap Analysis	9
5.1. Existing OAM Functions	9
5.2. Missing OAM Functions	10
5.3. Required OAM Functions	10
6. Security Considerations	10
7. IANA Considerations	10
8. Acknowledgements	10
9. References	11
9.1. Normative References	11
9.2. Informative References	11
Authors' Addresses	11

1. Introduction

Service Function Chaining (SFC) enables the creation of composite services that consist of an ordered set of Service Functions (SF) that must be applied to packets and/or frames selected as a result of classification. Service Function Chaining is a concept that provides for more than just the application of an ordered set of SFs to selected traffic; rather, it describes a method for deploying SFs in a way that enables dynamic ordering and topological independence of those SFs as well as the exchange of metadata between participating entities. Foundations of the SFC are described in below documents:

- o [I-D.ietf-sfc-problem-statement]: SFC problem statement.

- o Various individual drafts

This document provides reference framework for Operations, Administration and Maintenance (OAM, [RFC6291]) of the SFC. Specifically, this document provides:

- o In Section 2, an SFC layering model;
- o In Section 3, involved components within the SFC layer;
- o In Section 4, functional requirements for the SFC OAM;
- o In Section 5, an OAM gap analysis.

1.1. Document Scope

The focus of this document is to provide an architectural framework for the SFC OAM, particularly focused on the aspect of the Operation portion of the OAM. Actual solutions and mechanisms are outside the scope of this document.

2. SFC Layering Model

Multiple layers come into play for implementing the SFC. These include the service layer at SFC layer and the underlying Network, Transport, Link, etc., layers.

- o The service layer, refer to as the "Service Layer" in Figure 1, consists of classifiers and service functions, and uses the overlay network reach from a classifier to service functions and service functions to service functions.
- o The network overlay transport layer, refer to as the "Network", "transport" and layers below in Figure 1, extends in between various service functions and is mostly transparent to the service functions. It leverages various overlay network technologies interconnecting service functions and allows establishing of service function paths.
- o The link layer, refer to as the "Link" in Figure 1, is dependent upon the physical technology used. Ethernet is a popular choice for this layer, but other alternatives are deployed (e.g. POS, DWDM etc...).

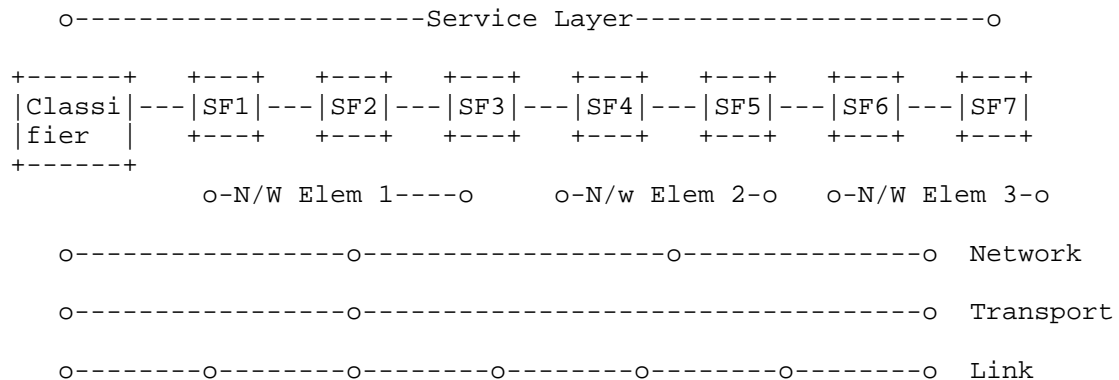


Figure 1: SFC Layering Example

3. SFC OAM Components

The SFC operates at the service layer. For the purpose of defining the OAM framework, the service layer is broken up into three distinct components.

1. Service function component: A function providing a specific service. OAM solutions for this component are to test the service functions from any SFC aware network devices (i.e. classifiers, controllers, other service nodes).
2. Service function chain component: An ordered set of service functions. OAM solution for this component are to test the service function chains and the service function paths.
3. Classifier component: A policy that describes the mapping from flows to service function chains. OAM solutions for this component are to test the validity of the classifiers.

Below figure illustrates an example where OAM for the three defined components are used within the SFC environment.

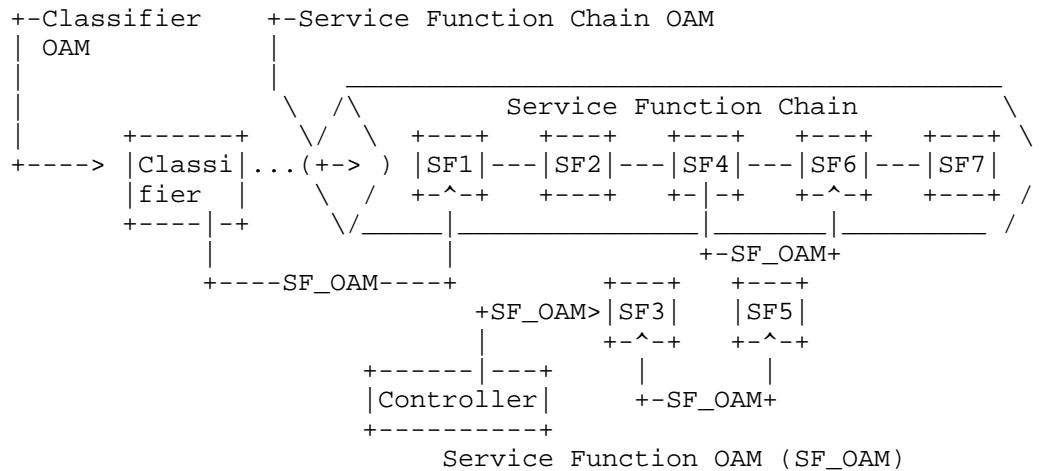


Figure 2: SFC OAM for Three Components

It is expected that multiple SFC OAM solutions will be defined, many targeting one specific component of the service layer. However, it is critical that SFC OAM solutions together provide the coverage of all three SFC OAM components: the service function component, the service function chain component and the classifier component.

3.1. Service Function Component

3.1.1. Service Function Availability

One SFC OAM requirement for the service function component is to allow an SFC aware network device to check the availability to a specific service function, located on the same or different network devices. Service function availability is an aspect which raises an interesting question. How does one determine that a service function is available? On one end of the spectrum, one might argue that a service function is sufficiently available if the service node (physical or virtual) hosting the service function is available and is functional. On the other end of the spectrum, one might argue that the service function availability can only be concluded if the packet, after passing through the service function, was examined and verified that the packet got expected service applied.

The former approach will likely not provide sufficient confidence to the actual service function availability, i.e. a service node and a service function are two different entities. The latter approach is capable of providing an extensive verification, but comes with a cost. Some service functions make direct modifications to packets, while other service functions do not make any modifications to

packets. Additionally, purpose of some service functions is to, conditionally, drop packets intentionally. In such case, packets will not be coming out from the service function. The fact is that there are many flavors of service functions available, and many more flavors of service functions will likely be introduced in future. Even a given service function may introduce a new functionality within a service function (ex: a new signature in a firewall). The cost of this approach is that verifier functions will need to be continuously modified to "keep up" with new services coming out: lack of extendibility.

This framework document provides a RECOMMENDED architectural model where generalized approach is taken to verify that a service function is sufficiently available. TBD - details will be provided in a later revision.

3.1.1.2. Service Function Performance Measurement

Second SFC OAM requirement for the service function component is to allow an SFC aware network device to check the loss and delay of a specific service function, located on the same or different network devices. TBD - details will be provided in a later revision.

3.2. Service Function Chain Component

3.2.1. Service Function Chain Availability

Verifying an SFC is a complicated process as the SFC could be comprised of varying SF's. Thus, SFC requires the OAM layer to perform validation and verification of SF's within an SFC Path, as well as connectivity and fault isolation.

In order to perform service connectivity verification of an SFC, the OAM could be initiated from any SFC aware network devices for end-to-end paths or partial path terminating on a specific SF within the SFC. This OAM function is to ensure the SF's chained together has connectivity as it is intended to when SFC was established. Necessary return code should be defined to be sent back in the response to OAM packet, in order to qualify the verification.

When ECMP exists at the service layer on a given SFC, there must be an ability to discover and traverse all available paths.

TBD - further details will be provided in a later revision.

3.2.2. Service Function Chain Performance Measurement

The ingress of the service function chain or an SFC aware network

device must have an ability to perform loss and delay measurements over the service function chain as a unit (i.e. end-to-end) or to a specific service function through the SFC.

3.3. Classifier Component

A classifier defines a flow and maps incoming traffic to a specific SFC, and it is vital that the classifier is correctly defined and functioning. The SFC OAM must be able to test the definition of flows and the mapping functionality to expected SFCs.

4. SFC OAM Functions

Section 3 described SFC OAM operations required on each SFC component. This section explores the same from the OAM functionality point of view, which many will be applicable to multiple SFC components.

Various SFC OAM requirements provides the need for various OAM functions at different layers. Many of the OAM functions at different layers are already defined and in existence. In order to support SFC and SF's, these functions have to be enhanced to operate a single SF to multiple SF's in an SFC and also multiple SFC's.

4.1. Connectivity Functions

Connectivity is mainly an on-demand function to verify that the connectivity exists between network elements and the availability exists to service functions. Ping is a common tool used to perform this function. OAM messages should be encapsulated with necessary SFC header and with OAM markings when testing the service function chain component. OAM messages MAY be encapsulated with necessary SFC header and with OAM markings when testing the service function component. Some of the OAM functions performed by connectivity functions are as follows:

- o Verify the MTU size from a source to the destination SF or through the SFC. This requires the ability for OAM packet to take variable length packet size.
- o Verify the packet re-ordering and corruption.
- o Verify the policy of an SFC or SF using OAM packet.
- o Verification and validating forwarding paths.
- o Proactively test alternate or protected paths to ensure reliability of network configurations.

4.2. Continuity Functions

Continuity is a model where OAM messages are sent periodically to validate or verify the reachability to a given SF or through a given SFC. This allows monitor network device to quickly detect failures like link failures, network failures, service function outages or service function chain outages. BFD is one such function which helps in detecting failures quickly. OAM functions supported by continuity check are as follows:

- o Ability to provision continuity check to a given SF or through a given SFC.
- o Notifying the failure upon failure detection for other OAM functions to take appropriate action.

4.3. Trace Functions

Tracing is an important OAM function that allows the operation to trigger an action (ex: response generation) from every transit device on the tested layer. This function is typically useful to gather information from every transit devices or to isolate the failure point towards an SF or through an SFC. Some of the OAM functions supported by trace functions are:

- o Ability to trigger action from every transit device on the tested layer towards an SF or through an SFC, using TTL or other means.
- o Ability to trigger every transit device to generate response with OAM code(s) on the tested layer towards an SF or through an SFC, using TTL or other means.
- o Ability to discover and traverse ECMP paths within an SFC.
- o Ability to skip un-supported SF's while tracing SF's in an SFC.

4.4. Performance Measurement Function

Performance management functions involve measuring of packet loss, delay, delay variance, etc. These measurements could be measured pro-actively and on-demand.

SFC OAM framework should provide the ability to perform packet loss for an SFC. In an SFC, there are various SF's chained together. Measuring packet loss is very important function. Using on-demand function, the packet loss could be measured using statistical means. Using OAM packets, the approximation of packet loss for a given SFC could be measured.

Delay within an SFC could be measured from the time it takes for a packet to traverse the SFC from ingress SF to egress SF. As the SFC's are generally unidirectional in nature, measurement of one-way delay is important. In order to measure one-way delay, the clocks have to be synchronized using NTP, GPS, etc.

Delay variance could also be measured by sending OAM packets and measuring the jitter between the packets passing through the SFC.

Some of the OAM functions supported by the performance measurement functions are:

- o Ability to measure the packet processing delay of a service function or a service function path along an SFC.
- o Ability to measure the packet loss of a service function or a service function path along an SFC.

5. Gap Analysis

This Section identifies various OAM functions available at different levels. It will also identify various gaps, if not all, existing within the existing toolset, to perform OAM function on an SFC.

5.1. Existing OAM Functions

There are various OAM tool sets available to perform OAM function and network layer, protocol layers and link layers. These OAM functions could validate some of the network overlay transport. Tools like ping and trace are in existence to perform connectivity check and tracing intermediate hops in a network. These tools support different network types like IP, MPLS, TRILL etc. There is also an effort to extend the tool set to provide connectivity and continuity checks within overlay networks. BFD is another tool which helps in detection of data forwarding failures.

Layer	Connectivity	Continuity	Trace	Performance
N/W Overlay	Ping	BFD, NVo3	Trace	IPPM
SF	None	+ None	+ None	+ None
SFC	None	+ None	+ None	+ None

Figure 3: OAM Tool GAP Analysis

5.2. Missing OAM Functions

As shown in Figure 3, OAM functions for SFC are not standardized yet. Hence, there are no standard based tools available to verify SF and SFC.

5.3. Required OAM Functions

Primary OAM functions exist for network, transport, link and other layers. Tools like ping, trace, BFD, etc., exist in order to perform these OAM functions. Configuration, orchestration and manageability of SF and SFC could be performed using CLI, Netconf etc.

For configuration, manageability and orchestration, providing data and information models for SFC is very much essential. With virtualized SF and SFC, manageability of these functions has to be done programmatically.

6. Security Considerations

SFC and SF OAM must provide mechanisms for:

- o Preventing usage of OAM channel for DDOS attacks.
- o OAM packets meant for a given SFC should not get leaked beyond that SFC.
- o Prevent OAM packets to leak the information of an SFC beyond its administrative domain.

7. IANA Considerations

No action is required by IANA for this document.

8. Acknowledgements

TBD

9. References

9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

9.2. Informative References

- [I-D.ietf-sfc-problem-statement]
Quinn, P. and T. Nadeau, "Service Function Chaining Problem Statement", draft-ietf-sfc-problem-statement-07 (work in progress), June 2014.
- [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, June 2011.

Authors' Addresses

Sam K. Aldrin
Huawei Technologies

Email: aldrin.ietf@gmail.com

Carlos Pignataro
Cisco Systems

Email: cpignata@cisco.com

Nobo Akiya
Cisco Systems

Email: nobo@cisco.com

SFC
Internet-Draft
Intended status: Informational
Expires: August 17, 2015

M. Boucadair
C. Jacquenet
France Telecom
Y. Jiang
Huawei Technologies Co., Ltd.
R. Parker
Affirmed Networks
K. Naito
NTT
February 13, 2015

Requirements for Service Function Chaining (SFC)
draft-boucadair-sfc-requirements-06

Abstract

This document identifies the requirements for the Service Function Chaining (SFC).

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on August 17, 2015.

Copyright Notice

Copyright (c) 2015 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	2
2. Terminology	2
3. Detailed Requirements List	3
3.1. Instantiating and Invoking Service Functions	3
3.2. Chaining Service Functions	4
3.3. MTU Requirements	5
3.4. Independence from the Underlying Transport Infrastructure Requirements	6
3.5. Traffic Classification Requirements	6
3.6. Data Plane Requirements	7
3.7. OAM Requirements	8
3.8. Recovery and Load Balancing Requirements	10
3.9. Compatibility with Legacy Service Functions Requirements	11
3.10. QoS Requirements	11
3.11. Security Requirements	11
4. IANA Considerations	12
5. Security Considerations	12
6. Contributors	12
7. Acknowledgements	13
8. References	13
8.1. Normative References	13
8.2. Informative References	13

1. Introduction

This document identifies the requirements for the Service Function Chaining (SFC).

The overall problem space is described in [I-D.ietf-sfc-problem-statement].

2. Terminology

The reader should be familiar with the terms defined in [I-D.ietf-sfc-problem-statement].

The document makes use of the following terms:

- o SFC-enabled domain: denotes a network (or a region thereof) that implements SFC.
- o Service Function Loop: If a Service Function Chain is structured to not invoke Service Functions multiple times, a loop is the error that occurs when the same Service Function is invoked several times when handling data bound to that Service Function Chain. In other words, a loop denotes an error that occurs when a packet handled by a Service Function, forwarded onwards, and arrives once again at that Service Function while this is not allowed by the Service Function Chain it is bound to.
- o Service Function Spiral: denotes a Service Function Chain in which data is handled by a Service Function, forwarded onwards, and arrives once again at that Service Function.
 - * Note that some Service Functions support built-in functions to accommodate spirals; these service-specific functions may require that the data received in a spiral should differ in a way that will result in a different processing decision than the original data. This document does not make such assumption.
 - * A Service Function Chain may involve one or more Service Function Spirals.
 - * Unlike Service Function loop, spirals are not considered as errors.

3. Detailed Requirements List

The following set of functional requirements should be considered for the design of the Service Function Chaining solution.

3.1. Instantiating and Invoking Service Functions

- SF_REQ#1: The solution MUST NOT make any assumption on whether Service Functions (SF) are deployed directly on physical hardware, as one or more Virtual Machines, or any combination thereof.
- SF_REQ#2: The solution MUST NOT make any assumption on whether Service Functions each reside on a separate addressable Network Element, or as a horizontal scaling of Service Functions, or are co-resident in a single addressable Network Element, or any combination thereof.

Note: Communications between Service Functions having the same locator are considered implementation-specific. These considerations are therefore out of scope of the SFC specification effort.

- SF_REQ#3: The solution MUST NOT require any IANA registry for Service Functions.
- SF_REQ#4: The solution MUST allow multiple instances of a given Service Function (i.e., instances of a Service Function can be embedded in or attached to multiple Network Elements).
- A. This is used for load-balancing, load-sharing, to minimize the impact of failures (e.g., by means of a hot or cold standby protection design), to accommodate planned maintenance operations, etc.
 - B. How these multiple devices are involved in the service delivery is deployment-specific.
- SF_REQ#5: The solution MUST separate SF-specific policy provisioning-related aspects from the actual handling of packets (including forwarding decisions).

3.2. Chaining Service Functions

- SFC_REQ#1: The solution MUST NOT assume any predefined order of Service Functions. In particular, the solution MUST NOT require any IANA registry to store typical Service Function Chains.
- SFC_REQ#2: The identification of instantiated Service Function Chains is local to each administrative domain; it is policy-based and deployment-specific.
- SFC_REQ#3: The solution MUST allow for multiple Service Chains to be simultaneously enforced within an administrative domain.
- SFC_REQ#4: The solution MUST allow the same Service Function to belong to multiple Service Function Chains.
- SFC_REQ#5: The solution MUST support the ability to deploy multiple SFC-enabled domains within the same administrative domain.
- SFC_REQ#6: The solution MUST be able to associate the same or distinct Service Function Chains for each direction

(inbound/outbound) of the traffic pertaining to a specific service. In particular, unidirectional Service Function Chains, bi-directional Service Function Chains, or any combination thereof MUST be supported.

Note, the solution must allow to involve distinct SFC Boundary Nodes for upstream and downstream. Multiple SFC Boundary Nodes may be deployed within an administrative domain.

SFC_REQ#7: The solution MUST be able to dynamically enforce Service Function Chains. In particular, the solution MUST allow the update or the withdrawal of existing Service Function Chains, the definition of a new Service Function Chain, the addition of new Service Functions without having any impact on other existing Service Functions or other Service Function Chains.

SFC_REQ#8: The solution MUST provide means to control the SF-inferred information to be leaked outside an SFC-enabled domain. In particular, an administrative entity MUST be able to prevent the exposure of the Service Function Chaining logic and its related policies outside the administrative domain.

SFC_REQ#9: The solution MUST prevent infinite Service Function Loops.

A. Service Functions MAY be invoked multiple times in the same Service Function Chain (denoted as SF Spiral), but the solution MUST prevent infinite forwarding loops.

3.3. MTU Requirements

Packet fragmentation can be very expensive in SFC environment where fragmented packets have to be reassembled before sending to each SF on the chain. It is also worth noting that IPv6 traffic can only be fragmented by the end systems.

MTU_REQ#1: The solution SHOULD minimize fragmentation; in particular, a minimal set of SFC-specific information should be conveyed in the data packet.

MTU_REQ#2: Traffic forwarding on a SFC basis MUST be undertaken without relying on dedicated resources to treat fragments. In particular, Out of order fragments MUST be

forwarded on a per-SFC basis without relying on any state.

MTU_REQ#3: Some SFs (e.g., NAT) may require dedicated resources (e.g., resources to store fragmented packets) or they may adopt a specific behavior (e.g, limit the time interval to accept fragments). The solution MUST NOT interfere with such practices.

3.4. Independence from the Underlying Transport Infrastructure Requirements

UN_REQ#1: The solution MUST NOT make any assumption on how RIBs (Routing Information Bases) and FIBs (Forwarding Information Bases) are populated. Particularly, the solution does not make any assumption on protocols and mechanisms used to build these tables.

UN_REQ#2: The solution MUST be transport independent.

A. The Service Function Chaining should operate regardless of the network transport used by the administrative entity. In particular, the solution can be used whatever the switching technologies deployed in the underlying transport infrastructure.

B. Techniques such as MPLS are neither required nor excluded.

UN_REQ#3: The solution MUST allow for chaining logics where involved Service Functions are not within the same layer 3 subnet.

UN_REQ#4: The solution MUST NOT exclude Service Functions to be within the same IP subnet (because this is deployment-specific).

3.5. Traffic Classification Requirements

TC_REQ#1: The solution MUST NOT make any assumption on how the traffic is to be bound to a given chaining policy. In other words, classification rules are deployment-specific and policy-based. For instance, classification can rely on a subset of the information carried in a received packet such as 5-tuple classification, be subscriber-aware, be driven by traffic engineering considerations, or any combination thereof.

Because a large number (e.g., 1000s) of classification policy entries may be configured, means .Means to reduce classification look-up time such as optimizing the size of the classification table (e.g., aggregation) should be supported by the Classifier.

TC_REQ#2: The solution MUST NOT require every Service Function to be co-located with a SFC Classifier; this is a deployment-specific decision.

TC_REQ#3: The solution MAY allow traffic re-classification at the level of Service Functions (i.e., a Service Function can also be co-located with a Classifier). The configuration of classification rules in such context are the responsibility of the administrative entity that operates the SFC-enabled domain.

TC_REQ#4: The solution MUST allow Service Function Nodes to be configured (or pushed) with the detailed policies on which local Service Functions to invoke for packets associated with some Service Function Chains. The solution MUST allow those steering policies to be updated based on demand.

3.6. Data Plane Requirements

DP_REQ#1: The solution MUST be able to forward traffic between two Service Functions (involved in the same Service Function Chain) without relying upon the destination address field of the a data packet.

DP_REQ#2: The solution MUST allow for the association of a context with the data packets. In particular:

A. The solution MUST support the ability to invoke differentiated sets of policies for a Service Function (such sets of policies are called Profiles). A profile denotes a set of policies configured to a local Service Function (e.g., content-filter-child, content-filter-adult).

a. Few profiles should be assumed per Service Function to accommodate the need for scalable solutions.

b. A finer granularity of profiles may be configured directly to each Service Function; there is indeed

no need to overload the design of Service Function Chains with policies of low-level granularity.

DP_REQ#3: Service Functions may be reachable using IPv4 and/or IPv6. The administrative domain entity MUST be able to define and enforce policies with regards to the address family to be used when invoking a Service Function.

- A. A Service Function Chain may be composed of IPv4 addresses, IPv6 addresses, or a mix of both IPv4 and IPv6 addresses.
- B. Multiple Service Functions can be reachable using the same IP address. Each of these Service Functions is unambiguously identified with a Service Function Identifier.

DP_REQ#4:

3.7. OAM Requirements

OAM_REQ#1: The solution MUST allow for Operations, Administration, and Maintenance (OAM) features [RFC6291]. In particular, the solution MUST:

- A. Support means to verify the completion of the forwarding actions until the SFC Border Node is reached (see Section 3.4.1 of [RFC5706]).
- B. Support means to ensure coherent classification rules are installed in and enforced by all the Classifiers of the SFC domain.
- C. Support means to correlate classification policies with observed forwarding actions.
- D. Support in-band liveness and functionality checking mechanisms for the instantiated Service Function Chains and the Service Functions that belong to these chains.

OAM_REQ#2: The solution MUST support means to detect the liveness of Service Functions of an SFC-enabled domain. In particular, the solution MUST support means to (dynamically) detect that a Service Function instance is out of service and notify the relevant elements accordingly (PDP and Classifiers, for one).

OAM_REQ#3: Detailed diagnosis requirements are listed below:

- A. The solution MUST allow to assess the status of the serviceability of a Service Function (i.e., the Service Function provides the service(s) it is configured for).
- B. The solution MUST NOT rely only on IP reachability to assess whether a Service Function is up and running.
- C. The solution MUST allow to diagnose the availability of a Service Function Chain (including the availability of a particular Service Function Path bound to a given Service Function Chain).
- D. The solution MUST allow to retrieve the set of Service Function Chains that are enabled within a domain.
- E. The solution MUST allow to retrieve the set of s Service Function Chains in which a given Service Function is involved.
- F. The solution MUST allow to assess whether an SFC-enabled domain is appropriately configured (including the configured chains are matching what should be configured in that domain).
- G. The solution MUST allow to assess the output of the classification rule applied on a packet presented to a Classifier of an SFC-enabled domain.
- H. The solution MUST support the correlation between a Service Function Chain and the actual forwarding path followed by a packet matching that SFC.
- I. The solution MUST allow to diagnose the availability of a segment of a Service Function Chain, i.e., a subset of Service Functions that belong to the said chain.
- J. The solution MUST support means to notify the PDPs whenever some events occur (for example, a malfunctioning Service Function instance).
- K. The solution MUST allow for local diagnostic procedures specific to each Service Function (i.e., SF built-in diagnostic procedures).

- L. The solution MUST allow for customized service diagnostic.

OAM_REQ#4: Liveness status records for all Service Functions (including Service Function instances), Service Function Nodes, Service Function Chains (including the Service Function Paths bound to a given chain) MUST be maintained.

OAM_REQ#5: SFC-specific counters and statistics MUST be provided. These data include (but not limited to):

- * Number of flows ever and currently assigned to a given Service Function Chain and a given Service Function Path.
- * Number of flows, packets, bytes dropped due to policy.
- * Number of packets and bytes in/out per Service Function Chain and per Service Function Path.
- * Number of flows, packets, bytes dropped due to unknown Service Function Chain or Service Function Path (this is valid in particular for a Service Function Node).

3.8. Recovery and Load Balancing Requirements

LB_REQ#1: The solution MUST allow for load-balancing among multiple instances of the same Service Function.

- A. Load-balancing may be provided by legacy technologies or protocols (e.g., make use of load-balancers)
- B. Load-balancing may be part of the Service Function itself.
- C. Load-balancer may be considered as a Service Function element.
- D. Because of the possible complications, load balancing SHOULD NOT be driven by the SFC Classifier.

LB_REQ#2: The solution MUST separate SF-specific policy provisioning-related aspects from the actual handling of packets (including forwarding decisions).

LB_REQ#3: The solution SHOULD support protection of the failed or over-utilized Service Function instances. The protection

mechanism can rely on local decisions among the nodes that are connected to both active/standby Service Function instances.

3.9. Compatibility with Legacy Service Functions Requirements

LEG_REQ#1: The solution MUST allow for gradual deployment in legacy infrastructures, and therefore coexist with legacy technologies that cannot support SFC-specific capabilities, such as Service Function Chain interpretation and processing. The solution MUST be able to work in a domain that may be partly composed of opaque elements, i.e., elements that do not support SFC-specific capabilities.

3.10. QoS Requirements

QoS_REQ#1: The solution MUST be able to provide different SLAs (Service Level Agreements, [RFC7297]). In particular,

- A. The solution MUST allow for different levels of service to be provided for different traffic streams (e.g., configure Classes of Service (CoSes)).
- B. The solution MUST be able to work properly within a Diffserv domain [RFC2475].
- C. The solution SHOULD support the two modes defined in [RFC2983].

QoS_REQ#2: ECN re-marking, when required, MUST be performed according to [RFC6040].

3.11. Security Requirements

SEC_REQ#1: The solution MUST provide means to prevent any information leaking that would be used as a hint to guess internal engineering practices (e.g., network topology, service infrastructure topology, hints on the enabled mechanisms to protect internal service infrastructures, etc.).

The solution MUST support means to protect the SFC domain as a whole against attacks that would lead to the discovery of Service Functions enabled in a SFC domain.
In particular, topology hiding means MUST be supported to avoid the exposure of the SFC-enabled domain

topology (including the set of the service function chains supported within the domain and the corresponding Service Functions that belong to these chains).

SEC_REQ#2: The solution MUST support means to protect the SFC-enabled domain against any kind of denial-of-service and theft of service (e.g., illegitimate access to the service) attack.

For example, a user should not be granted access to connectivity services he/she didn't subscribe to (including direct access to some SFs), at the risk of providing illegitimate access to network resources.

SEC_REQ#3: The solution MUST NOT interfere with IPsec [RFC4301] (in particular IPsec integrity checks).

4. IANA Considerations

This document does not require any action from IANA.

5. Security Considerations

Some security-related requirements to be taken into account when designing the Service Function Chaining solution are listed in Section 3.11. These requirements do not cover the provisioning interface used to enforce policies into the Classifier, Service Functions, and Service Function Nodes.

6. Contributors

The following individuals contributed text to the document:

Hongyu Li
Huawei Technologies Co., Ltd.
Bantian, Longgang district
Shenzhen 518129,
China

EMail: hongyu.lihongyu@huawei.com

Jim Guichard
Cisco Systems, Inc.
USA

EMail: jguichar@cisco.com

Paul Quinn
Cisco Systems, Inc.
USA

Email: paulq@cisco.com

Linda Dunbar
Huawei Technologies
5430 Legacy Drive, Suite #175
Plano TX
USA

EMail: linda.dunbar@huawei.com

7. Acknowledgements

Many thanks to K. Gray, N. Takaya, H. Kitada, H. Kojima, D. Dolson, B. Wright, and J. Halpern for their comments.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

8.2. Informative References

[I-D.ietf-sfc-problem-statement]
Quinn, P. and T. Nadeau, "Service Function Chaining Problem Statement", draft-ietf-sfc-problem-statement-11 (work in progress), February 2015.

- [RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.
- [RFC2983] Black, D., "Differentiated Services and Tunnels", RFC 2983, October 2000.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC5706] Harrington, D., "Guidelines for Considering Operations and Management of New Protocols and Protocol Extensions", RFC 5706, November 2009.
- [RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, November 2010.
- [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, June 2011.
- [RFC7297] Boucadair, M., Jacquenet, C., and N. Wang, "IP Connectivity Provisioning Profile (CPP)", July 2014.

Authors' Addresses

Mohamed Boucadair
France Telecom
Rennes 35000
France

EMail: mohamed.boucadair@orange.com

Christian Jacquenet
France Telecom
Rennes 35000
France

EMail: christian.jacquenet@orange.com

Yuanlong Jiang
Huawei Technologies Co., Ltd.
Bantian, Longgang district
Shenzhen 518129,
China

EMail: jiangyuanlong@huawei.com

Ron Parker
Affirmed Networks
Acton, MA
USA

EMail: Ron_Parker@affirmednetworks.com

Kengo Naito
NTT
Midori-Cho 3-9-11
Musashino-shi, Tokyo 180-8585
Japan

EMail: naito.kengo@lab.ntt.co.jp

SFC Working Group
Internet Draft
Category: Informational

R. Krishnan
Brocade
A. Ghanwani
Dell
Pedro A. Aranda Gutierrez
D. R. Lopez
Telefonica I+D
J. Halpern
S. Kini
Ericsson
Andy Reid
BT

Expires: October 2014

July 3, 2014

SFC OAM Requirements and Framework

draft-krishnan-sfc-oam-req-framework-00

Abstract

This document discusses SFC OAM requirements and proposes a SFC OAM Framework to handle these requirements.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on April, 2014.

Copyright Notice

Copyright (c) 2014 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.

Conventions used in this document

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 [RFC 2119].

Table of Contents

1. Introduction.....	3
1.1. Acronyms.....	4
2. SFC OAM Requirements.....	4
2.1. Topologies.....	4
2.2. Connectivity.....	4
2.2.1. Connectivity Check.....	4
2.2.2. SFP Trace.....	5
2.3. Performance.....	5
2.4. Leakage of OAM Messages.....	5
2.5. Appliance Types.....	5
3. IANA Considerations.....	6
4. Security Considerations.....	6
5. Acknowledgements.....	6
6. References.....	6
6.1. Normative References.....	6
6.2. Informative References.....	6
Authors' Addresses.....	7

1. Introduction

Operations, administration, and maintenance (OAM) is the general term applied to monitoring both the connectivity and performance in the network [RFC 6291] [RFC 7276]. The goal of SFC OAM then is to monitor these attributes for a service function chain (SFC).

Some clarification is needed regarding the scope of this work. SFC OAM does will not attempt to monitor the actual services. Also, SFC OAM does not replace or obviate the need for transport-level OAM functions such as NVO3 OAM, IEEE 802.1ag, MPLS OAM, or whatever else may be applicable depending on the network technology that the SFC is implemented on.

The following figure depicts the layering of OAM.

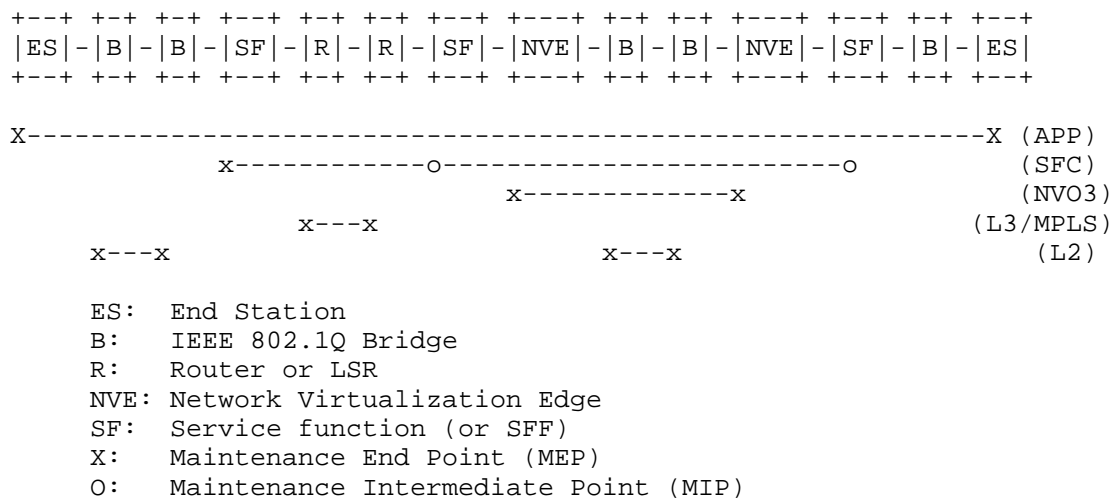


Figure 1: Layered OAM Architecture

The SFC layer resides above the transport layer (where the transport layer can simply be implemented using VLANs or may be done using overlays such as VXLAN or NVGRE), and below the application layer (APP). As mentioned earlier, depending on the underlying network technology, other OAM layers may be present (NVO3 OAM [NVO3 OAM], L3/MPLS OAM [RFC 7276], IEEE 802.1ag CFM [IEEE 802.1ag], etc.). The use of the terms maintenance end point (MEP) and maintenance (MIP) are consistent with IEEE 802.1Q are simply used to denote points where monitoring services are configured.

The systems denoted SF refer to devices in the network that either insert, modify, remove, or access the service chain header (SCH) [SCH draft]. These nodes may implement the actual service function (as would be the case for an SF-aware appliance) or they may be proxy nodes such as SFFs with the service function itself residing in a different device (as would be the case for an SF-unaware appliance).

1.1. Acronyms

DPI:	Deep Packet Inspection
MPLS:	Multiprotocol Label Switching
NVGRE:	Network Virtualization using Generic Routing Encapsulation
OAM:	Operations, Administration, and Maintenance
SF:	Service Function
SFC:	Service Function Chain
SFP:	Service Function Path
VXLAN:	Virtual Extensible LAN

2. SFC OAM Requirements

2.1. Topologies

Mechanisms must be provided to monitor the entire SFP or just a portion of the SFP.

SFC OAM must also be able to handle various topologies that can be created such a point-to-point or multipoint.

2.2. Connectivity

2.2.1. Connectivity Check

The purpose of the connectivity check tool is to test the liveness of a given service function along a given SFP (service function path).

Mechanisms must be provided so that the SFC OAM messages may be sent along the same path that a given data packet would follow. In other words, it should be possible to construct SFC OAM packets that would be treated by network devices such as bridges and routers as they would handle regular data packets on that SFP from the standpoint of functions such as link aggregation and equal cost multipath.

2.2.2. SFP Trace

The purpose of SFP trace is to provide the list of SFs that comprise the service function chain as defined by the SCH.

Mechanisms must be provided so that the SFC OAM messages may be sent along the same path that a given data packet would follow. In other words, it should be possible to construct SFC OAM packets that would be treated by network devices such as bridges and routers as they would handle regular data packets on that SFP from the standpoint of functions such as link aggregation and equal cost multipath.

2.3. Performance

It must be possible to measure various parameters of a given SFP such as the loss, delay, and delay variation through the service chain.

[Ed Note: Details TBD]

2.4. Leakage of OAM Messages

Mechanisms must be provided to ensure that OAM messages are received only by devices that need to process them. These messages must never be forwarded to devices that would terminate such messages as result of not knowing how to process them.

2.5. Appliance Types

SFC OAM must provide tools that operate through various types of appliances including:

- . Transparent appliances: These appliances typically do not make any modifications to the packet. In such cases, the SFF may be able to process OAM messages.
- . Appliances that modify the packet: These appliances modify packet fields. Certain appliances may modify only the headers corresponding to the network over which it is transported, e.g. the MAC headers or overlay headers. In other cases, the IP

header of the application's packet may be modified, e.g. NAT. In yet other cases, the application session itself may be terminated and a new session initiated, e.g. a load balancer that offers HTTPS termination.

In general, it should be possible to allow or disallow having a given SF operate on an OAM packet in the same way that it would on a regular data packet, but with the awareness that it is operating on an OAM packet. It is essential to recognize the OAM message so that its status (as an OAM message) can be preserved as it is processed through the normal data path.

3. IANA Considerations

This draft does not have any IANA considerations.

4. Security Considerations

TBD

5. Acknowledgements

6. References

6.1. Normative References

6.2. Informative References

[RFC 2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," March 1997.

[RFC 6291] Andersson, L. et al., "Guidelines for the Use of the "OAM" Acronym in the IETF," June 2011

[RFC 7276] Mizrahi, T. et al., "An Overview of Operations, Administration, and Maintenance (OAM) Tools," June 2014

[NVO3 OAM] Senevirathne, T., "NVO3 Fault Management," https://datatracker.ietf.org/doc/draft-tissa-nvo3-oam-fm/?include_text=1, August 2014

[STEALTH FIREWALL] Brandon Gillespie "Stealth firewalls", <http://www.giac.org/paper/gsec/629/stealth-firewalls/101440>

[SCH draft] Quinn, P. et al., "Network Service Header," <https://datatracker.ietf.org/doc/draft-quinn-sfc-nsh/>, February 2014

Authors' Addresses

Ram Krishnan
Brocade Communications
ramk@brocade.com

Anoop Ghanwani
Dell
anoop@alumni.duke.edu

Pedro A. Aranda Gutierrez
Telefonica I+D
Don Ramon de la Cruz, 82
Madrid, 28006, Spain
+34 913 129 041
pedroa.aranda@tid.es

Diego Lopez
Telefonica I+D
Don Ramon de la Cruz, 82
Madrid, 28006, Spain
+34 913 129 041
diego@tid.es

Joel Halpern
Ericsson
joel.halpern@ericsson.com

Sriganesh Kini
Ericsson
Sriganesh.kini@ericsson.com

Andy Reid
BT
andy.bd.reid@bt.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: March 21, 2015

W. Liu, Ed.
H. Li
O. Huang
Huawei Technologies
M. Boucadair, Ed.
France Telecom
N. Leymann
Deutsche Telekom AG
Q. Fu
China Mobile
Q. Sun
China Telecom
C. Pham
Telstra Corporation
C. Huang
Carleton University
J. Zhu
Huawei Technologies
P. He
Ciena Corp
September 17, 2014

Service Function Chaining (SFC) General Use Cases
draft-liu-sfc-use-cases-08

Abstract

The delivery of value-added services relies on the invocation of advanced Service Functions in a sequential order. This mechanism is called Service Function Chaining (SFC). The set of involved Service Functions and their order depends on the service context and other deployment-specific considerations.

Having a single use case document eases the effort of deriving requirements that are to be met by SFC solution(s). Moreover, it allows to identify commonalities between the various use cases that are of interest. This document presents a set of general use cases of 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 <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 March 21, 2015.

Copyright Notice

Copyright (c) 2014 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. Terminology	4
3. Service Function Chaining Use Cases	5
3.1. Service Function Chain in Fixed Broadband Networks	5
3.2. Service Function Chain in Mobile Networks: Brief Overview	6
3.3. Common Service Chain Network: A Convergence Tool	7
3.4. Distributed Service Function Chain	8
3.5. Service Function Chain in Data Center	9
3.6. Service Function Chain in Cloud CPE	9
4. Abstraction of SFC in Different Deployment Scenarios	10
4.1. Per-service characteristic SFC	11
4.2. Per-user/subscription requirement SFC	12
4.3. TE-Oriented SFC	12
4.4. SFC for Bi-directional Flow	12
4.5. SFC over Multiple Underlay Networks	13
4.6. SFC over Service Path Forking	14
4.7. Recursive SFC	15
5. Security Considerations	16
6. Acknowledgements	16
7. Informative References	16
Authors' Addresses	17

1. Introduction

The delivery of Value-Added Services (VAS) relies on the invocation of various Service Functions (SFs). Typically, the traffic is forwarded through a set of Network Elements embedding Service Functions, e.g.:

- a. Direct a portion of the traffic to a Network Element for monitoring and charging purposes.
- b. Before sending traffic to DC servers, steer the traffic to cross a load balancer to distribute the traffic load among several links, Network Elements, etc.
- c. Mobile network operators split mobile broadband traffic and steer them along an offloading path.
- d. Use a firewall to filter the traffic for IDS (Intrusion Detection System)/IPS (Intrusion Protection System).
- e. Use a security gateway to encrypt/decrypt the traffic. SSL offloading function can also be enabled.
- f. If the traffic has to traverse different networks supporting distinct address families, for example IPv4/IPv6, direct the traffic to a CGN (Carrier Grade NAT, [RFC6888][RFC6674]) or NAT64 [RFC6146].
- g. Some internal service platforms rely on implicit service identification. Dedicated Service Functions are enabled to enrich packets (e.g., HTTP header enrichment) with the identity of the subscriber or the UE (User Equipment).
- h. Operators offer VAS on a per subscription basis. It is desirable to steer traffic only from the subscribers, who have subscribed to VAS, to the relevant service platforms.

Having a single use case document eases the effort of deriving requirements that are to be met by SFC solution(s). Moreover, it allows to identify commonalities between the various use cases that are of interest. This document describes some use cases of Service Function Chaining (SFC). It is not the purpose of this document to be exhaustive, but instead, we try to draw the set of deployments context that are likely to see SFC solutions deployed.

For most of the use cases presented in this document:

- o Instantiated SFC are driven by business and engineering needs.

- o The amount of instantiated SFCs can vary in time, service engineering objectives and service engineering choices.
- o The amount of instantiated SFCs are policy-driven and are local to each administrative entity.
- o The technical characterization of each Service Function is not frozen in time. A Service Function can be upgraded to support new features or disable an existing feature, etc.
- o Some stateful SFs (e.g., NAT or firewall) may need to treat both outgoing and incoming packets. The design of SF Maps must take into account such constraints, otherwise, the service may be disturbed. The set of SFs that need to be invoked for both direction is up to the responsibility of each administrative entity operating an SFC-enabled domain.
- o For subscription-based traffic steering, subscriber-awareness capability is required. A UE is allocated a dynamic IPv4 address and/or IPv6 prefix when attaching to a network. This IPv4 address and/or IPv6 prefix can change from time to time. The requirement is to be able to correlate an IPv4 address and/or IPv6 prefix to a subscriber identity from that will be used to trigger the invocation of some Service Functions.
- o Some Service Functions may be in the same subnet; while others may not. Service Functions are deployed directly on physical hardware, as one or more Virtual Machines, or any combination thereof.

2. Terminology

This document makes use of the terms defined in [I-D.ietf-sfc-architecture].

Service Flow: packets/frames with specific service characteristics (e.g., packets matching a specific tuple of fields in the packet header and/or data) or determined by some service-inferred policies (such as access port and etc.).

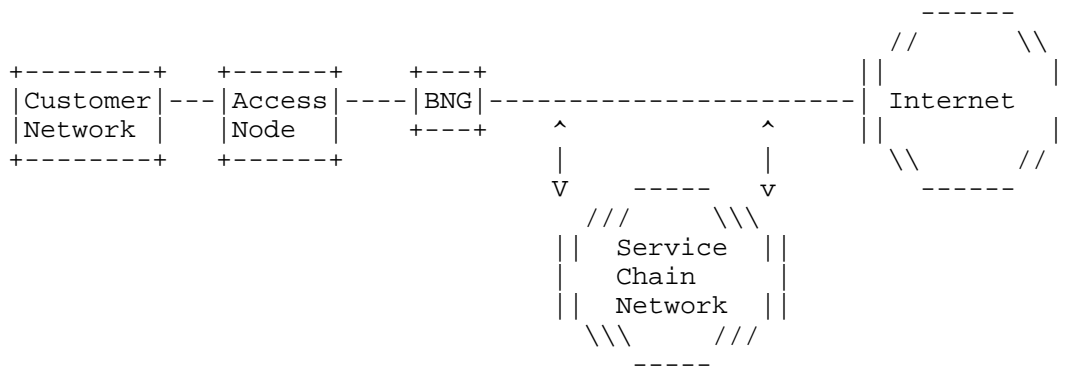
Gi interface: 3GPP defines the Gi interface as the reference point between the GGSN (Gateway GPRS Support Node) and an external PDN (Packet Domain Network). This interface reference point is called SGi in 4G networks (i.e., between the PDN Gateway (PGW) and an external PDN)[RFC6459].

3. Service Function Chaining Use Cases

Service Function Chains can be deployed in a diversity of scenarios such as broadband networks, mobile networks, and DC center. This section describes a set of scenarios for Service Function Chaining deployment. Please note that for each SFC there is a corresponding symmetrical reverse chain, so we added bidirectional links to each SFC use case figure below. For those links that do not have a directional mark, they are bidirectional by default.

3.1. Service Function Chain in Fixed Broadband Networks

In fixed broadband networks, users may be accessed into the network via different technologies, which typically includes DSL, Ethernet and PON. Whatever the access technology is, the architecture for access and metro network is similarly comprised of Access Nodes (ANs) and Broadband Network Gateways (BNGs), where the AN is usually a device providing access to network for customers with variant access technologies and the BNG is the first IP node and providing subscriber authorization, authentication and accounting.



Service Function Chain in Fixed Broadband Networks

Figure 1: An example of Service Function Chain in Broadband Networks

Figure 1 illustrates a fixed broadband network with Service Chaining. Service Chain Network is deployed behind BNG and before Internet. The Service Function Chain in Figure 1 may include several Service Functions to perform services such as DPI, NAT44, DS-Lite, NPTv6, Parental control, Firewall, load balancer, Cache, etc.

The Broadband Forum (BBF) is developing a study document (SD-326) about Flexible Service Chaining, whose scope includes identifying and

documenting use cases relevant to fixed broadband networks. Though SD-326 is an internal project within BBF, the content related to use cases has been communicated to IETF per the following Liaison Statement: Broadband Forum Work on Flexible Service Chaining (SD-326) (<http://datatracker.ietf.org/liaison/1304/>). As BBF is the leading organization on fixed broadband network architectures, this liaison will serve as reference for service chaining use cases applicable in such fixed broadband context. Future liaison statements from BBF may provide additional use cases, and will be referenced here as appropriate.

3.2. Service Function Chain in Mobile Networks: Brief Overview

3GPP defines the Gi interface as the reference point between the GGSN (Gateway GPRS Support Node) and an external PDN (Packet Domain Network) [RFC6459]. This interface reference point is called SGi in 4G networks (i.e., between the PDN Gateway (PGW) and an external PDN) [RFC6459]. Note, there is no standard specification of such reference points (i.e., Gi and SGi) in terms of functions to be located in that segment.

Note: The use cases do not include 3GPP release details. For more information on the 3GPP releases detail, the reader may refer to Section 6.2 of [RFC6459].

Traffic is directed to/from Internet traversing one or more Service Functions. Note, these Service Functions are called "enablers" by some operators. One example of enabler function is a HTTP Header Enrichment Function. There are also other VAS function such as Parental Control or network-based Firewall. Subscribers can opt-in and opt-out to these services anytime using a self-served portal or by calling the Operator's customer service.

In light of current deployments, plenty of Service Functions are enabled in the Gi Interface (e.g., DPI, billing and charging, TCP optimization, web optimization, video optimization, header enrichment, etc.). Some of these Service Functions are co-located on the same device while others are enabled in standalone boxes. In order to fulfill new business needs, especially to offer innovative added-value services, the number of enabled Service Functions in the Gi Interface is still growing. Some of these functions are not needed to be invoked for all services/UEs, e.g.,: TCP optimization function only for TCP flows, HTTP header enrichment only of HTTP traffic, Video optimization function for video flows, IPv6 firewall + NAT64 function for outgoing IPv6 packets, IPv4 firewall + NAT64 function for incoming IPv4 packets, etc..

3GPP has defined Traffic Detection Function (TDF) which implements DPI (detection) functionality along with enforcement and charging of the corresponding detected applications [TS.23203]. TDF resides on Gi/SGi interface.

Note: It was tempting to use TDF and DPI terms interchangeably, but given the diversity of deployments involving DPI modules the text uses DPI to refer to legacy deployments. The behavior of such DPI modules is deployment-specific.

Several (S)Gi Interfaces can be deployed within the same PLMN (Public Land Mobile Network). This depends mainly on the number of PDNs and other factors. Each of these interfaces may involve a differentiated set of Service Functions to be involved.

More details about SFC usage within Mobile Networks can be found in [I-D.ietf-sfc-use-case-mobility].

3.3. Common Service Chain Network: A Convergence Tool

From previous two use cases, we can see commonalities in service chaining. Even though fixed and mobile broadband networks are deployed separately, for integrated operators that running both networks it is obviously beneficial to provide service chaining to both networks from a common service chain network.

In addition to resource optimisation, a common service chain network can also enable seamless service switchover from one network to the other. For example, a customer is watching football game on his mobile phone via 3G network. After he arrives home, he can switch over to the WLAN on his home gateway, which is backhauled to the network by Fiber To The Home (FTTH, a typical PON service), 100 Mbps broadband access. In the case, it is easier to provide seamless service from a common service chain network.

SFC can be used as a tool to better address convergence needs (including adjust the service delivery to the access network constraints or to the capabilities of connected devices).

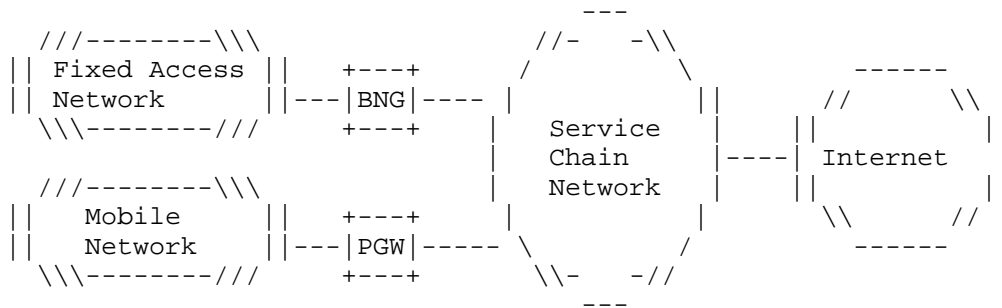


Figure 2: Common Service Chain Network

Figure 2 illustrates a common Service Chain Network is shared by both fixed and mobile broadband networks. Such a common service chain network can be deployed as consisting of only network nodes with specific functions, or in a data center. In both cases, service nodes, whether physical or virtual, are shared by both wireline and wireless networks. Operators manage service chains universally for both networks and traffic from both networks may go through the same service chain.

3.4. Distributed Service Function Chain

Besides the deployment use cases listed above, a Service Function Chain is not necessarily implemented in a single location but can also be distributed crossing several portions of the network (e.g., data centers) or even using a Service Function that is located at a network element close to the customer (e.g. certain security functions).

Multiple SFC-enabled domains can be enabled in the same administrative domain.

For steering traffic to subscription-based Service Functions, the SFC Classifier needs to understand which subscriber a flow belong to in order to retrieve the service profile to apply to this flow. In some contexts, it is not possible to identify in a permanent manner the subscriber by the source IP address because that IP address may be assigned dynamically. Out-of-band methods to correlate the source IP address and a subscriber identifier may be needed in a given administrative domain. The SFC Classifier can rely on pull or push methods to correlate an IP address and/or IPv6 prefix to a subscriber identity. Examples are querying the PCRF or receiving RADIUS Accounting messages respectively.

For steering traffic to traffic management Service Functions such as video optimisation platform, in mobile network, it is desirable to perform optimisation on when required. That is when there is congestion in the Radio cells. One option for the SFC Classifier to have this congestion-awareness is for the network to provide this information to the SFC Classifier, directly, or via an intermediate actionable-intelligence function, which can combine other inputs or policies. How those policies and feedback data are configured to the SFC Classifier may be specific to each administrative domain.

3.5. Service Function Chain in Data Center

In DC (Data Center), like in broadband and mobile networks, Service Function Chains may also be deployed to provide added-value services.

Figure 3 illustrates a possible scenario for Service Function Chain in Data Center: SFs are located between the DC Router (access router) and the Servers. From Servers to Internet, there are multiple Service Functions such as IDS/IPS, FW, NAT lined up and a monolithic SFC created for all incoming traffic.

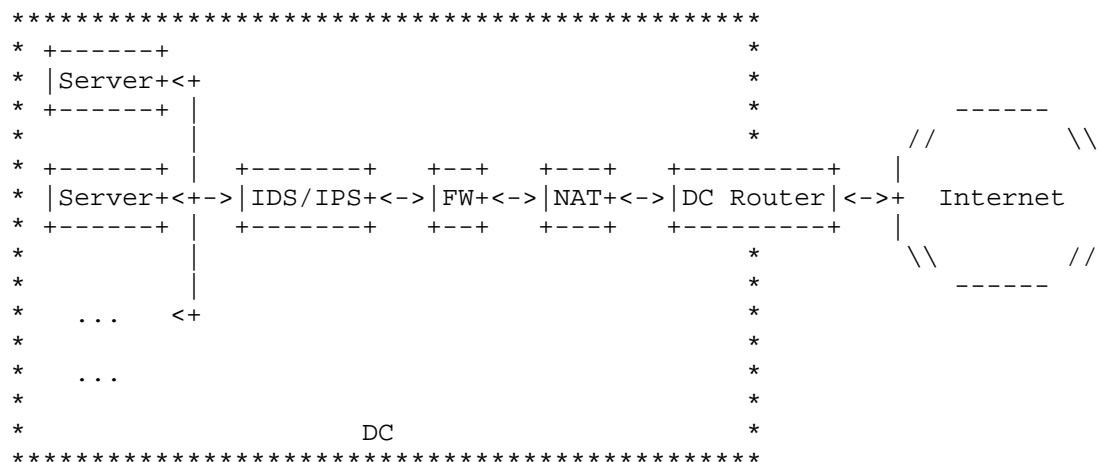


Figure 3: Service Function Chain in Data Center

More details about Service Function Chain in Data Center can be found in [I-D.ietf-sfc-dc-use-cases].

3.6. Service Function Chain in Cloud CPE

Cloud CPE is one deployment scenario where the value-added service functions are centralized (e.g., hosted in the network or cloud side), leaving the subscriber side box with basic L2/L3

functionalities. In this scenario, all the value added services are configured by subscribers and enabled in the network side.

Subscribers can define their own added value services. The Cloud CPE will translate those services requests into chains of Service Functions. Such architecture must support means to differentiate subscribers and their traffic.

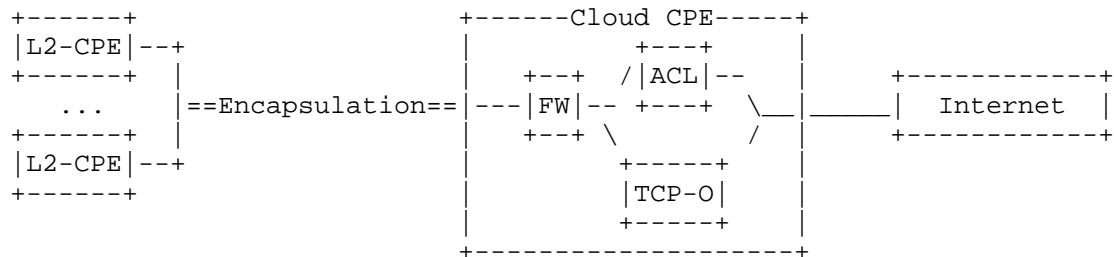


Figure 4: SFC in Cloud CPE

4. Abstraction of SFC in Different Deployment Scenarios

This section presents the SFC scenarios from a different angle, i.e., the abstraction of SFC use cases in different deployment scenarios. Each of the use case may belong to one or many of the categories listed below:

Category	Description
Per-service Characteristic SFC	Chain different Service Functions based on service/application characteristics
Per-user/subscription SFC	Chain different Service Functions based on user requirements or subscription information. Note, this does not mean that millions of SFCs will be instantiated but SF classification is subscriber-aware.
TE-Oriented SF	Chain different Service Functions for Traffic Engineering purposes. This may includes load, utilisation, planned maintenance, etc.
Bi-directional Flow SFC	Function path that contain bi-directional Service Functions
SFC over Multi-Underlay Networks	Service Functions distributed over different underlay networks
SFC over Service Functions Forking	SFC that contains the paths for different service or applications

4.1. Per-service characteristic SFC

The traffic in a network is usually forwarded based on destination IP or MAC addresses. In an operator's network, some Service Functions are implemented, where traffic is steered through these Service Functions in a certain sequence according to service characteristics and objectives.

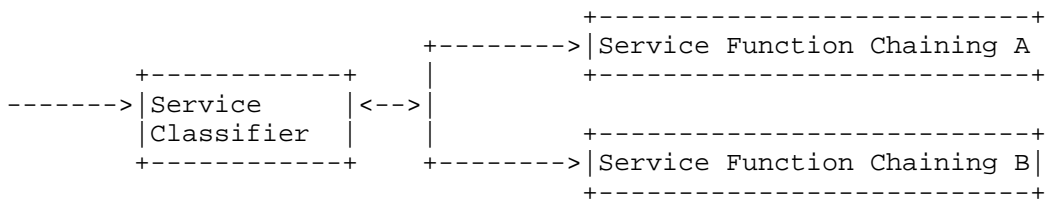


Figure 5: General Service Function Chain

Traffic enters a SFC-enabled domain in a service classifier, which identifies traffic and classifies it into service flows. Service flows are forwarded on a per SF Map basis.

4.2. Per-user/subscription requirement SFC

In operator networks with user subscription information, it is considered as a value added service to provide different subscribers with differentiated services. Subscribers may subscribe different services and the order handling at the operator side will translate those subscription request into configuration operations so that the service will be appropriately delivered to the subscribers. Configuration operations include in particular the provisioning of classification rules.

4.3. TE-Oriented SFC

TE-oriented SFC is required by operators in achieving flexible service operating. For example, if certain paths are congested or certain Service Functions are overloaded, SFC forwarding should be inferred accordingly.

4.4. SFC for Bi-directional Flow

Some Service Functions, for example, NAT or TCP optimization, need to handle bi-directional flows, while others SFs such as video optimization don't need to handle bi-directional flows.

Due to IPv4 address exhaustion, more and more operators have deployed or are about to deploy IPv6 transition technologies such as NAT64 [RFC6146]. The traffic traversing a NAT64 function may go through different types of IP address domains. One key feature of this scenario is that characteristics of packets before and after processed by the service processing function are different, e.g., from IPv6 to IPv4. The unpredictability of processed packets, due to the algorithm in the Service Function, brings difficulties in steering the traffic.

A variety of hosts can be connected to the same network: IPv4-only, dual-stack, and IPv6-only. A differentiated forwarding path can be envisaged for each of these hosts. In particular, DS hosts should not be provided with a DNS64, and as such there traffic should not be delivered to a NAT64 device. Means to guide such differentiated path can be implemented at the host side; but may also be enabled in the network side as well.

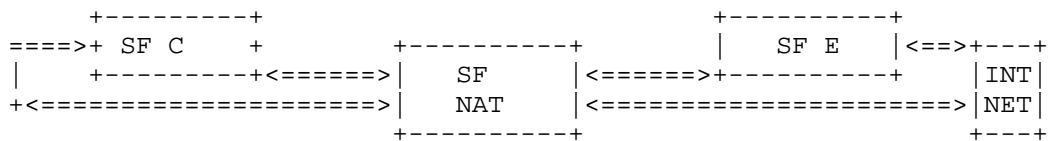


Figure 6: Service Function Chain with NAT64 function

Figure 6 shows a specific example of Service Function Chain with NAT function. Service flow1 is processed by SF(Service Function) C, NAT and E sequentially. In this example, the SF NAT performs NAT64. As a result, packets after processing by the SF NAT are in IPv4, which is a different version of IP header from the packets before processing. Service Function Chaining in this scenario should be able to identify the flow even if it is changed after processed by Service Functions.

4.5. SFC over Multiple Underlay Networks

Operators may need to deploy their networks with various types of underlay technologies. Therefore, Service Function Chaining needs to support different types of underlay networks.

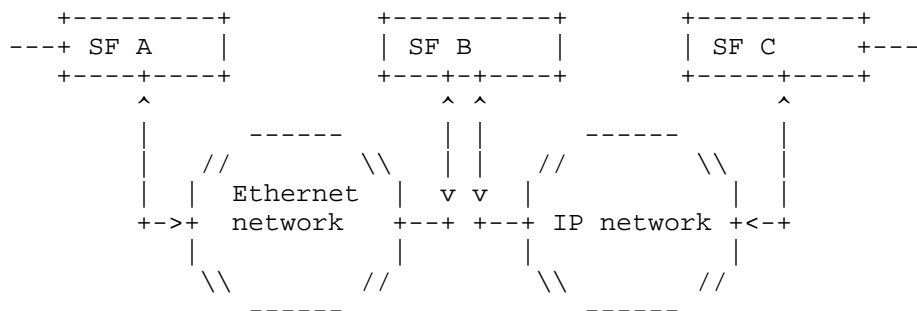


Figure 7: multiple underlay networks: Ethernet and IP

Figure 7 illustrates an example of Ethernet and IP network, very common and easy for deployment based on existing network status, as underlay networks. SF(Service Functions) A, B and C locate in Ethernet and IP networks respectively. To build a Service Function Chain of SF A, B and C, Service Function Chaining needs to support steering traffic across Ethernet and IP underlay networks.

4.6. SFC over Service Path Forking

To enable service or content awareness, operators need DPI functions to look into packets. When a DPI function is part of a Service Function Chain, packets processed by the DPI function may be directed to different paths according to result of DPI processing. That means a forking service path.

In this use case, the switching SF is another classifier which need to classify flow and shepherd them to different paths.

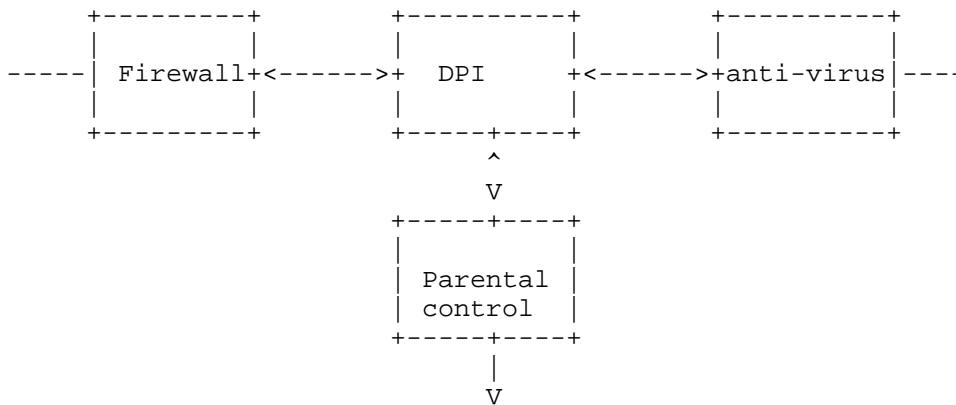


Figure 8: a forking service path

Figure 8 shows the use case of a forking service path. Traffic first goes through a firewall and then arrives at DPI function which discerns virus risk. If a certain pre-configured pattern is matched, the traffic is directed to an anti-virus function.

Such DPI function may fork out more than one path.

Service function sharing is sub-category of the service function forking. Some carrier grade hardware box or Service Functions running on high performance servers can be shared to support multiple Service Function Chains. Following is an example.

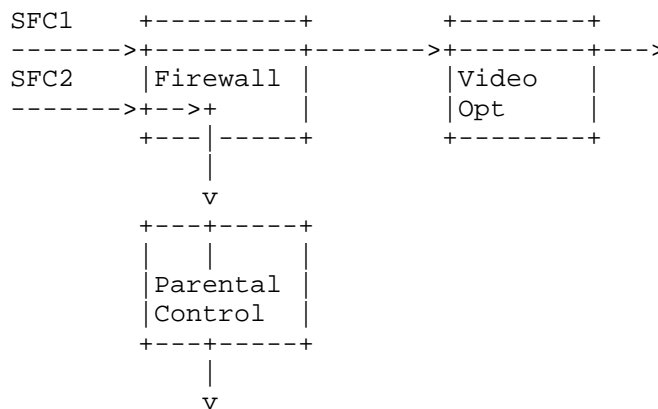


Figure 9: Two Service Function Chains share one Service Function

In Figure 9, there are three Service Functions, firewall, VideoOpt and Parental Control, and two Service Functions Chains SFC1 and SFC2. SFC1 serves broadband user group1 which subscribes to secure web surfing and Internet video optimization, while SFC2 serves broadband user group2 which subscribes to secure web surfing with parental control. SF Firewall is shared by both Service Function Chains.

4.7. Recursive SFC

SFCs can be provided in a recursive manner. A Level 1 service provider can sell SFC services to multiple clients. Each client can further partition its SFC and serve as a Level 2 service provider to sell differentiated SFCs to different clients. This process can continue several iterations making recursive service a new business model which is becoming popular today.

Consider a use case where an enterprise leases a virtual service network from a data center provider. The enterprise then creates two service chains out of the virtual service network. The first service chain, designed for its employees, will force traffic flows to go through NAT, DPI, firewall, LB, and various servers. The second one, designed for its customers, will only go through NAT and web servers. Its customers can create specific websites for different departments such as purchase department, service department, etc.

An important characteristic of recursive service is that each service provider is a separate entity who owns the SFC it purchased from lower level provider and who also decides the SFCs it creates for its clients.

5. Security Considerations

This document does not define an architecture nor a protocol. It focuses on listing use cases and typical Service Function examples. Some of these functions are security-related.

SFC-related security considerations are discussed in [I-D.ietf-sfc-architecture].

6. Acknowledgements

Jie Hu and Zhen Cao contributed to an earlier version of this document.

This document has benefited from reviews, suggestions, comments and proposed text provided by the following members of the Service Function Chaining Working Group (sfc WG), listed in alphabetical order: David Binet, Hui Deng, Alla Goldner, Yuanlong Jiang, Jerome Moisand, Lehong Niu, Ron Parker, and Lucy Yong.

7. Informative References

- [I-D.ietf-sfc-architecture]
Halpern, J. and C. Pignataro, "Service Function Chaining (SFC) Architecture", draft-ietf-sfc-architecture-01 (work in progress), September 2014.
- [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-01 (work in progress), July 2014.
- [I-D.ietf-sfc-use-case-mobility]
Haeffner, W., Napper, J., Stiemerling, M., Lopez, D., and J. Uttaro, "Service Function Chaining Use Cases in Mobile Networks", draft-ietf-sfc-use-case-mobility-01 (work in progress), July 2014.
- [RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.
- [RFC6459] Korhonen, J., Soininen, J., Patil, B., Savolainen, T., Bajko, G., and K. Iisakkila, "IPv6 in 3rd Generation Partnership Project (3GPP) Evolved Packet System (EPS)", RFC 6459, January 2012.

- [RFC6674] Brockners, F., Gundavelli, S., Speicher, S., and D. Ward, "Gateway-Initiated Dual-Stack Lite Deployment", RFC 6674, July 2012.
- [RFC6888] Perreault, S., Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida, "Common Requirements for Carrier-Grade NATs (CGNs)", BCP 127, RFC 6888, April 2013.
- [TS.23203] 3GPP, "Policy and charging control architecture", December 2013.

Authors' Addresses

Will(Shucheng) Liu (editor)
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: liushucheng@huawei.com

Hongyu Li
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: hongyu.li@huawei.com

Oliver Huang
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: oliver.huang@huawei.com

Mohamed Boucadair (editor)
France Telecom
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Nicolai Leymann
Deutsche Telekom AG

Email: n.leymann@telekom.de

Qiao Fu
China Mobile

Email: fuqiao@chinamobile.com

Qiong Sun
China Telecom
No.118 Xizhimennei street, Xicheng District
Beijing 100035
P.R. China

Email: sunqiong@ctbri.com.cn

Chuong Pham
Telstra Corporation
Level 8, 18 Smith Street
Parramatta 2150
Australia

Email: Pham, Chuong D <Chuong.D.Pham@team.telstra.com>

Changcheng Huang
Carleton University
1125 Colonel By Drive
Ottawa ON K1S 5B6
Canada

Email: huang@sce.carleton.ca

Jiafeng Zhu
Huawei Technologies
Santa Clara, CA
US

Email: Jiafeng.zhu@huawei.com

Peng He
Ciena Corp

Email: phe@ciena.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 23, 2015

J. Halpern, Ed.
Ericsson
C. Pignataro, Ed.
Cisco
August 22, 2014

Service Function Chaining (SFC) Architecture
draft-merged-sfc-architecture-02

Abstract

This document describes an architecture for the specification, creation, and ongoing maintenance of Service Function Chains (SFC) in a network. It includes architectural concepts, principles, and components used in the construction of composite services through deployment of SFCs. This document does not propose solutions, protocols, or extensions to existing protocols.

Status of This Memo

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

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

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

This Internet-Draft will expire on February 23, 2015.

Copyright Notice

Copyright (c) 2014 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	2
1.1. Scope	3
1.2. Assumptions	3
1.3. Definition of Terms	4
2. Architectural Concepts	6
2.1. Service Function Chains	6
2.2. Service Function Chain Symmetry	7
2.3. Service Function Paths	8
3. Architecture Principles	9
4. Core SFC Architecture Components	10
4.1. SFC Encapsulation	11
4.2. Service Function (SF)	12
4.3. Service Function Forwarder (SFF)	12
4.3.1. Transport Derived SFF	14
4.4. SFC-Enabled Domain	14
4.5. Network Overlay and Network Components	14
4.6. SFC Proxy	14
4.7. Classification	16
4.8. Re-Classification and Branching	16
4.9. Shared Metadata	17
5. Additional Architectural Concepts	17
5.1. The Role of Policy	17
5.2. SFC Control Plane	18
5.3. Resource Control	19
5.4. Infinite Loop Detection and Avoidance	19
5.5. Load Balancing Considerations	20
5.6. MTU and Fragmentation Considerations	21
5.7. SFC OAM	21
5.8. Resilience and Redundancy	22
6. Security Considerations	23
7. Contributors and Acknowledgments	23
8. IANA Considerations	25
9. References	25
9.1. Normative References	25
9.2. Informative References	25
Authors' Addresses	25

1. Introduction

This document describes an architecture used for the creation and ongoing maintenance of Service Function Chains (SFC) in a network. It includes architectural concepts, principles, and components.

An overview of the issues associated with the deployment of end-to-end service function chains, abstract sets of service functions and their ordering constraints that create a composite service and the subsequent "steering" of traffic flows through said service functions, is described in [I-D.ietf-sfc-problem-statement].

This architecture presents a model addressing the problematic aspects of existing service deployments, including topological independence and configuration complexity.

Service function chains enable composite services that are constructed from one or more service functions.

1.1. Scope

This document defines a framework to realize Service Function Chaining (SFC) with minimum requirements on the physical topology of the network. The proposed solution relies on initial packet classification. Packets are initially classified at the entry point of an SFC-enabled domain, and are then forwarded according to the ordered set of Service Functions (SFs) that need to be enabled to process these packets in the SFC-enabled domain.

This document does not make any assumption on the deployment context. The proposed framework covers both fixed and mobile networks.

The architecture described herein is assumed to be applicable to a single network administrative domain. While it is possible for the architectural principles and components to be applied to inter-domain SFCs, these are left for future study.

1.2. Assumptions

The following assumptions are made:

- o Not all SFs can be characterized with a standard definition in terms of technical description, detailed specification, configuration, etc.
- o There is no global or standard list of SFs enabled in a given administrative domain. The set of SFs varies as a function of the service to be provided and according to the networking environment.
- o There is no global or standard SF chaining logic. The ordered set of SFs that needs to be enabled to deliver a given service is specific to each administrative entity.

- o The chaining of SFs and the criteria to invoke them are specific to each administrative entity that operates an SF-enabled domain.
- o Several SF chaining policies can be simultaneously applied within an administrative domain to meet various business requirements.
- o No assumption is made on how FIBs and RIBs of involved nodes are populated.
- o How to bind traffic to a given SF chain is policy-based.

1.3. Definition of Terms

Network Service: An offering provided by an operator that is delivered using one or more service functions. This may also be referred to as a composite service. The term "service" is used to denote a "network service" in the context of this document.

Note: Beyond this document, the term "service" is overloaded with varying definitions. For example, to some a service is an offering composed of several elements within the operator's network, whereas for others a service, or more specifically a network service, is a discrete element such as a firewall. Traditionally, such services (in the latter sense) host a set of service functions and have a network locator where the service is hosted.

SFC Encapsulation: The SFC Encapsulation provides at a minimum SFP identification, and is used by the SFC-aware functions, such as the SFF and SFC-aware SFs. The SFC Encapsulation is not used for network packet forwarding. In addition to SFP identification, the SFC encapsulation carries dataplane context information, also referred to as metadata.

Classification: Locally instantiated policy and customer/network/service profile matching of traffic flows for identification of appropriate outbound forwarding actions.

Classifier: An element that performs Classification.

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

occurrences of the Service Function can exist in the same administrative domain.

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

An SF may be SFC encapsulation aware, that is it receives and acts on information in the SFC encapsulation, or unaware, in which case data forwarded to the SF does not contain the SFC encapsulation.

Service Function Forwarder (SFF): A service function forwarder is responsible for delivering traffic received from the network to one or more connected service functions according to information carried in the SFC encapsulation.

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

Service Function Path (SFP): The SFP provides a level of indirection between the fully abstract notion of service chain as an abstract sequence of functions to be delivered, and the fully specified notion of exactly which SFF/SFs the packet will visit when it actually traverses the network. By allowing the control components to specify this level of indirection, the operator may control the degree of SFF/SF selection authority that is delegated to the network.

Rendered Service Path (RSP): The Service Function Path is a constrained specification of where packets using a certain service chain must go. While it may be so constrained as to identify the exact locations, it can also be less specific. Packets themselves are of course transmitted from and to specific places in the network, visiting a specific sequence of SFFs and SFs. This sequence of actual visits by a packet to specific SFFs and SFs in the network is known as the Rendered

Service Path (RSP). This definition is included here for use by later documents, such as when solutions may need to discuss the actual sequence of locations the packets visit.

SFC-enabled Domain: A network or region of a network that implements SFC. An SFC-enabled Domain is limited to a single network administrative domain.

SFC Proxy: Removes and inserts SFC encapsulation on behalf of an SFC-unaware service function. SFC proxies are logical elements.

2. Architectural Concepts

The following sections describe the foundational concepts of service function chaining and the SFC architecture.

Service Function Chaining enables the creation of composite (network) services that consist of an ordered set of Service Functions (SF) that must be applied to packets and/or frames selected as a result of classification. Each SF is referenced using an identifier that is unique within an SF-enabled domain. No IANA registry is required to store the identity of SFs.

Service Function Chaining is a concept that provides for more than just the application of an ordered set of SFs to selected traffic; rather, it describes a method for deploying SFs in a way that enables dynamic ordering and topological independence of those SFs as well as the exchange of metadata between participating entities.

2.1. Service Function Chains

In most networks services are constructed as abstract sequences of SFs that represent SFCs. At a high level, an SFC is an abstracted view of a service that specifies the set of required SFs as well as the order in which they must be executed. Graphs, as illustrated in Figure 1, define each SFC. A given SF can be part of zero, one, or many SFCs. A given SF can appear one time or multiple times in a given SFC.

SFCs can start from the origination point of the service function graph (i.e.: node 1 in Figure 1), or from any subsequent node in the graph. SFs may therefore become branching nodes in the graph, with those SFs selecting edges that move traffic to one or more branches. An SFC can have more than one terminus.

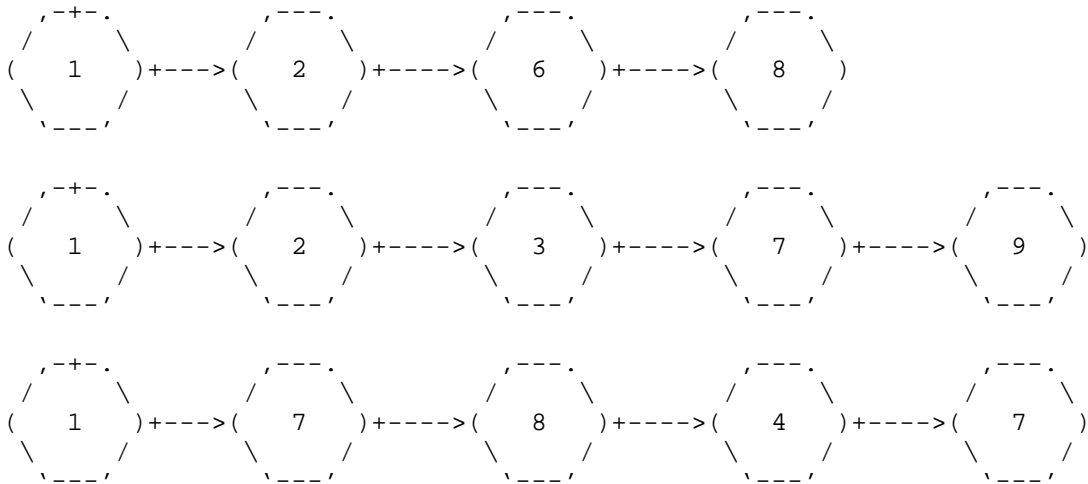


Figure 1: Service Function Chain Graphs

2.2. Service Function Chain Symmetry

SFCs may be unidirectional or bidirectional. A unidirectional SFC requires that traffic be forwarded through the ordered SFs in one direction (SF1 -> SF2 -> SF3), whereas a bidirectional SFC requires a symmetric path (SF1 -> SF2 -> SF3 and SF3 -> SF2 -> SF1), and in which the SF instances are the same in opposite directions. A hybrid SFC has attributes of both unidirectional and bidirectional SFCs; that is to say some SFs require symmetric traffic, whereas other SFs do not process reverse traffic or are independent of the corresponding forward traffic.

SFCs may contain cycles; that is traffic may need to traverse one or more SFs within an SFC more than once. Solutions will need to ensure suitable disambiguation for such situations.

The architectural allowance that is made for SFPs that delegate choice to the network for which SFs or SFFs a packet will visit creates potential issues here. A solution that allows such delegation needs to also describe how the solution ensures that those service chains that require service function chain symmetry can achieve that.

Further, there are state tradeoffs in symmetry. Symmetry may be realized in several ways depending on the SFF and classifier functionality. In some cases, "mirrored" classification (S -> D and D -> S) policy may be deployed, whereas in others shared state between classifiers may be used to ensure that symmetric flows are

correctly identified, then steered along the required SFP. At a high level, there are various common cases. In a non-exhaustive way, there can be for example: a single classifier (or a small number of classifiers), in which case both incoming and outgoing flows could be recognized at the same classifier, so the synchronization would be feasible by internal mechanisms internal to the classifier. Another case is the one of stateful classifiers where several classifiers may be clustered and share state. Lastly, another case entails fully distributed classifiers, where synchronization needs to be provided through unspecified means. This is a non-comprehensive list of common cases.

2.3. Service Function Paths

A service function path (SFP) is a mechanism used by service chaining to express the result of applying more granular policy and operational constraints to the abstract requirements of a service chain (SFC). This architecture does not mandate the degree of specificity of the SFP. Architecturally, within the same SFC-enabled domain, some SFPs may be fully specified, selecting exactly which SFF and which SF are to be visited by packets using that SFP, while other SFPs may be quite vague, deferring to the SFF the decisions about the exact sequence of steps to be used to realize the SFC. The specificity may be anywhere in between these extremes.

As an example of such an intermediate specificity, there may be two SFPs associated with a given SFC, where one SFP says essentially that any order of SFF and SF may be used as long as it is within data center 1, and where the second SFP allows the same latitude, but only within data center 2.

Thus, the policies and logic of SFP selection or creation (depending upon the solution) produce what may be thought of as a constrained version of the original SFC. Since multiple policies may apply to different traffic that uses the same SFC, it also follows that there may be multiple SFPs may be associated with a single SFC.

The architecture allows for the same SF to be reachable through multiple SFFs. In these cases, some SFPs may constrain which SFF is used to reach which SF, while some SFPs may leave that decision to the SFF itself.

Further, the architecture allows for two or more SFs to be attached to the same SFF, and possibly connected via internal means allowing more effective communication. In these cases, some solutions or deployments may choose to use some form of internal inter-process or inter-VM messaging (communication behind the virtual switching element) that is optimized for such an environment. This must be

coordinated with the SFF so that the service function forwarding can properly perform its job. Implementation details of such mechanisms are considered out of scope for this document, and can include a spectrum of methods: for example situations including all next-hops explicitly, others where a list of possible next-hops is provided and the selection is local, or cases with just an identifier, where all resolution is local.

This architecture also allows the same SF to be part of multiple SFPs.

3. Architecture Principles

Service function chaining is predicated on several key architectural principles:

1. Topological independence: no changes to the underlay network forwarding topology - implicit, or explicit - are needed to deploy and invoke SFs or SFCs.
2. Plane separation: dynamic realization of SFPs is separated from packet handling operations (e.g., packet forwarding).
3. Classification: traffic that satisfies classification rules is forwarded according to a specific SFP. For example, classification can be as simple as an explicit forwarding entry that forwards all traffic from one address into the SFP. Multiple classification points are possible within an SFC (i.e. forming a service graph) thus enabling changes/updates to the SFC by SFs.

Classification can occur at varying degrees of granularity; for example, classification can use a 5-tuple, a transport port or set of ports, part of the packet payload, or it can come from external systems.

4. Shared Metadata: Metadata/context data can be shared amongst SFs and classifiers, between SFs, and between external systems and SFs (e.g., orchestration).

Generally speaking, metadata can be thought of as providing and sharing the result of classification (that occurs within the SFC-enabled domain, or external to it) along an SFP. For example, an external repository might provide user/subscriber information to a service chain classifier. This classifier could in turn impose that information in the SFC encapsulation for delivery to the requisite SFs. The SFs could in turn utilize the user/subscriber information for local policy decisions.

5. Service definition independence: The SFC architecture does not depend on the details of SFs themselves. Additionally, no IANA registry is required to store the list of SFs.
 6. Service function chain independence: The creation, modification, or deletion of an SFC has no impact on other SFCs. The same is true for SFPs.
 7. Heterogeneous control/policy points: The architecture allows SFs to use independent mechanisms (out of scope for this document) to populate and resolve local policy and (if needed) local classification criteria.
4. Core SFC Architecture Components

At a very high level, the logical architecture of an SFC-enabled Domain comprises:

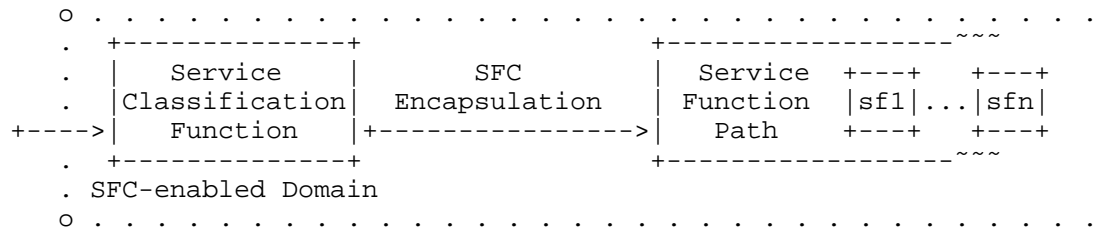


Figure 2: Service Function Chain Architecture

The following sub-sections provide details on each logical component that form the basis of the SFC architecture. A detailed overview of how each of these architectural components interact is provided in Figure 3:

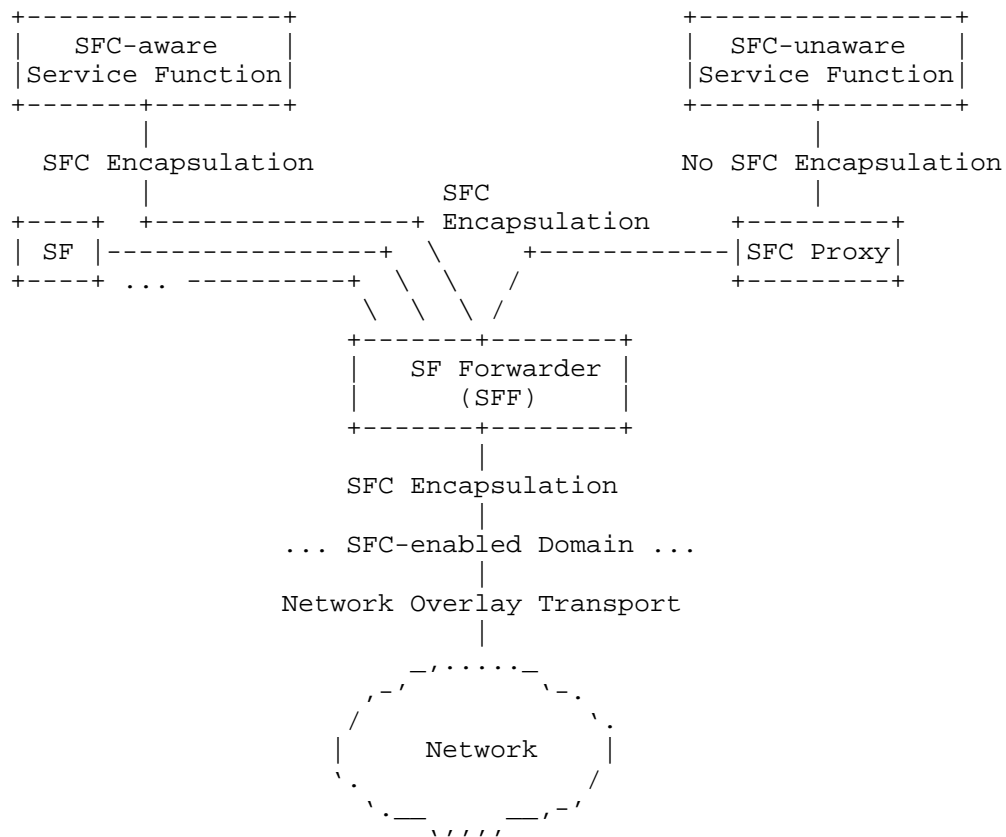


Figure 3: Service Function Chain Architecture Components

4.1. SFC Encapsulation

The SFC encapsulation enables service function path selection. It also enables the sharing of metadata/context information when such metadata exchange is required.

The SFC encapsulation provides explicit information used to identify the SFP. However, the SFC encapsulation is not a transport encapsulation itself: it is not used to forward packets within the network fabric. If packets need to flow between separate physical platforms, the SFC encapsulation therefore relies on an outer network transport. Transit forwarders -- such as router and switches -- simply forward SFC encapsulated packets based on the outer (non-SFC) encapsulation.

One of the key architecture principles of SFC is that the SFC encapsulation remain transport independent. As such any network transport protocol may be used to carry the SFC encapsulated traffic.

4.2. Service Function (SF)

The concept of an SF evolves; rather than being viewed as a bump in the wire, an SF becomes a resource within a specified administrative domain that is available for consumption as part of a composite service. SFs send/receive data to/from one or more SFFs. SFC-aware SFs receive this traffic with the SFC encapsulation.

While the SFC architecture defines a new encapsulation - the SFC encapsulation - and several logical components for the construction of SFCs, existing SF implementations may not have the capabilities to act upon or fully integrate with the new SFC encapsulation. In order to provide a mechanism for such SFs to participate in the architecture, an SFC proxy function is defined. The SFC proxy acts as a gateway between the SFC encapsulation and SFC-unaware SFs. The integration of SFC-unaware service functions is discussed in more detail in the SFC proxy section.

This architecture allows an SF to be part of multiple SFPs and SFCs.

4.3. Service Function Forwarder (SFF)

The SFF is responsible for forwarding packets and/or frames received from the network to one or more SFs associated with a given SFF using information conveyed in the SFC encapsulation. Traffic from SFs eventually returns to the same SFF, which is responsible for putting it back onto the network.

The collection of SFFs and associated SFs creates a service plane overlay in which SFC-aware SFs, as well as SFC-unaware SFs reside. Within this service plane, the SFF component connects different SFs that form a service function path.

SFFs maintain the requisite SFP forwarding information. SFP forwarding information is associated with a service path identifier that is used to uniquely identify an SFP. The service forwarding state enables an SFF to identify which SFs of a given SFP should be applied, and in what order, as traffic flows through the associated SFP. While there may appear to the SFF to be only one available way to deliver the given SF, there may also be multiple choices allowed by the constraints of the SFP.

If there are multiple choices, the SFF needs to preserve the property that all packets of a given flow are handled the same way, since the SF may well be stateful.

The SFF also has the information to allow it to forward packets to the next SFF after applying local service functions. Again, while there may be only a single choice available, the architecture allows for multiple choices for the next SFF. As with SFs, the solution needs to operate such that the behavior with regard to specific flows (see the Rendered Service Path) is stable. It should be noted that the selection of available SFs and next SFFs may be interwoven when an SFF supports multiple distinct service functions and the same service function is available at multiple SFFs. Solutions need to be clear about what is allowed in these cases.

It should be noted that even when the SFF supports and utilizes multiple choices, the decision as to whether to use flow-specific mechanisms or coarser grained means to ensure that the behavior of specific flows is stable is a matter for specific solutions and specific implementations.

The SFF component has the following primary responsibilities:

1. SFP forwarding : Traffic arrives at an SFF from the network. The SFF determines the appropriate SF the traffic should be forwarded to via information contained in the SFC encapsulation. Post-SF, the traffic is returned to the SFF, and if needed forwarded to another SF associated with that SFF. If there is another non-local (i.e., different SFF) hop in the SFP, the SFF further encapsulates the traffic in the appropriate network transport protocol and delivers it to the network for delivery to the next SFF along the path. Related to this forwarding responsibility, an SFF should be able to interact with metadata.
2. Terminating SFPs : An SFC is completely executed when traffic has traversed all required SFs in a chain. When traffic arrives at the SFF after the last SF has finished processing it, the final SFF knows from the service forwarding state that the SFC is complete. The SFF removes the SFC encapsulation and delivers the packet back to the network for forwarding.
3. Maintaining flow state: In some cases, the SFF may be stateful. It creates flows and stores flow-centric information. This state information may be used for a range of SFP-related tasks such as ensuring consistent treatment of all packets in a given flow, ensuring symmetry or for state-aware SFC Proxy functionality (see Section 4.8).

4.3.1. Transport Derived SFF

Service function forwarding, as described above, directly depends upon the use of the service path information contained in the SFC encapsulation. However, existing implementations may not be able to act on the SFC encapsulation. These platforms may opt to use existing transport information if it can be arranged to provide explicit service path information.

This results in the same architectural behavior and meaning for service function forwarding and service function paths. It is the responsibility of the control components to ensure that the transport path executed in such a case is fully aligned with the path identified by the information in the service chaining encapsulation.

4.4. SFC-Enabled Domain

Specific features may need to be enforced at the boundaries of an SFC-enabled domain, for example to avoid leaking SFC information. Using the term node to refer generically to an entity that is performing a set of functions, in this context, an SFC Boundary Node denotes a node that connects one SFC-enabled domain to a node either located in another SFC-enabled domain or in a domain that is SFC-unaware.

An SFC Boundary node can act as egress or ingress. An SFC Egress Node denotes a SFC Boundary Node that handles traffic leaving the SFC-enabled domain the Egress Node belongs to. Such a node is required to remove any information specific to the SFC Domain, typically the SFC Encapsulation. An SFC Ingress Node denotes an SFC Boundary Node that handles traffic entering the SFC-enabled domain. In most solutions and deployments this will need to include a classifier, and will be responsible for adding the SFC encapsulation to the packet.

4.5. Network Overlay and Network Components

Underneath the SFF there are components responsible for performing the transport (overlay) forwarding. They do not consult the SFC encapsulation or inner payload for performing this forwarding. They only consult the outer-transport encapsulation for the transport (overlay) forwarding.

4.6. SFC Proxy

In order for the SFC architecture to support SFC-unaware SFs (.e.g legacy service functions) a logical SFC proxy function may be used.

This function sits between an SFF and one or more SFs to which the SFF is directing traffic.

The proxy accepts packets from the SFF on behalf of the SF. It removes the SFC encapsulation, and then uses a local attachment circuit to deliver packets to SFC unaware SFs. It also receives packets back from the SF, reapplies the SFC encapsulation, and returns them to the SFF for processing along the service function path.

Thus, from the point of view of the SFF, the SFC proxy appears to be part of an SFC aware SF.

Communication details between the SFF and the SFC Proxy are the same as those between the SFF and an SFC aware SF. The details of that are not part of this architecture. The details of the communication methods over the local attachment circuit between the SFC proxy and the SFC-unaware SF are dependent upon the specific behaviors and capabilities of that SFC-unaware SF, and thus are also out of scope for this architecture.

Specifically, for traffic received from the SFF intended for the SF the proxy is representing, the SFC proxy:

- o Removes the SFC encapsulation from SFC encapsulated packets.
- o Identifies the required SF to be applied based on available information including that carried in the SFC encapsulation.
- o Selects the appropriate outbound local attachment circuit through which the next SF for this SFP is reachable. This is derived from the identification of the SF carried in the SFC encapsulation, and may include local techniques. Examples of a local attachment circuit include, but are not limited to, VLAN, IP-in-IP, L2TPv3, GRE, VXLAN.
- o Forwards the original payload via the selected local attachment circuit to the appropriate SF.

When traffic is returned from the SF:

- o Applies the required SFC encapsulation. The determination of the encapsulation details may be inferred by the local attachment circuit through which the packet and/or frame was received, or via packet classification, or other local policy. In some cases, packet ordering or modification by the SF may necessitate additional classification in order to re-apply the correct SFC encapsulation.

- o Delivers the packet with the SFC Encapsulation to the SFF, as would happen with packets returned from an SFC-aware SF.

Alternatively, a service provider may decide to exclude legacy service functions from an SFC domain.

4.7. Classification

Traffic from the network that satisfies classification criteria is directed into an SFP and forwarded to the requisite service function(s). Classification is handled by a service classification function; initial classification occurs at the ingress to the SFC domain. The granularity of the initial classification is determined by the capabilities of the classifier and the requirements of the SFC policy. For instance, classification might be relatively coarse: all packets from this port are subject to SFC policy X and directed into SFP A, or quite granular: all packets matching this 5-tuple are subject to SFC policy Y and directed into SFP B.

As a consequence of the classification decision, the appropriate SFC encapsulation is imposed on the data, and a suitable SFP is selected or created. Classification results in attaching the traffic to a specific SFP.

4.8. Re-Classification and Branching

The SFC architecture supports re-classification (or non-initial classification) as well. As packets traverse an SFP, re-classification may occur - typically performed by a classification function co-resident with a service function. Reclassification may result in the selection of a new SFP, an update of the associated metadata, or both. This is referred to as "branching".

For example, an initial classification results in the selection of SFP A: DPI_1 --> SLB_8. However, when the DPI service function is executed, attack traffic is detected at the application layer. DPI_1 re-classifies the traffic as attack and alters the service path to SFP B, to include a firewall for policy enforcement: dropping the traffic: DPI_1 --> FW_4. Subsequent to FW_4, surviving traffic would be returned to the original SFF. In this simple example, the DPI service function re-classifies the traffic based on local application layer classification capabilities (that were not available during the initial classification step).

When traffic arrives after being steered through an SFC-unaware SF, the SFC Proxy must perform re-classification of traffic to determine the SFP. The SFC Proxy is concerned with re-attaching information

for SFC-unaware SFs, and a stateful SFC Proxy simplifies such classification to a flow lookup.

4.9. Shared Metadata

Sharing metadata allows the network to provide network-derived information to the SFs, SF-to-SF information exchange and the sharing of service-derived information to the network. Some SFCs may not require metadata exchange. SFC infrastructure enables the exchange of this shared data along the SFP. The shared metadata serves several possible roles within the SFC architecture:

- o Allows elements that typically operate as ships in the night to exchange information.
- o Encodes information about the network and/or data for post-service forwarding.
- o Creates an identifier used for policy binding by SFs.

Context information can be derived in several ways:

- o External sources
- o Network node classification
- o Service function classification

5. Additional Architectural Concepts

There are a number of issues which solutions need to address, and which the architecture informs but does not determine. This section lays out some of those concepts.

5.1. The Role of Policy

Much of the behavior of service chains is driven by operator and per-customer policy. This architecture is structured to isolate the policy interactions from the data plane and control logic.

Specifically, it is assumed that the service chaining control plane creates the service paths. The service chaining data plane is used to deliver the classified packets along the service chains to the intended service functions.

Policy, in contrast, interacts with the system in other places. Policies and policy engines may monitor service functions to decide if additional (or fewer) instances of services are needed. When

applicable, those decisions may in turn result in interactions that direct the control logic to change the SFP placement or packet classification rules.

Similarly, operator service policy, often managed by operational or business support systems (OSS or BSS), will frequently determine what service functions are available. Operator service policies also determine which sequences of functions are valid and are to be used or made available.

The offering of service chains to customers, and the selection of which service chain a customer wishes to use, are driven by a combination of operator and customer policies using appropriate portals in conjunction with the OSS and BSS tools. These selections then drive the service chaining control logic, which in turn establishes the appropriate packet classification rules.

5.2. SFC Control Plane

This is part of the overall architecture but outside the scope of this document.

The SFC control plane is responsible for constructing SFPs, translating SFCs to forwarding paths and propagating path information to participating nodes to achieve requisite forwarding behavior to construct the service overlay. For instance, an SFC construction may be static; selecting exactly which SFFs and which SFs from those SFFs are to be used, or it may be dynamic, allowing the network to perform some or all of the choices of SFF or SF to use to deliver the selected service chain within the constraints represented by the service path.

In the SFC architecture, SFs are resources; the control plane manages and communicates their capabilities, availability and location in fashions suitable for the transport and SFC operations in use. The control plane is also responsible for the creation of the context (see below). The control plane may be distributed (using new or existing control plane protocols), or be centralized, or a combination of the two.

The SFC control plane provides the following functionality:

1. An SFC-enabled domain wide view of all available service function resources as well as the network locators through which they are reachable.
2. Uses SFC policy to construct service function chains, and associated service function paths.

3. Selection of specific SFs for a requested SFC, either statically (using specific SFs) or dynamically (using service explicit SFs at the time of delivering traffic to them).
4. Provides requisite SFC data plane information to the SFC architecture components, most notably the SFF.
5. Allocation of metadata associated with a given SFP and propagation of the metadata to relevant SFs and/or SFC encapsulation-proxies or their respective policy planes.

5.3. Resource Control

The SFC system may be responsible for managing all resources necessary for the SFC components to function. This includes network constraints used to plan and choose network path(s) between service function forwarders, network communication paths between service function forwarders and their attached service functions, characteristics of the nodes themselves such as memory, number of virtual interfaces, routes, and instantiation, configuration, and deletion of SFs.

The SFC system will also be required to reflect policy decisions about resource control, as expressed by other components in the system.

While all of these aspects are part of the overall system, they are beyond the scope of this architecture.

5.4. Infinite Loop Detection and Avoidance

This SFC architecture is predicated on topological independence from the underlying forwarding topology. Consequently, a service topology is created by Service Function Paths or by the local decisions of the Service Function Forwarders based on the constraints expressed in the SFP. Due to the overlay constraints, the packet-forwarding path may need to visit the same SFF multiple times, and in some less common cases may even need to visit the same SF more than once. The Service Chaining solution needs to permit these limited and policy-compliant loops. At the same time, the solutions must ensure that indefinite and unbounded loops cannot be formed, as such would consume unbounded resources without delivering any value.

In other words, this architecture prevents infinite Service Function Loops, even when Service Functions may be invoked multiple times in the same SFP.

5.5. Load Balancing Considerations

Supporting function elasticity and high-availability should not overly complicate SFC or lead to unnecessary scalability problems.

In the simplest case, where there is only a single function in the SPF (the next hop is either the destination address of the flow or the appropriate next hop to that destination), one could argue that there may be no need for SFC.

In the cases where the classifier is separate from the single function or a function at the terminal address may need sub-prefix or per-subscriber metadata, a single SPF exists (the metadata changes but the SPF does not), regardless of the number of potential terminal addresses for the flow. This is the case of the simple load balancer. See Figure 4.

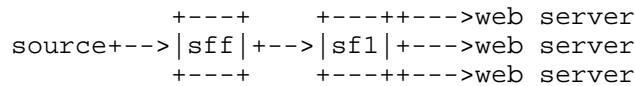


Figure 4: Simple Load Balancing

By extrapolation, in the case where intermediary functions within a chain had similar "elastic" behaviors, we do not need separate chains to account for this behavior - as long as the traffic coalesces to a common next-hop after the point of elasticity.

In Figure 5, we have a chain of five service functions between the traffic source and its destination.

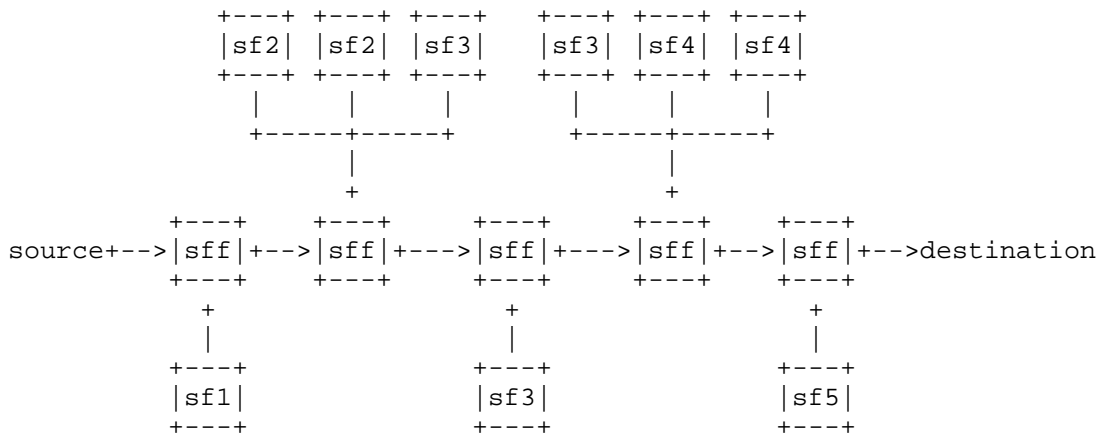


Figure 5: Load Balancing

This would be represented as one service function path: sf1->sf2->sf3->sf4->sf5. The SFF is a logical element, which may be made up of one or multiple components. In this architecture, the SFF may handle load distribution based on policy.

It can also be seen in the above that the same service function may be reachable through multiple SFFs, as discussed earlier. The selection of which SFF to use to reach SF3 may be made by the control logic in defining the SFP, or may be left to the SFFs themselves, depending upon policy, solution, and deployment constraints. In the latter case, it needs to be assured that exactly one SFF takes responsibility to steer traffic through SF3.

5.6. MTU and Fragmentation Considerations

This architecture prescribes additional information being added to packets to identify service function paths and often to represent metadata. It also envisions adding transport information to carry packets along service function paths, at least between service function forwarders. This added information increases the size of the packet to be carried by service chaining. Such additions could potentially increase the packet size beyond the MTU supported on some or all of the media used in the service chaining domain.

Such packet size increases can thus cause operational MTU problems. Requiring fragmentation and reassembly in an SFF would be a major processing increase, and might be impossible with some transports. Expecting service functions to deal with packets fragmented by the SFC function might be onerous even when such fragmentation was possible. Thus, at the very least, solutions need to pay attention to the size cost of their approach. There may be alternative or additional means available, although any solution needs to consider the tradeoffs.

These considerations apply to any generic architecture that increases the header size. There are also more specific MTU considerations: Effects on Path MTU Discovery (PMTUD) as well as deployment considerations. Deployments within a single administrative control or even a single Data Center complex can afford more flexibility in dealing with larger packets, and deploying existing mitigations that decrease the likelihood of fragmentation or discard.

5.7. SFC OAM

Operations, Administration, and Maintenance (OAM) tools are an integral part of the architecture. These serve various purposes, including fault detection and isolation, and performance management. For example, there are many advantages of SFP liveness detection,

including status reporting, support for resiliency operations and policies, and an enhanced ability to balance load.

Service Function Paths create a services topology, and OAM performs various functions within this service layer. Furthermore, SFC OAM follows the same architectural principles of SFC in general. For example, topological independence (including the ability to run OAM over various overlay technologies) and classification-based policy.

We can subdivide the SFC OAM architecture in two parts:

- o In-band: OAM packets follow the same path and share fate with user packets, within the service topology. For this, they also follow the architectural principle of consistent policy identifiers, and use the same path IDs as the service chain data packets. Load balancing and SFC encapsulation with packet forwarding are particularly important here.
- o Out-of-band: reporting beyond the actual data plane. An additional layer beyond the data-plane OAM allows for additional alerting and measurements.

This architecture prescribes end-to-end SFP OAM functions, which implies SFF understanding of whether an in-band packet is an OAM or user packet. However, service function validation is outside of the scope of this architecture, and application-level OAM is not what this architecture prescribes.

Some of the detailed functions performed by SFC OAM include fault detection and isolation in a Service Function Path or a Service Function, verification that connectivity using SFPs is both effective and directing packets to the intended service functions, service path tracing, diagnostic and fault isolation, alarm reporting, performance measurement, locking and testing of service functions, validation with the control plane (see Section 5.2), and also allow for vendor-specific as well as experimental functions. SFC should leverage, and if needed extend relevant existing OAM mechanisms.

5.8. Resilience and Redundancy

As a practical operational requirement, any service chaining solution needs to be able to respond effectively, and usually very quickly, to failure conditions. These may be failures of connectivity in the network between SFFs, failures of SFFs, or failures of SFs. Per-SF state, as for example stateful-firewall state, is the responsibility of the SF, and not addressed by this architecture.

Multiple techniques are available to address this issue. Solutions can describe both what they require and what they allow to address failure. Solutions can make use of flexible specificity of service function paths, if the SFF can be given enough information in a timely fashion to do this. Solutions can also make use of MAC or IP level redundancy mechanisms such as VRRP. Also, particularly for SF failures, load balancers co-located with the SFF or as part of the service function delivery mechanism can provide such robustness.

Similarly, operational requirements imply resilience in the face of load changes. While mechanisms for managing (e.g., monitoring, instantiating, loading images, providing configuration to service function chaining control, deleting, etc.) virtual machines are out of scope for this architecture, solutions can and are aided by describing how they can make use of scaling mechanisms.

6. Security Considerations

This document does not define a new protocol and therefore creates no new security issues.

Security considerations apply to the realization of this architecture. Such realization ought to provide means to protect the SFC-enabled domain and its borders against various forms of attacks, including DDoS attacks. Further, SFC OAM Functions need to not negatively affect the security considerations of an SFC-enabled domain. Additionally, all entities (software or hardware) interacting with the service chaining mechanisms need to provide means of security against malformed, poorly configured (deliberate or not) protocol constructs and loops. These considerations are largely the same as those in any network, particularly an overlay network.

7. Contributors and Acknowledgments

The editors would like to thank Sam Aldrin, Linda Dunbar, Alla Goldner, Ken Gray, Shunsuke Homma, Dave Hood, Nagendra Kumar, Andrew Malis, Kengo Naito, Ron Parker, Xiaohu Xu, and Lucy Yong for a thorough review and useful comments.

The initial version of this "Service Function Chaining (SFC) Architecture" document is the result of merging two previous documents, and this section lists the aggregate of authors, editors, contributors and acknowledged participants, all who provided important ideas and text that fed into this architecture.

[I-D.boucadair-sfc-framework]:

Authors:

Mohamed Boucadair
Christian Jacquenet
Ron Parker
Diego R. Lopez
Jim Guichard
Carlos Pignataro

Contributors:

Parviz Yegani
Paul Quinn
Linda Dunbar

Acknowledgements:

Many thanks to D. Abgrall, D. Minodier, Y. Le Goff, D. Cheng, R. White, and B. Chatras for their review and comments.

[I-D.quinn-sfc-arch]:

Authors:

Paul Quinn (editor)
Joel Halpern (editor)

Contributors:

Puneet Agarwal
Andre Beliveau
Kevin Glavin
Ken Gray
Jim Guichard
Surendra Kumar
Darrel Lewis
Nic Leymann
Rajeev Manur
Thomas Nadeau
Carlos Pignataro
Michael Smith
Navindra Yadav

Acknowledgements:

The authors would like to thank David Ward, Abhijit Patra, Nagaraj Bagepalli, Darrel Lewis, Ron Parker, Lucy Yong and Christian Jacquenet for their review and comments.

8. IANA Considerations

This document creates no new requirements on IANA namespaces [RFC5226].

9. References

9.1. Normative References

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

9.2. Informative References

[I-D.boucadair-sfc-framework]
Boucadair, M., Jacquenet, C., Parker, R., Lopez, D., Guichard, J., and C. Pignataro, "Service Function Chaining: Framework & Architecture", draft-boucadair-sfc-framework-02 (work in progress), February 2014.

[I-D.ietf-sfc-problem-statement]
Quinn, P. and T. Nadeau, "Service Function Chaining Problem Statement", draft-ietf-sfc-problem-statement-10 (work in progress), August 2014.

[I-D.quinn-sfc-arch]
Quinn, P. and J. Halpern, "Service Function Chaining (SFC) Architecture", draft-quinn-sfc-arch-05 (work in progress), May 2014.

[RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.

[RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, June 2011.

Authors' Addresses

Joel Halpern (editor)
Ericsson

Email: jmh@joelhalpern.com

Carlos Pignataro (editor)
Cisco Systems, Inc.

Email: cpignata@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 28, 2015

P. Quinn
J. Guichard
S. Kumar
M. Smith
Cisco Systems, Inc.
W. Henderickx
Alcatel-Lucent
T. Nadeau
Brocade
P. Agarwal

R. Manur
Broadcom
A. Chauhan
Citrix
J. Halpern
Ericsson
S. Majee
F5
U. Elzur
Intel
D. Melman
Marvell
P. Garg
Microsoft
B. McConnell
Rackspace
C. Wright
Red Hat Inc.
K. Glavin
Riverbed
C. Zhang
L. Fourie
Huawei US R&D
R. Parker
Affirmed Networks
M. Zarny
Goldman Sachs
February 24, 2015

Network Service Header
draft-quinn-sfc-nsh-07.txt

Abstract

This draft describes a Network Service Header (NSH) inserted onto encapsulated packets or frames to realize service function paths.

NSH also provides a mechanism for metadata exchange along the instantiated service path.

1. Requirements Language

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

Status of this Memo

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

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

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

This Internet-Draft will expire on August 28, 2015.

Copyright Notice

Copyright (c) 2015 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 Language	3
2. Introduction	5
2.1. Definition of Terms	5
2.2. Problem Space	7
3. Network Service Header	9
3.1. Network Service Header Format	9
3.2. NSH Base Header	9
3.3. Service Path Header	11
3.4. NSH MD-type 1	11
3.4.1. Mandatory Context Header Allocation Guidelines	12
3.5. NSH MD-type 2	13
3.5.1. Optional Variable Length Metadata	14
4. NSH Actions	16
5. NSH Encapsulation	18
6. NSH Usage	19
7. NSH Proxy Nodes	20
8. Fragmentation Considerations	21
9. Service Path Forwarding with NSH	22
9.1. SFFs and Overlay Selection	22
9.2. Mapping NSH to Network Overlay	24
9.3. Service Plane Visibility	25
9.4. Service Graphs	25
10. Policy Enforcement with NSH	27
10.1. NSH Metadata and Policy Enforcement	27
10.2. Updating/Augmenting Metadata	28
10.3. Service Path ID and Metadata	30
11. NSH Encapsulation Examples	31
11.1. GRE + NSH	31
11.2. VXLAN-gpe + NSH	31
11.3. Ethernet + NSH	32
12. Security Considerations	33
13. Open Items for WG Discussion	34
14. Contributors	35
15. Acknowledgments	36
16. IANA Considerations	37
16.1. NSH EtherType	37
16.2. Network Service Header (NSH) Parameters	37
16.2.1. NSH Base Header Reserved Bits	37
16.2.2. MD Type Registry	37
16.2.3. TLV Class Registry	38
16.2.4. NSH Base Header Next Protocol	38
17. References	39
17.1. Normative References	39
17.2. Informative References	39
Authors' Addresses	41

2. Introduction

Service functions are widely deployed and essential in many networks. These service functions provide a range of features such as security, WAN acceleration, and server load balancing. Service functions may be instantiated at different points in the network infrastructure such as the wide area network, data center, campus, and so forth.

The current service function deployment models are relatively static, and bound to topology for insertion and policy selection. Furthermore, they do not adapt well to elastic service environments enabled by virtualization.

New data center network and cloud architectures require more flexible service function deployment models. Additionally, the transition to virtual platforms requires an agile service insertion model that supports elastic service delivery; the movement of service functions and application workloads in the network and the ability to easily bind service policy to granular information such as per-subscriber state are necessary.

The approach taken by NSH is composed of the following elements:

1. Service path identification
2. Transport independent per-packet/frame service metadata.
3. Optional variable TLV metadata.

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

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

The SFC Architecture document [SFC-arch] provides an overview of a service chaining architecture that clearly defines the roles of the various elements and the scope of a service function chaining encapsulation.

2.1. Definition of Terms

Classification: Locally instantiated policy and customer/network/service profile matching of traffic flows for identification of appropriate outbound forwarding actions.

SFC Network Forwarder (NF): SFC network forwarders provide network connectivity for service functions forwarders and service functions. SFC network forwarders participate in the network overlay used for service function chaining as well as in the SFC encapsulation.

Service Function Forwarder (SFF): A service function forwarder is responsible for delivering traffic received from the NF to one or more connected service functions, and from service functions to the NF.

Service Function (SF): A function that is responsible for specific treatment of received packets. A service function can act at the network layer or other OSI layers. A service function can be a virtual instance or be embedded in a physical network element. One of multiple service functions can be embedded in the same network element. Multiple instances of the service function can be enabled in the same administrative domain.

Service Node (SN): Physical or virtual element that hosts one or more service functions and has one or more network locators associated with it for reachability and service delivery.

Service Function Chain (SFC): A service function chain defines an ordered set of service functions that must be applied to packets and/or frames selected as a result of classification. The implied order may not be a linear progression as the architecture allows for nodes that copy to more than one branch. The term service chain is often used as shorthand for service function chain.

Service Function Path (SFP): The instantiation of a SFC in the network. Packets follow a service function path from a classifier through the requisite service functions

Network Node/Element: Device that forwards packets or frames based on outer header information. In most cases is not aware of the presence of NSH.

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

Network Service Header: Data plane header added to frames/packets. The header contains information required for service chaining, as well as metadata added and consumed by network nodes and service elements.

Service Classifier: Function that performs classification and imposes an NSH. Creates a service path. Non-initial (i.e. subsequent) classification can occur as needed and can alter, or create a new service path.

Service Hop: NSH aware node, akin to an IP hop but in the service overlay.

Service Path Segment: A segment of a service path overlay.

NSH Proxy: Acts as a gateway: removes and inserts NSH on behalf of a service function that is not NSH aware.

2.2. Problem Space

Network Service Header (NSH) addresses several limitations associated with service function deployments today.

1. **Topological Dependencies:** Network service deployments are often coupled to network topology. Such dependency imposes constraints on the service delivery, potentially inhibiting the network operator from optimally utilizing service resources, and reduces the flexibility. This limits scale, capacity, and redundancy across network resources.
2. **Service Chain Construction:** Service function chains today are most typically built through manual configuration processes. These are slow and error prone. With the advent of newer service deployment models the control/management planes provide not only connectivity state, but will also be increasingly utilized for the creation of network services. Such a control/management planes could be centralized, or be distributed.
3. **Application of Service Policy:** Service functions rely on topology information such as VLANs or packet (re) classification to determine service policy selection, i.e. the service function specific action taken. Topology information is increasingly less viable due to scaling, tenancy and complexity reasons. The topological information is often stale, providing the operator with inaccurate placement that can result in suboptimal resource utilization. Furthermore topology-centric information often does not convey adequate information to the service functions, forcing functions to individually perform more granular classification.
4. **Per-Service (re)Classification:** Classification occurs at each service function independent from previously applied service functions. More importantly, the classification functionality often differs per service function and service functions may not

leverage the results from other service functions.

5. Common Header Format: Various proprietary methods are used to share metadata and create service paths. An open header provides a common format for all network and service devices.
6. Limited End-to-End Service Visibility: Troubleshooting service related issues is a complex process that involve both network-specific and service-specific expertise. This is especially the case when service function chains span multiple DCs, or across administrative boundaries. Furthermore, the physical and virtual environments (network and service) can be highly divergent in terms of topology and that topological variance adds to these challenges.
7. Transport Dependence: Service functions can and will be deployed in networks with a range of transports requiring service functions to support and participate in many transports (and associated control planes) or for a transport gateway function to be present.

Please see the Service Function Chaining Problem Statement [SFC-PS] for a more detailed analysis of service function deployment problem areas.

3. Network Service Header

A Network Service Header (NSH) contains metadata and service path information that are added to a packet or frame and used to create a service plane. The packets and the NSH are then encapsulated in an outer header for transport.

The service header is added by a service classification function - a device or application - that determines which packets require servicing, and correspondingly which service path to follow to apply the appropriate service.

3.1. Network Service Header Format

An NSH is composed of a 4-byte base header, a 4-byte service path header and context headers, as shown in Figure 1 below.

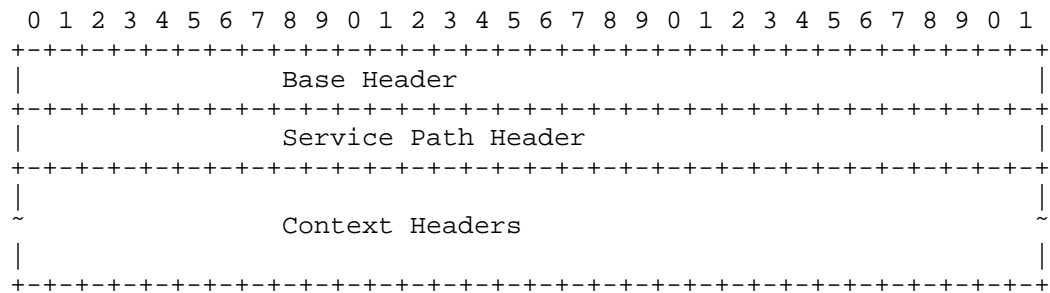


Figure 1: Network Service Header

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

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

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

3.2. NSH Base Header

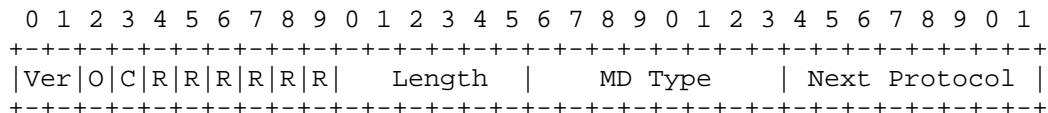


Figure 2: NSH Base Header

Base Header Field Descriptions

Version: The version field is used to ensure backward compatibility going forward with future NSH updates.

O bit: Indicates that this packet is an operations and management (OAM) packet. SFF and SFs nodes MUST examine the payload and take appropriate action (e.g. return status information).

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

C bit: Indicates that a critical metadata TLV is present (see Section 3.4.2). This bit acts as an indication for hardware implementers to decide how to handle the presence of a critical TLV without necessarily needing to parse all TLVs present. The C bit MUST be set to 1 if one or more critical TLVs are present.

All other flag fields are reserved.

Length: total length, in 4-byte words, of the NSH header, including optional variable TLVs.

MD Type: indicates the format of NSH beyond the base header and the type of metadata being carried. This typing is used to describe the use for the metadata. A new registry will be requested from IANA for the MD Type.

NSH defines two MD types:

0x1 which indicates that the format of the header includes fixed length context headers.

0x2 which does not mandate any headers beyond the base header and service path header, and may contain optional variable length context information.

The format of the base header is invariant, and not described by MD Type.

NSH implementations MUST support MD-Type 0x1, and SHOULD support MD-Type 0x2.

Next Protocol: indicates the protocol type of the original packet. A new IANA registry will be created for protocol type.

This draft defines the following Next Protocol values:

```
0x1 : IPv4
0x2 : IPv6
0x3 : Ethernet
```

3.3. Service Path Header

```

 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               Service Path ID                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               Service Index                               |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Service path ID (SPI): 24 bits
 Service index (SI): 8 bits

Figure 3: NSH Service Path Header

Service Path Identifier (SPI): identifies a service path. Participating nodes **MUST** use this identifier for path selection. An administrator can use the service path value for reporting and troubleshooting packets along a specific path.

Service Index (SI): provides location within the service path. Service index **MUST** be decremented by service functions or proxy nodes after performing required services. **MAY** be used in conjunction with service path for path selection. Service Index is also valuable when troubleshooting/reporting service paths. In addition to location within a path, SI can be used for loop detection.

3.4. NSH MD-type 1

When the base header specifies MD Type 1, NSH defines four 4-byte mandatory context headers, as per Figure 4. These headers must be present and the format is opaque as depicted in Figure 5.

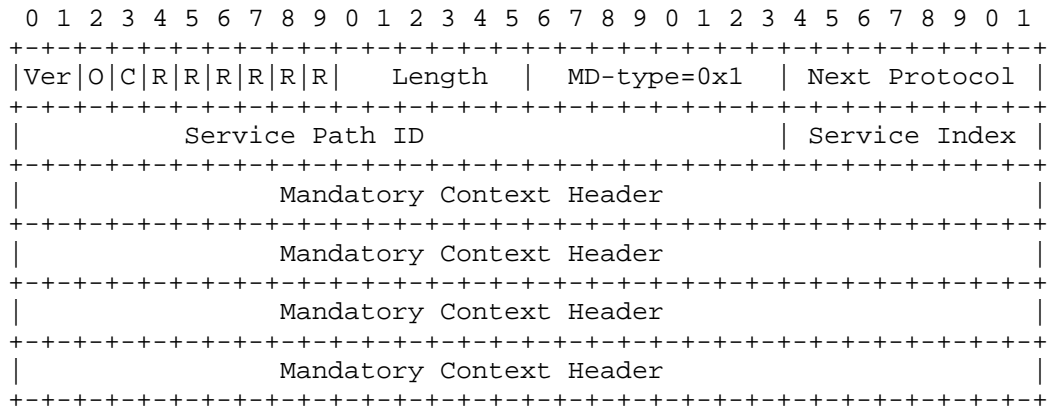


Figure 4: NSH MD-type=0x1

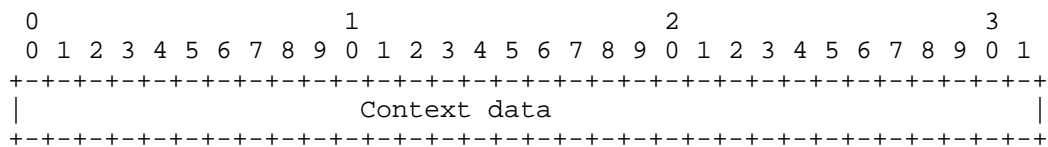


Figure 5: Context Header

3.4.1. Mandatory Context Header Allocation Guidelines

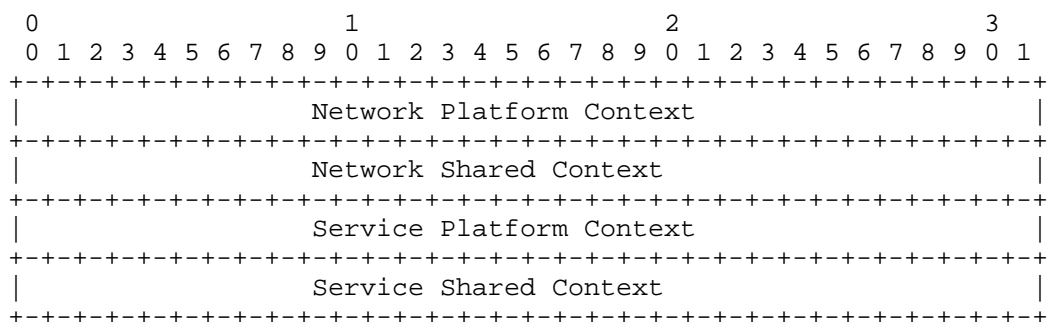


Figure 6: Context Data Significance

Figure 6, above, and the following examples of context header allocation are guidelines that illustrate how various forms of information can be carried and exchanged via NSH.

Network platform context: provides platform-specific metadata shared between network nodes. Examples include (but are not limited to) ingress port information, forwarding context and encapsulation type.

Network shared context: metadata relevant to any network node such as the result of edge classification. For example, application information, identity information or tenancy information can be shared using this context header.

Service platform context: provides service platform specific metadata shared between service functions. This context header is analogous to the network platform context, enabling service platforms to exchange platform-centric information such as an identifier used for load balancing decisions.

Service shared context: metadata relevant to, and shared, between service functions. As with the shared network context, classification information such as application type can be conveyed using this context.

The data center[dcalloc] and mobility[moballoc] context header allocation drafts provide guidelines for the semantics of NSH fixed context headers in each respective environment.

3.5. NSH MD-type 2

When the base header specifies MD Type 2, NSH defines variable length only context headers. There may be zero or more of these headers as per the length field.

```

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|Ver|O|C|R|R|R|R|R|R|   Length   | MD-type=0x2 | Next Protocol |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               | Service Path ID | Service Index |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               |                               |
~                               ~
|                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Figure 7: NSH MD-type=0x2

3.5.1. Optional Variable Length Metadata

NSH MD Type 2 MAY contain optional variable length context headers. The format of these headers is as described below.

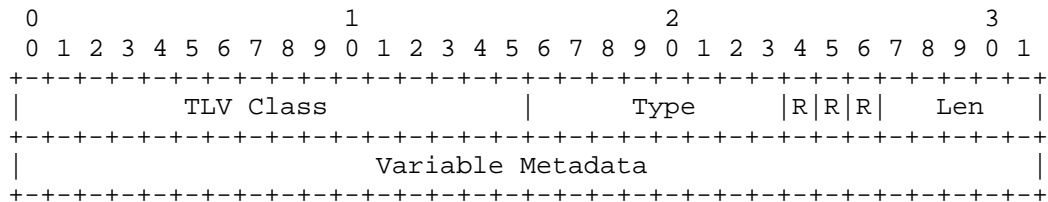


Figure 8: Variable Context Headers

TLV Class: describes the scope of the "Type" field. In some cases, the TLV Class will identify a specific vendor, in others, the TLV Class will identify specific standards body allocated types.

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

The most significant bit of the Type field indicates whether the TLV is mandatory for the receiver to understand/process. This effectively allocates Type values 0 to 127 for non-critical options and Type values 128 to 255 for critical options. Figure 9 below illustrates the placement of the Critical bit within the Type field.

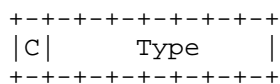


Figure 9: Critical Bit Placement Within the TLV Type Field

Encoding the criticality of the TLV within the Type field is consistent with IPv6 option types.

If a receiver receives an encapsulated packet containing a TLV with the Critical bit set in the Type field and it does not understand how to process the Type, it MUST drop the packet. Transit devices MUST NOT drop packets based on the setting of this bit.

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

reserved bits MUST be zero.

Length: Length of the variable metadata, in 4-byte words.

4. NSH Actions

Service header aware nodes - service classifiers, SFF, SF and NSH proxies, have several possible header related actions:

1. Insert or remove service header: These actions can occur at the start and end respectively of a service path. Packets are classified, and if determined to require servicing, a service header imposed. The last node in a service path, an SFF, removes the NSH. A service classifier MUST insert an NSH. At the end of a service function chain, the last node operating on the service header MUST remove it.

A service function can re-classify data as required and that re-classification might result in a new service path. In this case, the SF acts as a logical classifier as well. When the logical classifier performs re-classification that results in a change of service path, it MUST remove the existing NSH and MUST impose a new NSH with the base header reflecting the new path.

2. Select service path: The base header provides service chain information and is used by SFFs to determine correct service path selection. SFFs MUST use the base header for selecting the next service in the service path.
3. Update a service header: NSH aware service functions MUST decrement the service index. A service index = 0 indicates that a packet MUST be dropped by the SFF performing NSH-based forwarding.

Service functions MAY update context headers if new/updated context is available.

If an NSH proxy (see Section 7) is in use (acting on behalf of a non-NSH-aware service function for NSH actions), then the proxy MUST update service index and MAY update contexts. When an NSH proxy receives an NSH-encapsulated packet, it removes the NSH before forwarding it to an NSH unaware SF. When it receives a packet back from an NSH unaware SF, it re-encapsulates it with the NSH, decrementing the service index.

4. Service policy selection: Service function instances derive policy selection from the service header. Context shared in the service header can provide a range of service-relevant information such as traffic classification. Service functions SHOULD use NSH to select local service policy.

Figure 10 maps each of the four actions above to the components in the SFC architecture that can perform it.

Component	Insert or remove service header			Select service path	Update a service header		Service Policy
	Insert	Remove	Remove and Insert		Dec. Service Index	Update Context Header	
Service Classification Function	+					+	
Service Function Forwarder(SFF)		+		+		+	
Service Function (SF)					+	+	+
NSH Proxy	+	+			+	+	

Figure 10: NSH Action and Role Mapping

5. NSH Encapsulation

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

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

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

See Section 11 for NSH encapsulation examples.

6. NSH Usage

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

1. **Topological Independence:** Service forwarding occurs within the service plane, via a network overlay, the underlying network topology does not require modification. Service functions have one or more network locators (e.g. IP address) to receive/send data within the service plane, the NSH contains an identifier that is used to uniquely identify a service path and the services within that path.
2. **Service Chaining:** NSH contains path identification information needed to realize a service path. Furthermore, NSH provides the ability to monitor and troubleshoot a service chain, end-to-end via service-specific OAM messages. The NSH fields can be used by administrators (via, for example a traffic analyzer) to verify (account, ensure correct chaining, provide reports, etc.) the path specifics of packets being forwarded along a service path.
3. **Metadata Sharing:** NSH provides a mechanism to carry shared metadata between network devices and service function, and between service functions. The semantics of the shared metadata is communicated via a control plane to participating nodes. Examples of metadata include classification information used for policy enforcement and network context for forwarding post service delivery.
4. **Transport Agnostic:** NSH is transport independent and is carried in an overlay, over existing underlays. If an existing overlay topology provides the required service path connectivity, that existing overlay may be used.

7. NSH Proxy Nodes

In order to support NSH-unaware service functions, an NSH proxy is used. The proxy node removes the NSH header and delivers the original packet/frame via a local attachment circuit to the service function. Examples of a local attachment circuit include, but are not limited to: VLANs, IP in IP, GRE, VXLAN. When complete, the service function returns the packet to the NSH proxy via the same or different attachment circuit.

NSH is re-imposed on packets returned to the proxy from the non-NSH-aware service.

Typically, an SFF will act as an NSH-proxy when required.

An NSH proxy MUST perform NSH actions as described in Section 4.

8. Fragmentation Considerations

Work in progress

9. Service Path Forwarding with NSH

9.1. SFFs and Overlay Selection

As described above, NSH contains a service path identifier (SPI) and a service index (SI). The SPI is, as per its name, an identifier. The SPI alone cannot be used to forward packets along a service path. Rather the SPI provide a level of indirection between the service path/topology and the network transport. Furthermore, there is no requirement, or expectation of an SPI being bound to a pre-determined or static network path.

The service index provides an indication of location within a service path. The combination of SPI and SI provides the identification and location of a logical SF (locator and order). The logical SF may be a single SF, or a set of SFs that are equivalent. In the latter case, the SFF provides load distribution amongst the collection of SFs as needed. SI may also serve as a mechanism for loop detection with in a service path since each SF in the path decrements the index; an index of 0 indicates that a loop occurred and packet must be discarded.

This indirection -- path ID to overlay -- creates a true service plane. That is the SFF/SF topology is constructed without impacting the network topology but more importantly service plane only participants (i.e. most SFs) need not be part of the network overlay topology and its associated infrastructure (e.g. control plane, routing tables, etc.). As mentioned above, an existing overlay topology may be used provided it offers the requisite connectivity.

The mapping of SPI to transport occurs on an SFF. The SFF consults the SPI/ID values to determine the appropriate overlay transport protocol (several may be used within a given network) and next hop for the requisite SF. Figure 10 below depicts an SPI/SI to network overlay mapping.

SPI	SI	NH	Transport
10	3	1.1.1.1	VXLAN-gpe
10	2	2.2.2.2	nvGRE
245	12	192.168.45.3	VXLAN-gpe
10	9	10.1.2.3	GRE
40	9	10.1.2.3	GRE
50	7	01:23:45:67:89:ab	Ethernet
15	1	Null (end of path)	None

Figure 11: SFF NSH Mapping Example

Additionally, further indirection is possible: the resolution of the required SF function locator may be a localized resolution on an SFF, rather than a service function chain control plane responsibility, as per figures 11 and 12 below.

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

Figure 12: NSH to SF Mapping Example

SF	NH	Transport
SF2	10.1.1.1	VXLAN-gpe
SF34	192.168.1.1	UDP
SF9	1.1.1.1	GRE

Figure 13: SF Locator Mapping Example

Since the SPI is a representation of the service path, the lookup may return more than one possible next-hop within a service path for a given SF, essentially a series of weighted (equally or otherwise) overlay links to be used (for load distribution, redundancy or policy), see Figure 13. The metric depicted in Figure 13 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 sate, capacity, etc.)

SPI	SI	NH	Metric
10	3	10.1.1.1	1
		10.1.1.2	1
20	12	192.168.1.1	1
		10.2.2.2	1
30	7	10.2.2.3	10
		10.3.3.3	5

(encap type omitted for formatting)

Figure 14: NSH Weighted Service Path

9.2. Mapping NSH to Network Overlay

As described above, the mapping of SPI to network topology may result in a single overlay path, or it might result in a more complex topology. Furthermore, the SPIx to overlay mapping occurs at each SFF independently. Any combination of topology selection is possible.

Examples of mapping for a topology:

1. Next SF is located at SFFb with locator 10.1.1.1
SFFa mapping: SPI=10 --> VXLAN-gpe, dst-ip: 10.1.1.1
2. Next SF is located at SFFc with multiple locator for load distribution purposes:
SFFb mapping: SPI=10 --> VXLAN-gpe, dst_ip:10.2.2.1, 10.2.2.2, 10.2.2.3, equal cost
3. Next SF is located at SFFd with two path to SFFc, one for redundancy:
SFFc mapping: SPI=10 --> VXLAN-gpe, dst_ip:10.1.1.1 cost=10, 10.1.1.2, cost=20

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

The network operator retains the ability to engineer the overlay paths as required. For example, the overlay path between service functions forwarders may utilize traffic engineering, QoS marking, or

ECMP, without requiring complex configuration and network protocol support to be extended to the service path explicitly. In other words, the network operates as expected, and evolves as required, as does the service function plane.

9.3. Service Plane Visibility

The SPI and SI serve an important function for visibility into the service topology. An operator can determine what service path a packet is "on", and its location within that path simply by viewing the NSH information (packet capture, IPFIX, etc.). The information can be used for service scheduling and placement decisions, troubleshooting and compliance verification.

9.4. Service Graphs

In some cases, a service path is exactly that -- a linear list of service functions that must be traversed. However, increasingly, the "path" is actually a true directed graph. Furthermore, within a given service topology several directed graphs may exist with packets moving between graphs based on non-initial classification (usually performed by a service function). Note: strictly speaking a path is a form of graph; the intent is to distinguish between a directed graph and a path.

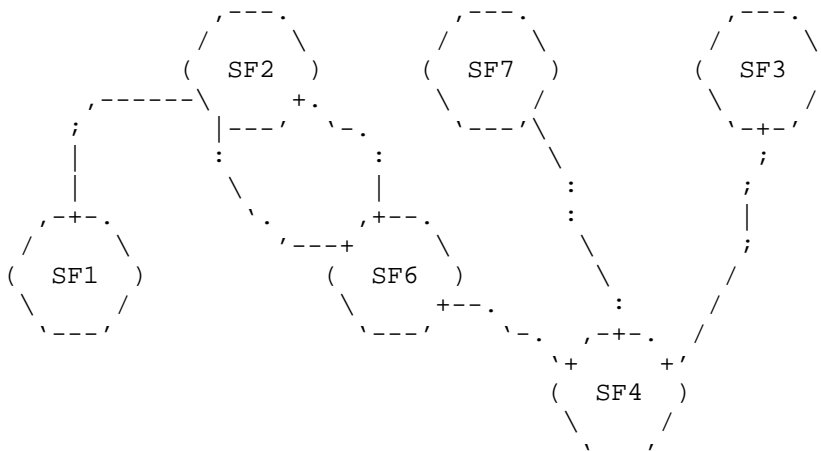


Figure 15: Service Graph Example

The SPI/SI combination provides a simple representation of a directed graph, the SPI represents a graph ID; and the SI a node ID. The

service topology formed by SPI/SI support cycles, weighting, and alternate topology selection, all within the service plane. The realization of the network topology occurs as described above: SPI/ID mapping to an appropriate transport and associated next network hops.

NSH-aware services receive the entire header, including the SPI/SI. An SF can now, based on local policy, alter the SPI, which in turn effects both the service graph, and in turn the selection of overlay at the SFF. The figure below depicts the policy associated with the graph in Figure 14 above. Note: this illustrates multiple graphs and their representation; it does not depict the use of metadata within a single service function graph.

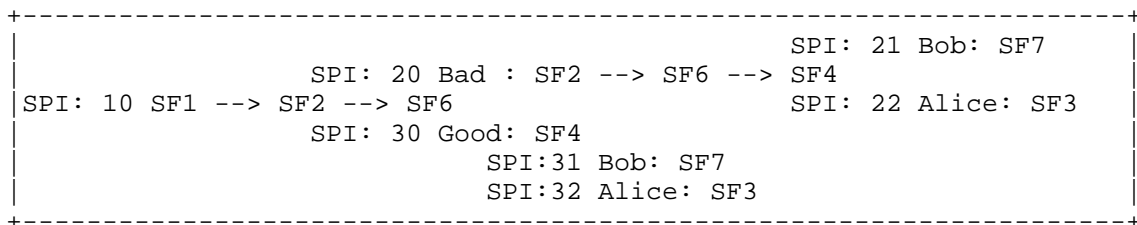


Figure 16: Service Graphs Using SPI

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

10. Policy Enforcement with NSH

10.1. NSH Metadata and Policy Enforcement

As described in Section 3, NSH provides the ability to carry metadata along a service path. This metadata may be derived from several sources, common examples include:

Network nodes: 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 pathing.

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.

Service functions: Service functions often perform very detailed and valuable classification. In some cases they may terminate, and be able to inspect encrypted traffic. SFs may update, alter or impose metadata information.

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

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

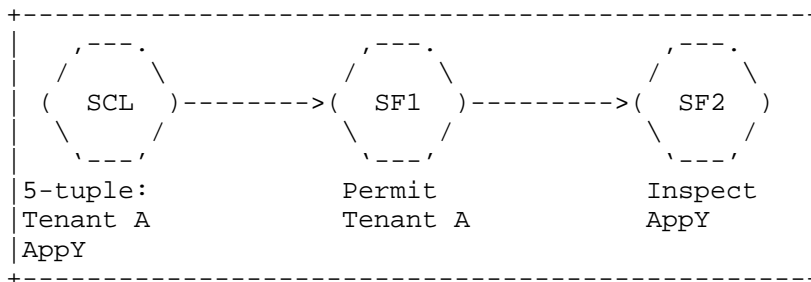


Figure 17: Metadata and Policy

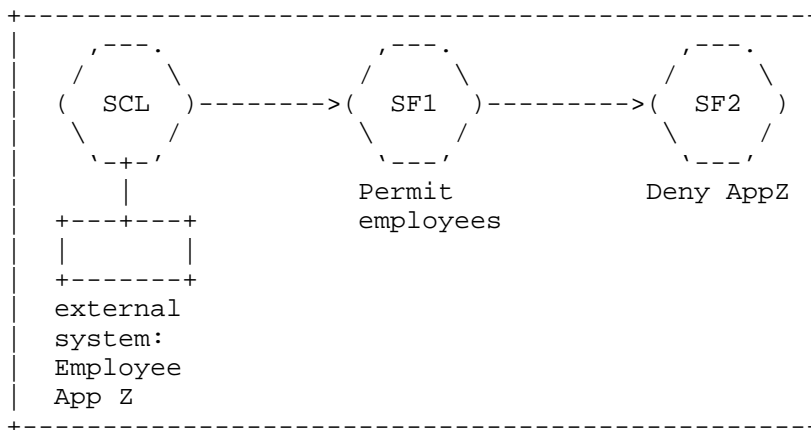


Figure 18: External Metadata and Policy

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

10.2. Updating/Augmenting Metadata

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

1. **Metadata Augmentation:** Information may be added to NSH's existing metadata, as depicted in Figure 18. For example, if the initial classification returns the tenant information, a secondary classification (perhaps a DPI or SLB) may augment the tenant classification with application information. 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 19 illustrates an example of updating metadata.

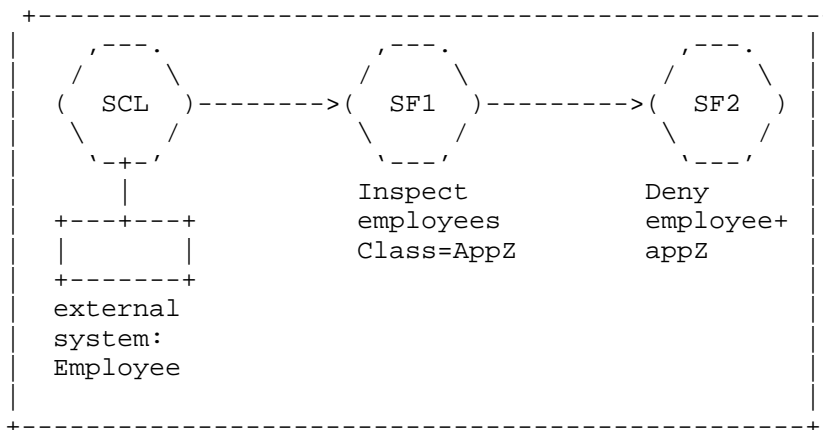


Figure 19: Metadata Augmentation

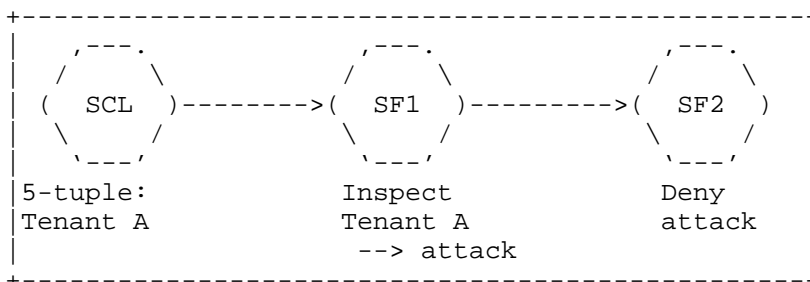


Figure 20: Metadata Update

10.3. Service Path ID and Metadata

Metadata information may influence the service path selection since the service path identifier can represent the result of classification. A given SPI can represent all or some of the metadata, and be updated based on metadata classification results. This relationship provides the ability to create a dynamic services plane based on complex classification without requiring each node to be capable of such classification, or requiring a coupling to the network topology. This yields service graph functionality as described in Section 9.4. Figure 20 illustrates an example of this behavior.

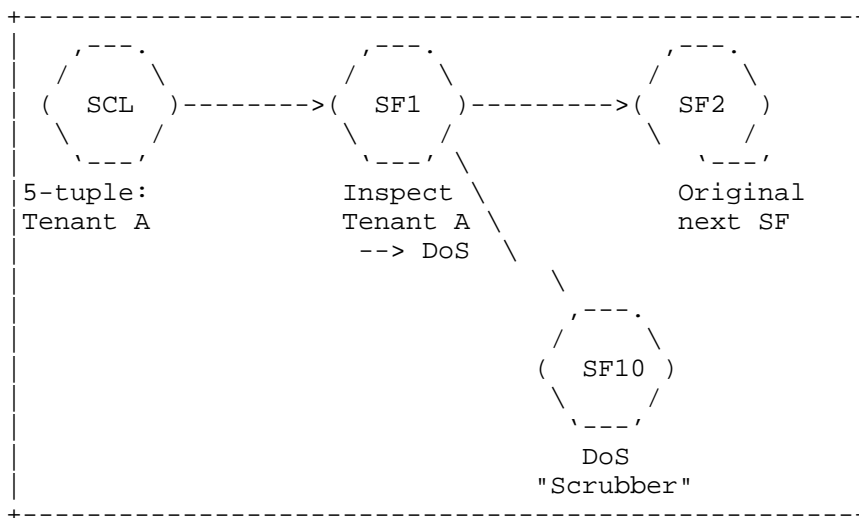


Figure 21: Path ID and Metadata

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

11. NSH Encapsulation Examples

11.1. GRE + NSH

```

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

```

```

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

```

Figure 22: GRE + NSH

11.2. VXLAN-gpe + NSH

```

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

```

```

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

```

Figure 23: VXLAN-gpe + NSH

11.3. Ethernet + NSH

IPv4 Packet:

Outer Ethernet, ET=0x894F	NSH, NP = 0x1	original IP Packet
---------------------------	---------------	--------------------

L2 Frame:

Outer Ethernet, ET=0x894F	NSH, NP = 0x3	original frame
---------------------------	---------------	----------------

Figure 24: Ethernet + NSH

12. Security Considerations

As with many other protocols, NSH data can be spoofed or otherwise modified. In many deployments, NSH will be used in a controlled environment, with trusted devices (e.g. a data center) thus mitigating the risk of unauthorized header manipulation.

NSH is always encapsulated in a transport protocol and therefore, when required, existing security protocols that provide authenticity (e.g. RFC 2119 [RFC6071]) can be used.

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

13. Open Items for WG Discussion

1. MD type 1 metadata semantics specifics
2. Bypass bit in NSH.
3. Rendered Service Path ID (RSPID).

14. Contributors

The following people are active contributors to this document and have provided review, content and concepts (listed alphabetically by surname):

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

Rex Fernando
Cisco Systems
Email: rex@cisco.com

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

Navindra Yadav
Cisco Systems
Email: nyadav@cisco.com

15. Acknowledgments

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

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

Additionally the authors would like to thank Carlos Pignataro and Larry Kreeger for their invaluable ideas and contributions which are reflected throughout this draft.

16. IANA Considerations

16.1. NSH EtherType

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

16.2. Network Service Header (NSH) Parameters

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

16.2.1. NSH Base Header Reserved Bits

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

Bits 0-1 - Version
Bit 2 - OAM (O bit)
Bits 2-9 - Reserved

16.2.2. MD Type Registry

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

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

Table 1

16.2.3. TLV Class Registry

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

16.2.4. NSH Base Header Next Protocol

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

Next Protocol	Description	Reference
0	Reserved	This document
1	IPv4	This document
2	IPv6	This document
3	Ethernet	This document
4..253	Unassigned	

Table 2

17. References

17.1. Normative References

- [RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

17.2. Informative References

- [RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, March 2000.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
- [RFC6071] Frankel, S. and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap", RFC 6071, February 2011.
- [SFC-PS] Quinn, P., Ed. and T. Nadeau, Ed., "Service Function Chaining Problem Statement", 2014, <<http://datatracker.ietf.org/doc/draft-ietf-sfc-problem-statement/>>.
- [SFC-arch] Quinn, P., Ed. and J. Halpern, Ed., "Service Function Chaining (SFC) Architecture", 2014, <<http://datatracker.ietf.org/doc/draft-quinn-sfc-arch>>.
- [VXLAN-gpe] Quinn, P., Agarwal, P., Kreeger, L., Lewis, D., Maino, F., Yong, L., Xu, X., Elzur, U., and P. Garg, "Generic Protocol Extension for VXLAN", <<https://datatracker.ietf.org/doc/draft-quinn-vxlan-gpe/>>.
- [dcalloc] Guichard, J., Smith, M., and S. Kumar, "Network Service Header (NSH) Context Header Allocation (Data Center)", 2014, <<https://datatracker.ietf.org/doc/draft-guichard-sfc-nsh-dc-allocation/>>.
- [moballoc] Napper, J. and S. Kumar, "NSH Context Header Allocation -- Mobility", 2014, <<https://datatracker.ietf.org/doc/>>.

draft-napper-sfc-nsh-mobility-allocation/>.

Authors' Addresses

Paul Quinn
Cisco Systems, Inc.

Email: paulq@cisco.com

Jim Guichard
Cisco Systems, Inc.

Email: jguichar@cisco.com

Surendra Kumar
Cisco Systems, Inc.

Email: smkumar@cisco.com

Michael Smith
Cisco Systems, Inc.

Email: michsmit@cisco.com

Wim Henderickx
Alcatel-Lucent

Email: wim.henderickx@alcatel-lucent.com

Tom Nadeau
Brocade

Email: tnadeau@lucidvision.com

Puneet Agarwal

Email: puneet@acm.org

Rajeev Manur
Broadcom

Email: rmanur@broadcom.com

Abhishek Chauhan
Citrix

Email: Abhishek.Chauhan@citrix.com

Joel Halpern
Ericsson

Email: joel.halpern@ericsson.com

Sumandra Majee
F5

Email: S.Majee@F5.com

Uri Elzur
Intel

Email: uri.elzur@intel.com

David Melman
Marvell

Email: davidme@marvell.com

Pankaj Garg
Microsoft

Email: Garg.Pankaj@microsoft.com

Brad McConnell
Rackspace

Email: bmcconne@rackspace.com

Chris Wright
Red Hat Inc.

Email: chrisw@redhat.com

Kevin Glavin
Riverbed

Email: kevin.glavin@riverbed.com

Hong (Cathy) Zhang
Huawei US R&D

Email: cathy.h.zhang@huawei.com

Louis Fourie
Huawei US R&D

Email: louis.fourie@huawei.com

Ron Parker
Affirmed Networks

Email: ron_parker@affirmednetworks.com

Myo Zarny
Goldman Sachs

Email: myo.zarny@gs.com

Network Working Group
Internet-Draft
Intended Status: Proposed Standard

H. Zhang
L. Fourie
Huawei
R. Parker
Affirmed Networks
M. Zarny
Goldman Sachs

Expires: May 24, 2015

December 23, 2014

Service Chain Header
draft-zhang-sfc-sch-03

Abstract This document describes a service chaining header format and encapsulation mechanism that is used to facilitate the forwarding of data packets along the service function chain path. This header also allows for the transport of metadata to support various service chain related functionality.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

Copyright and License Notice

Copyright (c) 2014 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. Conventions used in this document	3
3. Terminology	3
4.2 Mandatory Fields	6
4.3 Optional Metadata Fields	7
4.3.1 Generic Context Block	8
4.3.2 Rendered Service Path Identifier	8
4.4 Header Associated Operations	9
5. SCH Encapsulation	11
5.1 SCH in Overlay GRE Encapsulation	11
5.2 SCH in Overlay VXLAN Encapsulation	11
5.3 SCH in IP/UDP Encapsulation	12
5.4 SCH in MAC Encapsulation	12
6. Security Considerations	13
7. IANA and IEEE Considerations	13
8. References	13
8.1 Normative References	13
8.2 Informative References	13
9. Appendix A	15
9.1 OAM Operation	15
9.2 Service Function Selector	15
9.3 Target Address	15
9.4 OAM Service Function Trace	16
10. Acknowledgments	16
Authors' Addresses	16

1. Introduction

Service chaining is a traffic steering technology for applying an ordered set of network service functions to traffic flows between two endpoints. Network service functions may include NAT, firewall, server load balancing, WAN optimization, and others, many of which can operate up to OSI Layer 7, and run on hardware appliances or virtual machines.

Service function chaining is an enabling technology for providing more agile service delivery required by cloud computing and network functions virtualization. (See [SFC-PS] and [SFC-ARCH] for more detailed discussions on service chaining.)

In order to support service function chaining, a mechanism is needed to carry service function chain (SFC) path and optional metadata information across various SFC entities. Ideally, the mechanism imposes minimal network overhead, works over various transport technologies at different OSI layers, and can accommodate future services. Given the likelihood of innovation in existing and future service functions and the impossibility of predicting what form every type of metadata will take, the mechanism should allow for the flexibility of carrying different types and lengths of metadata for different usage scenarios, and accommodate the addition of new types of service metadata. (See [SFC-META] for metadata considerations for service function chains.)

It is acknowledged that an either out-of-band or in-band mechanism or a combination of both may be utilized to transfer metadata in service function chains. This document describes only an in-band mechanism.

This document proposes the use of the Service Chain Header (SCH), which comprises a fixed-length mandatory portion used for steering purposes and an optional variable-length TLV (Type-Length-Value) approach to carry in-band metadata between service function chaining entities.

The SCH may be used to carry: (1) both SFC path steering information and metadata; (2) only SFC path steering information, in which case the Metadata Length field shall be set to zero; or (3) only metadata, in which case the Path Identifier and SF Index fields shall be set to zero for transmit and ignored upon receipt.

2. Conventions used in this document

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].

3. Terminology

Most of the terminologies used in this document are from [SFC-ARCH], [SFC-FWK] and [SFC-META], and are summarized here for convenience:

Service Function (SF): A network function that provides a value-added service to packet flows. A service function can act at any OSI layer. Service functions include: firewall, DPI (Deep Packet Inspection), NAT, HTTP Header Enrichment function, TCP optimizer, load-balancer, etc.

SF Chain: An ordered list of Service Function instances. See SF Map.

SF Chain Identifier: Identifies an SF Chain (SF Map Index).

SFC-enabled domain: Denotes a network (or a region thereof) that implements SFC.

SF Identifier: A unique identifier that unambiguously identifies an SF within an SFC-enabled domain. SF Identifiers are assigned, configured and managed by the administrative entity that operates the SFC-enabled domain. SF identifiers can be structured as strings, or in other formats. SF Identifiers are not required to be globally unique nor be exposed to or used by another SF-enabled domain.

Service Function Forwarder (SFF): Provides service layer forwarding. An SFF receives frames/packets from an SFC Network Forwarder and forwards the traffic to the associated SF(s) using information contained in the SCH.

SFC Network Forwarder (SNF): Forwards traffic flows along the SFPs they belong to based on the information contained in the SFC encapsulation.

SF Map: Refers to an ordered list of SF identifiers. Each SF Map is identified with a unique identifier called SF Map Index. This is referred to as a service function chain in this document.

SF Locator: An SF Node identifier used to reach the said SF node. A locator is typically an IP address or a FQDN.

Legacy Node: Refers to any node that is not an SF Node nor an SFC Boundary Node. This node can be located within an SFC-enabled domain or outside an SFC-enabled domain.

SF Proxy Node: A Network Element along the data path, to enforce SFC functions on behalf of legacy SF nodes.

SFC Boundary Node (or Boundary Node): Denotes a node that connects one SFC-enabled domain to a node either located in another SFC-

enabled domain or in a domain that is SFC-unaware.

SFC Egress Node (or Egress Node): Denotes an SFC Boundary Node that handles traffic which leaves the SFC-enabled domain the Egress Node belongs to.

SFC Ingress Node (or Ingress Node): Denotes an SFC Boundary Node that handles the traffic entering the SFC-enabled domain the Ingress Node belongs to.

SF Node: Denotes any node within an SFC-enabled domain that embeds one or multiple SFs.

Service Function Instance (SFI): Denotes an instantiation of a service function on a service node, such as a FW instance. A service function can have multiple service instances running on the same SF node with each service instance having its own service profile.

SFC Classifier (or Classifier): An entity that classifies frames or packets for service chaining according to classification rules defined in an SFC Policy Table. Frames/packets are then marked with the corresponding SF Chain Identifier. SFC Classifier can be embedded in an SFC Boundary (Ingress) Node or SF proxy node in which case the Classifier will do the encapsulation after getting the L2/L3 frame/packet from a Legacy SN. The SFC Classifier can also run on an independent (physical or virtual) platform.

Metadata: Provides contextual information about the data packets which traverse a service chain. Metadata can be used to convey contextual information not available at one location in the network to another location in the network where that information is not readily available. While primarily intended for consumption by SF(s), metadata MAY also be interpreted by other SFC entities.

4. Service Chain Header 4.1 Header Format

The Service Chain Header (SCH) consists of a mandatory fixed length part followed by a number of optional variable length metadata as shown in Figure 1. The mandatory fields carry "SFC path" information, which is used to steer the frames or packets through an ordered set of service function instances along the service function chain. The optional variable length fields carry application/service/content related metadata information which can be used by any SFC entities. The optional fields are formatted as Type-Length-Value structures. If any field in the header is not in use, the value of that field MUST be set to zero.

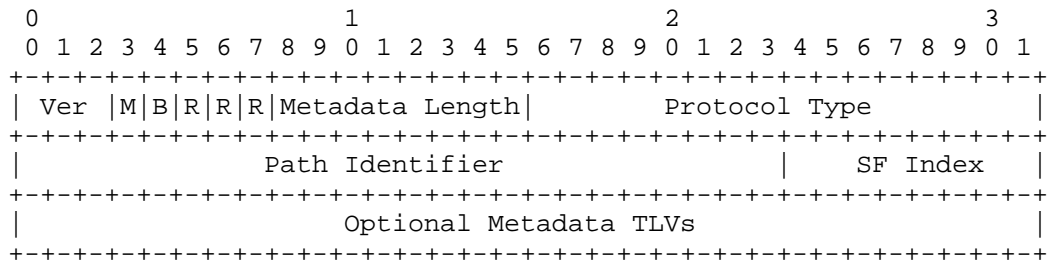


Figure 1: Service Chain Header

4.2 Mandatory Fields

Ver: Represents the Service Chain Header version. This field is 3 bits long, and the current version is 0.

M bit: Indicates that this frame/packet is an operations and management (OAM) packet. SF nodes MUST examine the payload and take appropriate action (i.e. return status information). The M-bit may be used in conjunction with OAM-specific TLVs (See Appendix A for an example). OAM message specifics and handling details are for future study and are outside the scope of this document.

B bit: The B (Bypass) bit, when set to 1, it is used by a Service Function to signal to its Service Function Forwarder that no further packets are to be sent to it for the flow specified in encapsulated packet.

R bits: Reserved bits for future extensions. These bits MUST be set to 0 on transmit, and ignored on receive.

Metadata length: The total length of all of the optional Metadata TLVs, in 4 octet units.

Protocol type: The 2-octet IEEE EtherType of the packet or frame immediately following the entire SCH header [ETYPES].

Path Identifier: Identifies a service function path. It represents a sequence of network locators, one for each service function that is to be invoked. Participating SFC entities MUST use this identifier when selecting the next hop for the packet or frame. The path identifier can also be used for diagnostics and troubleshooting associated with a service chain. For cases where the SCH is used solely to convey metadata, the Path Identifier is set to 0

on transmit and ignored on receive.

Service Function Index: An index to each service function instance associated with the service path. The service path index can be used to handle loop detection, flow reentry into a previous SF node of the SFC requiring another different service instance treatment, etc. The first service function instance in the path is identified with service index 1. For cases where the SCH is used solely to convey metadata, the Service Function Index is set to 0 on transmit and ignored on receive.

4.3 Optional Metadata Fields

Optional metadata are added after the fixed part of the SCH. Each option is of variable length and has a minimum length of four octets. An optional 3-octet Organizational Unique Identifier (OUI) may be provided to differentiate multiple private number spaces for the Type field. If the OUI is not provided, the Type is assumed to be a registered globally unique type. Any SFC entities MAY add, modify, or remove metadata TLVs.

The Appendix provides some examples of potential metadata TLV types and usage for reference.

Figure 2 shows the format of the TLV:

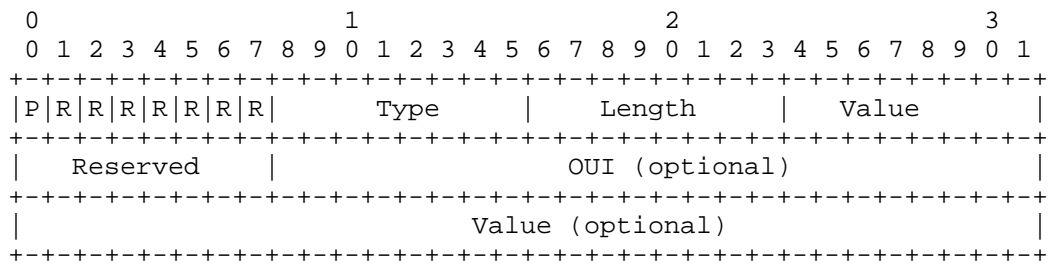


Figure 2: TLV format

P bit: The P (private) bit, when set to P=1, indicates that the optional OUI word (8-bit Reserved and 24-bit OUI) is present in this TLV. P=0 indicates that the optional OUI word is not present.

R bits: Reserved bits for future extensions. These bits MUST be set to 0 on transmit and ignored on receive.

Type: The type of the TLV, interpreted globally or per the OUI if present. A maximum of 256 types may be expressed within any

particular number space.

Length: The total length of the TLV in 4-octet units. A maximum length of 1024 octets may be expressed.

OUI: The 24-bit Organizational Unique Identifier [IEEE-OUI], present only if the P bit is set.

Value: Usage of the 8-bit value in the first word is purely optional and may be useful for certain types of metadata, allowing registered types to fit into one 32-bit word and OUI-based types to fit into two 32-bit words. When the 8-bit value is insufficient, additional words for the value are added to the end of the TLV. The existence of optional Value words is inferred from the content of the Length field and the value of the P bit.

4.3.1 Generic Context Block

The Generic Context Block creates a 16-octet metadata scratchpad to be used in a deployment-specific manner.

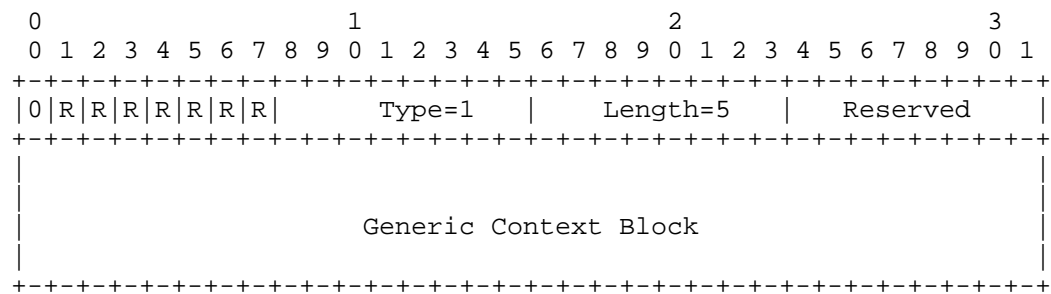


Figure 3: Generic Context Block

4.3.2 Rendered Service Path Identifier

The Rendered Service Path Identifier TLV is used to identify a specific Rendered Service Path (RSP). The Rendered Service Path is the exact sequence of SFFs and SF instances followed for a given flow per [SFC-ARCH].

An RSP may be generated in the control plane with the exact sequence of next-hops and communicated to all involved nodes via control plane or management channels. The exact sequence of SFFs and SF instances is known a priori. This may be viewed as an early binding of SFFs and SF instances to an RSP. This can be satisfied by the Path Identifier alone.

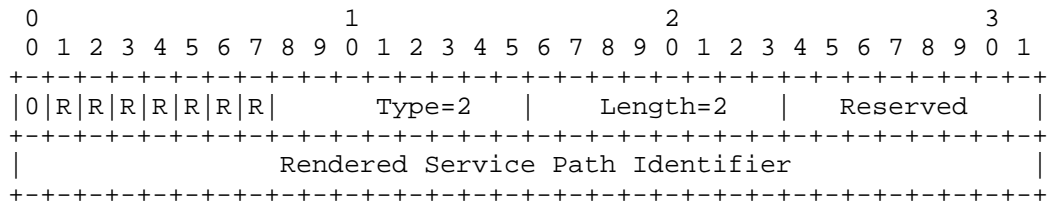


Figure 4: Rendered Service Path Identifier

There is also a need to provide a late binding approach to support load balancing across a group of SF instances, for example, to maintain statefulness. This allows the RSP to be constructed dynamically on a hop-by-hop basis by the the SFFs. The RSP identifies the exact sequence of SF instances to ensure that all traffic for a flow traverses the same SF instances.

When this option is used, a new RSP Identifier is generated for the given SFP ID and is added as an optional metadata TLV to the SCH. The binding of each SFF and SF instance to a RSP is done through the hop-by-hop selection of the SFFs and SF instances as the first packet for a flow traverses the chain.

The packet is sent to each SFF with the SFP ID and the RSP ID. The first SFF will select an instance of the first required SF and bind it to the {SFP ID, RSP ID}. The SFF will then choose a next SFF that hosts at least one instance of the next SF. (The next SFF could be the same SFF). The SF instance selection and binding to the {SFP ID, RSP ID} may be performed as such at each hop.

Once the binding of the SFFs and SF instances is complete, the RSP ID is used to steer all subsequent traffic.

A sufficient number of RSP IDs must be allocated for load balancing purposes.

4.4 Header Associated Operations

The Service Chain Header is processed by SFC-aware entities. These entities include but are not limited to the SFC classifiers, SF nodes, SF forwarding nodes, SFC boundary nodes, and SF proxy nodes. The SCH can be used by any SFC entities. Its use between the SFF and SF is optional and in many cases will be implementation dependent. The typical SFF behavior is simplified due to the presence of the metadata length field in the fixed portion of the header.

These entities can perform the following operations related to the

SCH:

1. Insertion of the Service Chain Header. This occurs at the start of a service chain. An SFC classifier determines which frames/packets are associated with which service chain. The classifier performs this task by matching the incoming frames/packets against certain flow descriptors and assigning them to specific chains. The service classifier MUST then insert appropriate SCH in the frames/packets of a flow that has been assigned to a service chain.
2. Removal of the Service Chain Header. This occurs at the end of the service path. The last Service Network Forwarder (SNF) in the service path MUST remove the SCH from the frames/packets and then forward them to their final destination using its existing transport mechanism. This MUST also be performed by an SF Proxy Node if it is the last forwarding node.
3. Service Path Selection. The Service Network Forwarder (SNF) uses the Service Chain Identification information in the SCH to steer the traffic flow along the SFC path.
4. Service Function Instance Selection. The Service Function Forwarder (SFF) uses the Service Chain Identification information in the SCH to locate the service instance and forward the traffic flow to the service instance. The mapping of the Service Chain Identification to a service instance can be established through the management/control plane, which is out of scope of this document.
5. Service Treatment. The Service Node could use the Service Chain Identification information as well as service metadata in the SCH to locate the service instance associated with the service chain and apply appropriate service treatment.
6. Service Bypass. There are cases, e.g., long-lived flows, where a Service Function does not need to process any further packets for flow and that it should be bypassed. The Service Function signals this by setting the Bypass bit in the SCH of packets that it sends back to the SFF. The flow in question is identified from the encapsulated packet header. When the SFF receives such a packet it marks that flow as an exception flow and does not send any further packets for that flow to the SF but instead forwards the packets to the next SFF or next SF in the chain. The SFF resets the Bypass bit to zero before sending the packet to the next downstream SFF or SF.

5. SCH Encapsulation

The SFC classifier adds the SCH to the original frame or packet (of types defined in [ETYPES]), and then adds the transport header used to forward the frame/packet through the service chain.

The SCH is independent of the underlying transport used to provide reachability amongst the SFC entities. It can be encapsulated inside any transport mechanism. The following examples illustrate how the SCH can be encapsulated in typical transport mechanisms.

5.1 SCH in Overlay GRE Encapsulation

The SCH can be encapsulated in a GRE transport as follows:

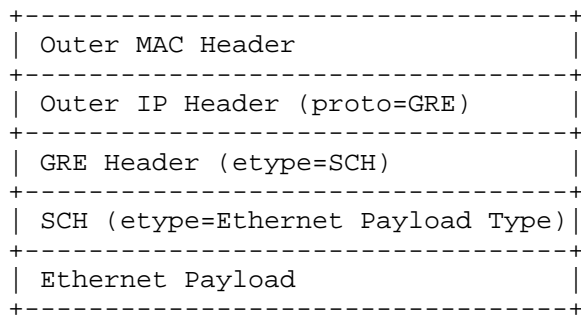


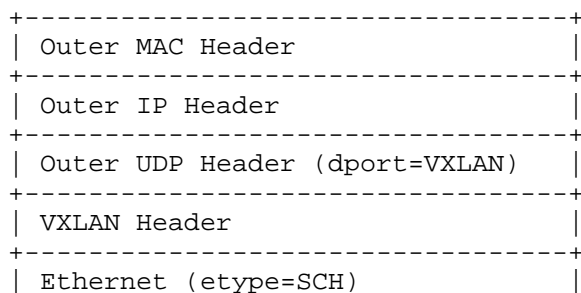
Figure 5: SCH Encapsulation in GRE Transport

Set GRE Header Protocol = SCH Ethertype

SCH inserted between the GRE header and MAC payload

5.2 SCH in Overlay VXLAN Encapsulation

The SCH can be encapsulated in a VXLAN transport as follows:



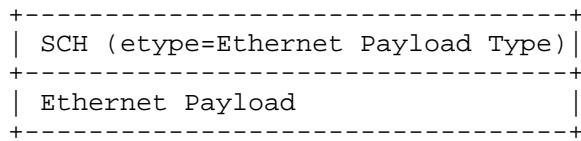


Figure 6: SCH Encapsulation in VXLAN Transport

Set VXLAN's inner Ethernet Ethertype= SCH Ethertype

SCH inserted between the inner Ethernet header and MAC payload

5.3 SCH in IP/UDP Encapsulation

The SCH can be encapsulated in a IP/UDP transport as follows.:

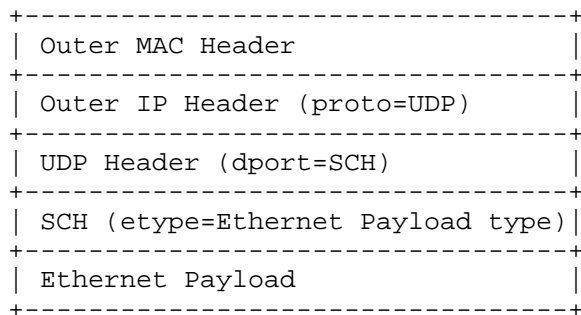


Figure 7: SCH Encapsulation in IP/UDP Transport

Set UDP Header dport = SCH Ethertype, a special UDP port value should be assigned to SCH by IANA

SCH inserted between the IP/UDP header and MAC payload.

5.4 SCH in MAC Encapsulation

The SCH can be encapsulated directly in a MAC transport (VLAN optional) as follows:

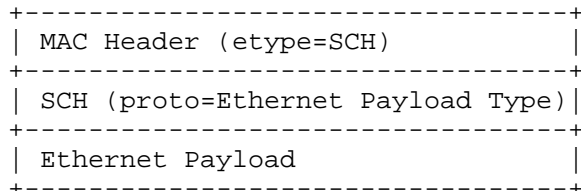


Figure 8: SCH Encapsulation in MAC Transport

Set MAC Header Ethertype = SCH Ethertype

SCH inserted between the MAC header and MAC Payload

6. Security Considerations

Although the SCH can be modified or spoofed by an unauthorized party, various options are available to avoid this.

Existing security protocols RFC 6071 [RFC6071] may be used to encrypt the content of a packet that includes the SCH. It should be noted that encryption and decryption of the SCH will impose a performance penalty. Existing security protocols that provide authenticity and authorization (e.g., RFC 2119 [RFC6071]) can be used.

If possible, the SCH should be used in a controlled network with trusted devices, for example, a data center or a Gi-LAN network, thus reducing the risk of unauthorized header manipulation.

7. IANA and IEEE Considerations

Non-OUI TLV types shall be assigned by IANA.

An EtherType assignment for the SCH will be requested from IEEE.

8. References

8.1 Normative References

- [RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, March 2000.
- [RFC0781] Su, Z., "Specification of the Internet Protocol (IP) timestamp option", RFC 781, May 1981.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6071] Frankel, S. and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap", RFC 6071, February 2011.

8.2 Informative References

- [SFC-ARCH] Halpern, J. et al, Service Function Chaining (SFC) Architecture <<https://datatracker.ietf.org/doc/draft-ietf-sfc-architecture/>>.
- [SFC-FWK] Boucadair, M. et al, Service Function Chaining: Framework & Architecture <<http://datatracker.ietf.org/doc/draft-boucadair-sfc-framework/>>.
- [SFC-META] Rijsman, B. and J. Moisand, Metadata Considerations <<http://http://datatracker.ietf.org/doc/draft-rijsman-sfc-metadata-considerations/>>.
- [SFC-PS] Quinn, P., Ed. and T. Nadeau, Ed., "Service Function Chaining Problem Statement", 2014, <<http://datatracker.ietf.org/doc/draft-ietf-sfc-problem-statement/>>.

9. Appendix A

Appendix A gives some examples of potential metadata TLV types and usage. It is out of scope of the document to define the list of types and usage. For exemplary purposes only, the formats assume globally registered types (i.e., the OUI word is not present).

9.1 OAM Operation

The OAM Operation TLV, used in conjunction with the M-bit in the fixed portion of the SCH, allows a service function to specify an action to be taken by a downstream service function as a result of its service processing.

```

      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-----+-----+-----+-----+-----+-----+-----+-----+
      |0|R|R|R|R|R|R|R|      Type=3      |      Length=1      |      Action      |
      +-----+-----+-----+-----+-----+-----+-----+-----+

```

Service actions could include the following:

- 1 = drop packet
- 2 = redirect flow
- 3 = mirror flow
- 4 = terminate connection

9.2 Service Function Selector

The Service Function Identifier TLV allows a service function to select a specific service function to be used on a downstream service node.

```

      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-----+-----+-----+-----+-----+-----+-----+-----+
      |0|R|R|R|R|R|R|R|      Type=4      |      Length=2      |      Reserved      |
      +-----+-----+-----+-----+-----+-----+-----+-----+
      |                                     Service Function Identifier                                     |
      +-----+-----+-----+-----+-----+-----+-----+-----+

```

9.3 Target Address

The Target Address TLV allows an original destination IP address to be transported across the service chain to the last service function which restores that IP address to the destination IP address of the original packet.

```

      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

```

```

+-----+
|0|R|R|R|R|R|R|R|   Type=5   |   Length=3   |   Reserved   |
+-----+
|                                     Service Function Identifier                                     |
+-----+
|                                     Target IPv4 Address                                     |
+-----+

```

9.4 OAM Service Function Trace

The OAM Service Function List TLV supports OAM functionality and is used to obtain a list of service functions that have been traversed by the packet. Each service function adds its service function identifier to this list if the OAM Operation TLV indicates an SF Identifier.

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+
|0|R|R|R|R|R|R|R|   Type=6   |   Length   |   Reserved   |
+-----+
|                                     Service Function Identifiers                                     |
+-----+
|                                     ...                                     |
+-----+

```

10. Acknowledgments

The authors would like to thank Nicolas Bouthors, Linda Dunbar, Lucy Yong and Kevin Glavin for their review and comments.

Authors' Addresses

Hong (Cathy) Zhang
Huawei US R&D

EMail: cathy.h.zhang@huawei.com

Louis Fourie
Huawei US R&D

EMail: louis.fourie@huawei.com

Ron Parker
Affirmed Networks

EMail: ron_parker@affirmednetworks.com

Myo Zarny
Goldman Sachs

EMail: myo.zarny@gs.com