

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 19, 2014

A. Beliveau
Ericsson
October 16, 2013

Service Function Chaining Architecture
draft-beliveau-sfc-architecture-00

Abstract

This document describes an architecture for Service Function Chaining. It addresses operational aspects of Service Function Chaining such administration of Service Function Chains, network and forwarding principles. It also covers architectural principles to support scale-in and scale-out of Service Functions.

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 April 19, 2014.

Copyright Notice

Copyright (c) 2013 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	3
3. Architecture principles	4
3.1. Introduction	4
3.2. Service Function Chain	4
3.3. Service Chaining Infrastructure	5
3.4. Service Function scaling	7
3.5. Service Function Classification	7
3.6. Service Chaining Controller	7
4. Acknowledgements	8
5. IANA Considerations	8
6. Security Considerations	8
7. References	9
7.1. Normative References	9
7.2. Informative References	9
Author's Address	10

1. Introduction

This document describes an architecture for Service Function Chaining (SFC). It describes components and architecture principles to provide Service Function Chaining in a network.

2. Terminology

This document makes use of the following terms:

Service Function(SF): An application or service which performs specific treatments when packets traverses it. A non-exhaustive list of Service Functions include: firewall (e.g.,[RFC6092]), DPI (Deep Packet Inspection), NAT44 [RFC3022], NAT64 [RFC6146], HOST_ID injection, HTTP Header Enrichment function, load-balancer, etc. The exact definition of each Service Function is specific to each Service Function provider.

Service Function Identity(SF-ID): A unique identifier which represents each SF in a network. SF-ID is unique within each SFC network and does not need to be globally unique. Even if multiple instances of the same Service Function are available in an SFC network, the same SF-ID is used to identify the Service Function.

Service Function Locator(SF-Loc): A unique name or address which identifies each Service Function. When multiple instances of the same Service Function are available in an SFC network, each instance has its own Service Function Locator.

Service Function Chain (SFC): An ordered list of SF which should be traversed.

Service Function Chain Identity (SC-ID): A unique identifier which represents each SFC in a network. SC-ID is unique within each SFC network and does not need to be globally unique.

Service Function Chain Enforcement Point (SCEP): A node which is able to guarantee that a list of SF will be traversed in a specific order for flows which are associated with such SF chain.

Service Chaining Infrastructure Network (SC-IN): is formed by a one or more SCEP nodes connected together.

Service Chaining Classification Function (SC-CL): A logical function part of a SCEP node. SC-CL is mandatory to execute when packets enter the SC-IN (ingress).

Service Chaining Forwarding Function (SC-FWD): A logical function part of a SCEP node. It is responsible for forwarding packets to SF, forwarding packets to other SCEP nodes and to remove SC-IN specific information from packets when exiting (egress) SC-IN. SC-FWD is mandatory in all SCEP nodes.

Service Function Path: Specific path taken by packets through SC-IN. Service Function Path includes specific SCEP nodes and SF nodes traversed by an individual flow.

3. Architecture principles

3.1. Introduction

The concept of Service Function Chaining consists of applying a number of Service Functions in a specific order. The proposed architecture for Service Function Chaining ties together four different mechanisms to guarantee the execution of Service Functions in a specific order.

Those four separate mechanisms are:

- o A mechanism to specify a Service Function Chain (SFC) as an ordered list of Service Functions.
- o A mechanism to deliver packets between SF instances in the specified order: Service Chaining Infrastructure (SC-IN).
- o A mechanism to specify which packets should be associated with a specific Service Function Chain: SFC classification.
- o A mechanism to support scaling in/out the number of instances of each SF.

3.2. Service Function Chain

A Service Function Chain (SFC) consists of an ordered list of Service Function (SF). Each SF is defined by an identifier which is unique within an administrative domain (SF-ID). No IANA registry is required to store the identity of SFs.

Multiple Service Function Chains can exist in the same administrative domain. Each Service Function Chain (SFC) is defined by an identifier which is unique within an administrative domain (SC-ID). No IANA registry is required to store the identity of Service Function Chains.

As no information about topology, SF classification or SF scaling is represented in the SF chain definition therefore, Service Function Chains are independent from changes in topology, classification or scaling instances of a Service Function.

Here is a some examples of Service Function Chains:

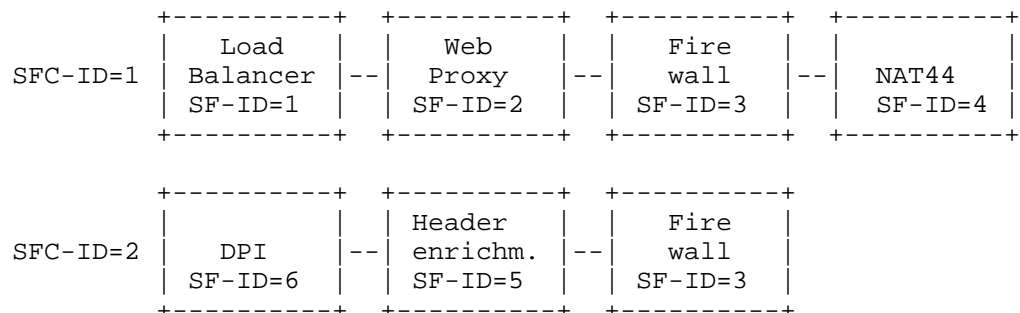


Figure 1: Service Function Chain examples

3.3. Service Chaining Infrastructure

The Service Chaining Infrastructure (SC-IN) consists of Service Chaining Enforcement Points (SCEP) and Service Functions interconnected as a network. An SCEP node contains a forwarding function (SC-FWD) and optionally a classification function (SC-CL).

Service Chaining Infrastructure (SC-IN) is illustrated in Figure 2.

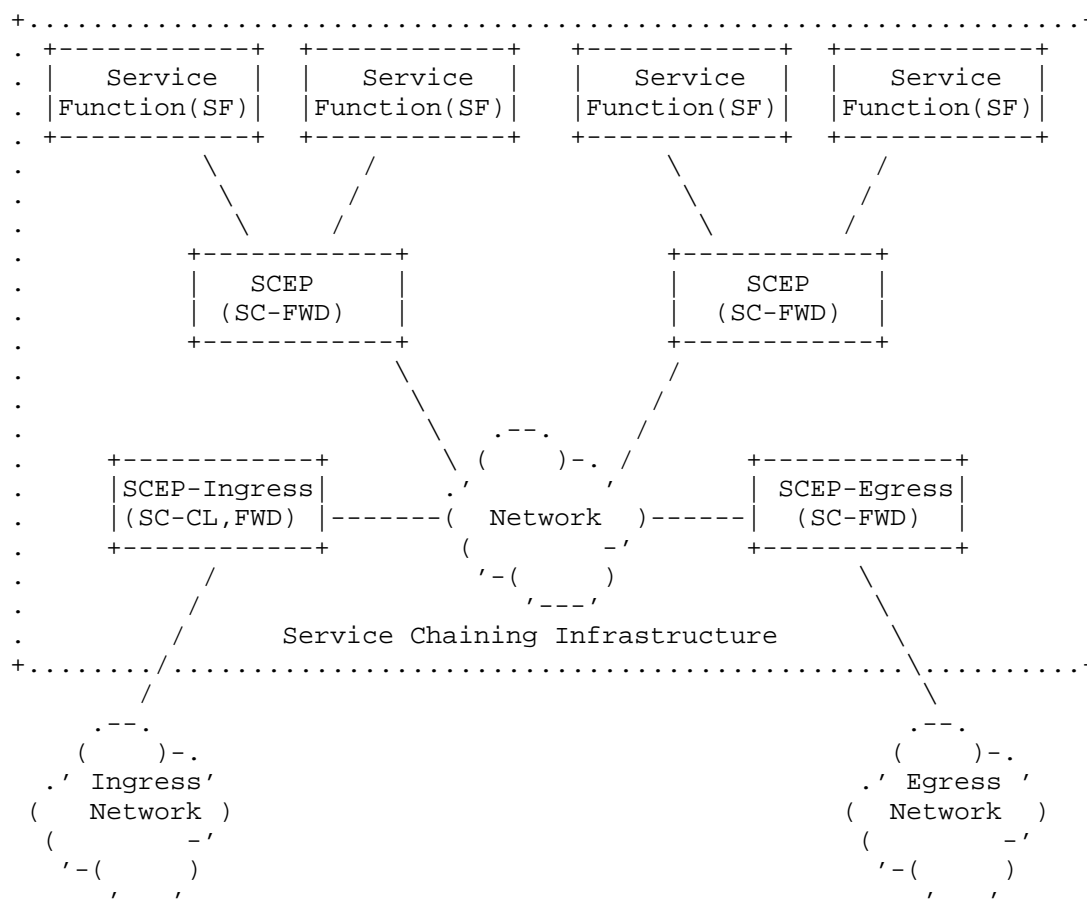


Figure 2: Service Chaining Infrastructure

SCEP can directly reach other SCEP nodes within the SC-IN. Service Functions (SF) are directly connected to an SCEP. To enforce the execution of SF in the specified order, Service Functions (SF) cannot communicate directly between each other without going through an SCEP. One or many SFs can be attached to the same SCEP.

When entering an SC-IN, an ingress SCEP which contains an SC-CL will map packets into a specific Service Function Path. Once this mapping is done, the SC-FWD will determine the locator for the next SF on the Service Function Path and forward the packet to the SF to be invoked.

3.4. Service Function scaling

Multiple instances of the same Service Function can exist in the SC-IN. Each new instances of a SF, when created, will be attached to an SCEP in the SC-IN and a unique locator (SF-Loc) will be allocated to it.

When instances of a Service Function needs to be removed, the Service Function Controller will ensure that no packet can be forwarded to the instance of Service Function to be removed. Once done, the Service Function instance can be removed.

3.5. Service Function Classification

In order to direct specific packets to follow a certain Service Function Path, SC-CL will analyse the packet headers and determine which Service Function Path should be followed. A Service Function Path is a list of Service Function locators (SF-Loc) for the specific instances of SF which constitutes a SFC.

Once the Service Function Path is determined, the Service Chaining Forwarding function (SC-FWD) will determine the next Service Function and forward the traffic to it.

3.6. Service Chaining Controller

The Service Function Chaining Controller is responsible to configure SCEP nodes in the Service Chaining Infrastructure. It is the controller which ties together Service Function Chains/Paths, the topology of the Service Chaining Infrastructure, the locating information of SF instances as they are being scaled in/out and Service Chaining Classification rules.

It is responsible to guarantee the consistency of the configuration of SCEP across the SC-IN. Service Chaining Controller is illustrated in Figure 3.

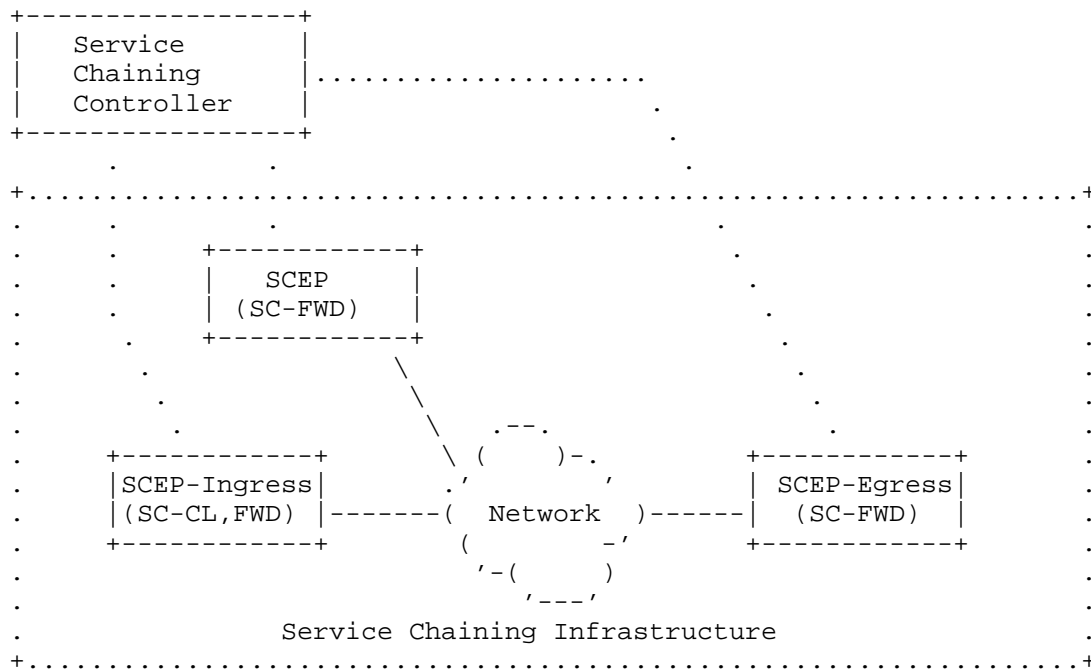


Figure 3: Service Chaining Controller

4. Acknowledgements

This template was derived from an initial version written by Pekka Savola and contributed by him to the xml2rfc project.

5. IANA Considerations

This document has no IANA actions.

6. Security Considerations

SFC must address at least the following security considerations:

- o Secure and authenticate communication between controller and SCEP nodes
- o Authenticate communication between SF and SCEP node.

- o Isolate SC-IN network when infrastructure is shared with nodes which are not SCEP nodes.
- o Protect interface at border of SC-IN (ingress/egress SCEP) against fraudulent usage.
- o Protect SFC specific protocol/metadata information against fraudulent usage.
- o When an SCEP participate in multiple networks, isolation between them.
- o Protect interface between SF and SCEP node against fraudulent usage.

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.
- [RFC6092] Woodyatt, J., "Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service", RFC 6092, January 2011.
- [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.

7.2. 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-00 (work in progress), October 2013.
- [I-D.boucadair-sfc-requirements]
Boucadair, M., Jacquenet, C., Jiang, Y., Parker, R., and C. Pignataro, "Requirements for Service Function

Chaining", draft-boucadair-sfc-requirements-00 (work in progress), October 2013.

[I-D.quinn-sfc-problem-statement]

Quinn, P., Guichard, J., Surendra, S., Agarwal, P., Manur, R., Chauhan, A., Leymann, N., Boucadair, M., Jacquenet, C., Smith, M., Yadav, N., Nadeau, T., Gray, K., McConnell, B., and K. Kevin, "Service Function Chaining Problem Statement", draft-quinn-sfc-problem-statement-00 (work in progress), October 2013.

Author's Address

Andre Beliveau
Ericsson
8400 Decarie Blvd.
Town of Mount Royal, QC
Canada

Phone: +1 514 345 2708
Email: andre.beliveau@ericsson.com

SFC
Internet-Draft
Intended status: Standards Track
Expires: August 16, 2014

M. Boucadair
C. Jacquenet
France Telecom
R. Parker
Affirmed Networks
D. Lopez
Telefonica I+D
J. Guichard
C. Pignataro
Cisco Systems, Inc.
February 12, 2014

Service Function Chaining: Framework & Architecture
draft-boucadair-sfc-framework-02

Abstract

IP networks rely more and more on the combination of advanced functions (besides the basic routing and forwarding functions) for the delivery of added value services. This document defines a reference architecture and a framework to enforce Service Function Chaining (SFC) with minimum requirements on the physical topology of the network.

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 16, 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	3
1.1. On the Proliferation of Service Functions	3
1.2. Scope	4
1.3. Objectives	4
1.4. Assumptions	4
1.5. Rationale	5
2. Terminology	6
3. Functional Elements	7
4. SFC Provisioning	8
4.1. Assign Service Function Identifiers	8
4.2. Service Function Locator	8
4.3. Service Function Discovery	8
4.4. Building Service Function Maps	9
4.5. Building Service Function Chaining (SFC) Policy Tables	9
5. Theory Of Operation	11
5.1. SFC Boundary Node	11
5.2. SFC Classifier	11
5.3. SFC Ingress Node	12
5.4. SFC Egress Node	13
5.5. SF Node	13
5.6. Intermediate Nodes	14
6. Fragmentation Considerations	14
7. Differentiated Services	14
8. ECN (Explicit Congestion Notification) Considerations	14
9. Design Considerations	14
9.1. Transmit A SFC Map Index In A Packet	15
9.1.1. SFC Map Index	15
9.1.2. Where To Store SFC Map Indexes In A Packet?	15
9.2. Steer Paths To Cross Specific SF Nodes	15
10. Deployment Considerations	15
10.1. Generic Requirements	15

10.2.	Deployment Models	16
10.2.1.	1.1. Proxy Node for Legacy Service Functions	16
10.3.	On Service Function Profiles (a.k.a., Contexts)	17
10.4.	SF Node is also a Classifier	18
10.5.	SFs within the Same Subnet	19
10.6.	Service Function Loops	19
10.7.	Lightweight SFC Policy Table	20
10.8.	Liveness Detection Of SFs By The PDP	21
11.	IANA Considerations	21
12.	Security Considerations	21
13.	Contributors	22
14.	Acknowledgments	22
15.	References	22
15.1.	Normative References	22
15.2.	Informative References	22

1. Introduction

1.1. On the Proliferation of Service Functions

IP networks rely more and more on the combination of advanced functions (besides the basic routing and forwarding functions) for the delivery of added value services. Typical examples of such functions include firewall (e.g., [RFC6092]), DPI (Deep Packet Inspection), LI (Lawful Intercept) module, NAT44 [RFC3022], NAT64 [RFC6146], DS-Lite AFTR [RFC6333], NPTv6 [RFC6296], HOST_ID injection, HTTP Header Enrichment function, TCP tweaking and optimization function, transparent caching, charging function, load-balancer, etc.

Such advanced functions are denoted SF (Service Function) in this document.

The dynamic enforcement of a SF-derived, adequate forwarding policy for packets entering a network that supports such advanced Service Functions has become a key challenge for operators and service providers. SF-inferred differentiated forwarding is ensured by tweaking the set of Service Functions to be invoked. How to bind a flow of packets that share at least one common characteristic to a forwarding plane is policy-based, and subject to the set of SF functions that need to be solicited for the processing of this specific flow.

Service Providers need to rationalize their service delivery logics and master its underlying complexity.

The overall problem space is described in [I-D.ietf-sfc-problem-statement]. A companion document that lists a set of requirements is available at [I-D.boucadair-sfc-requirements].

1.2. Scope

This document defines a framework to enforce Service Function Chaining (SFC) with minimum requirements on the physical topology of the network. The proposed solution allows for differentiated forwarding: packets are initially classified at the entry point of an SFC-enabled network, and are then forwarded according to the ordered set of SF functions that need to be activated 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 (e.g., to rationalize the proliferation of advanced features at the Gi Interface [RFC6459]).

Considerations related to the chaining of Service Functions that span domains owned by multiple administrative entities is out of scope. Note, a single administrative entity may manage multiple domains.

1.3. Objectives

The main objectives of the proposed framework are listed below:

- o Create service-inferred forwarding planes.
- o Efficiently master the chained activation of Service functions, regardless of the network topology and routing policies.
- o Allow packets to be forwarded to the required Service Functions without changing the network topology or overlay transports necessary for packet delivery to/from Service Functions.
- o Allow for differentiated packet forwarding by selecting the set of Service functions to be invoked.
- o Allow to easily change the sequentiality of the activation of Service functions to be invoked.
- o Allow to easily change the set of Service functions to be invoked.
- o Ease management (including withdrawal) of Service functions and minimize any subsequent topology update.
- o Automate the overall process of generating and enforcing policies to accommodate a set of network connectivity service objectives.

1.4. 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 nor 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 nor standard SF chaining logic. The ordered set of SFs that need to be activated to deliver a given connectivity service is specific to each administrative entity.
- o The chaining of SFs and the criteria to invoke some of them are specific to each administrative entity that operates the SF-enabled network (also called administrative domain).
- o SF chaining logic and related policies should not be exposed outside a given administrative domain.
- o Several SF chaining logics can be simultaneously enforced 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 the traffic to a given SF chaining is policy-based.

1.5. Rationale

Given the assumptions listed in Section 1.4, the rationale of the framework is as follows:

- o The framework separates the dynamic provisioning of required SF functions from packet handling operations (e.g., forwarding decisions).
- o The technical characterization of each SF is not required to design the SFC architecture and SFC operations.
- o No IANA registry is required to store the list of SFs. In particular, assignment of identifiers, header fields, or any other indication of the Service Function Chain, are all strictly local in scope. An identifier assigned in one administrative domain will not indicate the same set of SFs in another administrative domain.
- o No IANA registry is required to store the SF chaining candidates. The set of SFCs are local to each administrative domain, and are as such not global.
- o No specific SF chaining is assumed. The description of SF chains is an information that will be processed by the nodes that participate to the delivery of a network service. The set of listed/chained SF functions is generated by each administrative entity operating the network.
- o SF handling is policy-based: SF chains can be updated or deleted, new SFs can be added without any impact on existing SFs, etc. In

particular, this design is compliant with the global framework discussed in [I-D.sin-sdnrg-sdn-approach].

- o For the sake of efficiency, policy enforcement is automated (but policies can be statically enforced, for example).
- o To minimize fragmentation, a minimal set of information needs to be signaled (possibly in data packets).
- o Advanced features (e.g., load balancing) are also described and may be configured according to policies that can be service-specific. Policy decisions are made by a Policy Decision Point [RFC2753] and the solicited enforcement points are responsible for applying these decisions, whatever the objective to achieve.
- o SFs can be embedded in nodes that intervene in the transport service or supported by dedicated nodes (e.g., dedicated servers). The decision to implement one of these two models (or a combination thereof) is deployment-specific and it is orthogonal to the overall procedure.
- o Multiple SFC-enabled domains can be deployed within the same administrative domain. Nodes are provisioned with the policy table of the SFC-enabled domain they belong to.
- o The overall consistency of the differentiated forwarding policy is ensured by the PDP.
- o The PDP can be responsible to enforce other policies than those described in the SFC Policy Tables.

2. Terminology

This document makes use of the following terms:

- o SF (Service Function): refers to a function which is enabled in the network operated by an administrative entity. One or many Service Functions can be involved in the delivery of added-value services. A non-exhaustive list of Service Functions include: firewall (e.g., [RFC6092]), DPI (Deep Packet Inspection), LI (Lawful Intercept) module, NAT44 [RFC3022], NAT64 [RFC6146], DS-Lite AFTR [RFC6333], NPTv6 [RFC6296], HOST_ID injection, HTTP Header Enrichment function, TCP optimizer, load-balancer, etc. This document does not make any assumption in the OSI Layer on which the Service Function acts on; the exact definition of each Service Function is deployment-specific.
- o SFC-enabled domain: denotes a network (or a region thereof) that implements SFC.
- o SF Identifier: is a unique identifier that unambiguously identifies a SF within a 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; other formats can be used. SF Identifiers are not

required to be globally unique nor be exposed to or used by another SF-enabled domain.

- o SF Map: refers to an ordered list of SF identifiers. Each SF Map is identified with a unique identifier called SF Map Index.
- o SFC Policy Table: is a table containing a list of SF Maps, SFC classification rules and Locators for all SF Nodes. A SFC Policy Table may contain a default SF Map.
- o SF Locator: A SF Node identifier used to reach the said SF node. A locator is typically an IP address or a FQDN.
- o Legacy Node (Node for short): refers to any node that is not a SF Node nor a SFC Boundary Node. This node can be located within a SFC-enabled domain or outside a SFC-enabled domain.
- o SF Proxy Node: a Network Element along the data path, to enforce SFC functions on behalf of legacy SF nodes.

3. Functional Elements

The following functional elements are defined in this document:

- o 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.
- o SFC Egress Node (or Egress Node): denotes a SFC Boundary Node that handles traffic which leaves the SFC-enabled domain the Egress Node belongs to.
- o SFC Ingress Node (or Ingress Node): denotes a SFC Boundary Node that handles traffic which enters the SFC-enabled domain the ingress Node belongs to.
- o SF Node: denotes any node within an SFC-enabled domain that embeds one or multiple SFs.
- o SFC Classifier (or Classifier): an entity that classifies packets for service chaining according to classification rules defined in a SFC Policy Table. Packets are then marked with the corresponding SF Map Index. SFC Classifier is embedded in a SFC boundary (Ingress) Node. A SFC Classifier may be considered as a Service Function, and therefore be uniquely identified by a dedicated SF Identifier.

4. SFC Provisioning

It is out of scope of this document to discuss SF-specific policy enforcement; only SFC considerations are elaborated.

4.1. Assign Service Function Identifiers

The administrative entity that operates a SFC-enabled domain maintains a local repository that lists the enabled SFs. This administrative entity assigns a unique SF identifier for each SF type.

SF identifiers are structured as character strings. SF identifiers are case-sensitive.

The main constraint on the format is that two SFs MUST be assigned with different SF identifiers if they do not provide the exact same function, or do provide the same function but are unable to differentiate packets based on policies provisioned to the SF using an appropriate mechanism.

4.2. Service Function Locator

A SF may be embedded in one or several SF Nodes. The SF locator is typically the IP address or the FQDN to reach a given SF.

The use of an IP address is RECOMMENDED to avoid any extra complexity related to the support of name resolution capabilities in SF Nodes. Resolution capabilities are supported by the PDP (Policy Decision Point). In the rest of the document, we assume a SF locator is structured as an IP address (IPv4 or IPv6).

A SF can be reached by one or more locators. When multiple SF locators are in use, the locator to be used to reach a given SF can be driven by the PDP, a SF in a SFC, result of a load-balancing heuristic, etc.

4.3. Service Function Discovery

The local repository that lists the enabled SFs within an SFC-enabled domain may be built as a direct input from the administrative entity, or they may be discovered dynamically through appropriate protocol discovery means.

Whichever method is selected by the administrative entity is a local decision and is therefore outside the scope of this document. Any Service Function Discovery solution must comply with the requirements identified in [I-D.boucadair-sfc-requirements].

4.4. Building Service Function Maps

Added-value services delivered to the end-user rely on the invocation of several SFs. For each of these services, the administrative entity that operates an SFC-enabled domain builds one or several SF Maps. Each of these maps characterizes the list of SFs to be invoked with their exact invocation order.

Each SF Map is unambiguously identified with a unique identifier called the SF Map Index. The SF Map Index **MUST** be described as an unsigned integer.

Distinct chains can be applied for inbound and outbound traffic. The directionality of traffic is not included as an attribute of the SF Map, but it may be implicitly described by using two SF Maps installed and maintained in the SFC Policy Table. In such case, incoming packets would be marked with Index_1 for example, while outgoing packets would be forwarded according to a distinct SF Map identified with Index_2.

An example of SF Map to handle IPv6 traffic destined to an IPv4 remote server is defined as follows:

```
{15, {IPv6_Firewall, HOST_ID_Inject, NAT64}}.
```

To handle incoming packets destined to the same IPv6 host, the following SF Map can be defined:

```
{10, {IPv4_Firewall, NAT64}}.
```

4.5. Building Service Function Chaining (SFC) Policy Tables

A PDP (Policy Decision Point, [RFC2753]) is the central entity which is responsible for maintaining SFC Policy Tables (Figure 1), and enforcing appropriate policies in SF Nodes and SFC Boundary Nodes (Figure 1). PDP-made decisions can be forwarded to the participating nodes by using a variety of protocols (e.g., NETCONF [RFC6241]).

One or multiple SFC-enabled domains may be under the responsibility of the same PDP. Delimiting the scope of each SFC-enabled domain is under the responsibility of the administrative entity that operates the SF-enabled network.

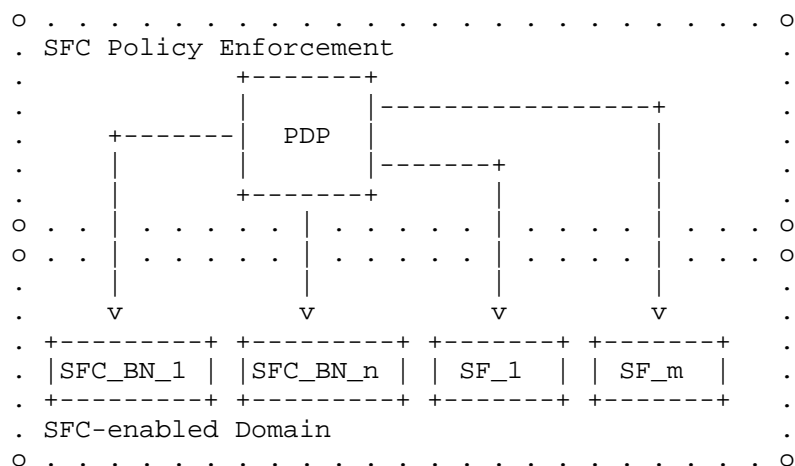


Figure 1: SFC Policy Enforcement Scheme.

The SF Node MUST be provisioned with the following information:

- o Local SF Identifier(s): This information is required for an SF to identify itself within an SF Map.
- o List of SF Maps: The PDP may configure the full list (default mode) or only as subset of SF Maps in which SF(s) supported by the SF Node is involved (see Section 10.7).
- o List of SF Locators: The PDP may configure the full list of locators (default mode) or only the locators of next hop SFs of SF Maps in which SF(s) supported by the local SF node is involved (see Section 10.7).

[DISCUSSION NOTE: Discuss if we maintain both forms of the SFC Policy table (full and lite) or select only one of them.]

Likewise, the SFC Boundary Node MUST be provisioned with the following information:

- o List of SF Maps
- o List of SF Locators
- o List of SF Map Classification Rules (see Section 5.2).

In addition to the SFC Policy Table, other SF-specific policies can be installed by the PDP (e.g., configure distinct user profiles, activate specific traffic filters, configure traffic conditioners, etc.).

Policies managed by the PDP may be statically instantiated or dynamically triggered by external means (e.g., a AAA server).

In the event of any update (e.g., define a new SF Map, delete an SF Map, add a new SF Locator, update classification policy), the PDP MUST forward the updated policy configuration information in all relevant SF Nodes and SFC Boundary Nodes.

Distributing the load among several SF Nodes supporting the same SF can be driven by the PDP. Indeed, the PDP can generate multiple classification rules and SF Maps to meet some load-balancing objectives.

Load balancing may also be achieved locally by an SF Node. If the SF Node, SF Classifier, or SF Boundary Node has a table that provides the SF locator(s) of SF Nodes that provide a particular SF then it is possible to make that local load balancing decision.

The processing of packets by the nodes that belong to a SFC-enabled domain does not necessarily require any interaction with the PDP, depending on the nature of the SF supported by the nodes and the corresponding policies to be enforced. For example, traffic conditioning capabilities [RFC2475] are typical SF functions that may require additional solicitation of the PDP for the SF node to decide what to do with some out-of-profile traffic.

5. Theory Of Operation

The behavior of each node of a SFC-enabled domain is specified in the following sections. We assume that the provisioning operations discussed in Section 4 have been successful (i.e., SF functions have been adequately configured according to the SFC-specific policy to be enforced).

5.1. SFC Boundary Node

SFC Boundary Nodes act both as a SFC Ingress Node and as a SFC Egress Node for the respective directions of the traffic.

Traffic enters a SFC-enabled domain at a SFC Ingress Node (Section 5.3) and exits the domain at a SFC Egress Node (Section 5.4).

5.2. SFC Classifier

The SFC Classifier classifies packets based on (some of) the contents of the packet. Particularly, it classifies packets based on the possible combination of one or more header fields, such as source address, destination address, DS field, protocol ID, source port and destination port numbers, and any other information.

Each SF Map Classification Rule MUST be bound to one single SF Map (i.e., the classification rule must include only one SF Map Index).

5.3. SFC Ingress Node

When a packet is received through an interface of the SFC Ingress Node that connects to the outside of the SFC domain, the Ingress Node MUST:

- o Inspect the received packet and check whether any existing SF Map Index is included in the packet.
 - * The SFC Ingress Node SHOULD be configurable with a parameter to indicate whether received SF Map Index is to be preserved or striped. The default behavior is to strip any received SF Map Index.
 - * Unless explicitly configured to trust SF Map index, The SFC Ingress Node MUST strip any existing SF Map Index if the packet is received from an SFC-enabled domain that has not explicitly been designated as "trusted".
- o Check whether the received packet matches an existing classification rule (see Section 5.2).
- o If no rule matches, forward the packet to the next hop according to legacy forwarding behavior (e.g., based upon the IP address conveyed in the DA field of the header).
- o If a rule matches, proceed with the following operations:
 - * Retrieve the locator of the first SF as indicated in the SF Map entry the rule matches. If multiple locators are available, the selection can be based on local criteria (e.g., the closest /best path).
 - * Check whether the corresponding SF node is an immediate (L3) neighbor.
 - + If so, update the packet with the SF Map Index of SF Map entry it matches and then forward the packet to the corresponding SF Node.
 - + If not, (1) encapsulate the original packet into a new one that will be forwarded to the corresponding SF node, (2) update the encapsulated packet with the SF Map Index of SF Map entry it matches, and (3) forward the packet to the next hop to reach the first SF node.

As a result of this process, the packet will be sent to an SF Node or an Intermediate Node.

5.4. SFC Egress Node

When a packet is received through an interface that connects the SFC Egress Node to its SFC domain, the Egress Node MUST:

- o Strip any existing SF Map Index.
- o Forward the packet according to legacy forwarding policies.

5.5. SF Node

This section assumes the default behavior is each SF Node does not embed a Classifier as discussed in Section 10.4.

When a packet is received by a SF Node, the SF Node MUST:

- o Check whether the packet conveys a SF Map Index.
- o If no SF Map Index is included, forward the packet according to legacy forwarding policies.
- o If the packet conveys a SF Map Index,
 - * Retrieve the corresponding SF Map from the SFC Policy Table. If no entry is found in the table, forward the packet according to legacy forwarding policies.

[DISCUSSION NOTE: Another design choice is to drop the packet and send a notification to the PDP. The justification for avoiding to drop the packet is that an SF can be part of the forwarding path of an SFC to which it does not belong to.]

- * If an entry is found in the SFC Policy Table, check whether the local SF Identifier is present in the SF Map:

- + If not, forward the packet according to legacy forwarding policies.

[DISCUSSION NOTE: One would argue the packet should be dropped. The justification for avoiding to drop the packet is that an SF can be part of the forwarding path of an SFC to which it does not belong to + the SF node is provisioned with the full SFC Policy Table.]

- + If so, the packet is decapsulated (if needed) and then presented as an input to the local SF. In case several SFs are co-located in the same node, the packet is processed by all SFs indicated in the SF Map. Once the packet is successfully handled by local SF(s), the packet is forwarded to the next SF Node in the list or to an intermediate node (if the local SF Node is the last element in the SF Map). If the local SF node is not the last one in the SF Map, it

retrieves the next SF Node from the list, retrieve its locator for the SFC Policy Table, and forwards the packet to the next hop. If the local SF Node is the last element in the SF Map, it forwards the packet to the next hop according to legacy forwarding policies.

5.6. Intermediate Nodes

An Intermediate Node is any node that does not support any Service Function and which is located within a SFC-enabled domain.

No modification is required to intermediate nodes to handle incoming packets. In particular, routing and forwarding are achieved using legacy procedures.

6. Fragmentation Considerations

If adding the Service Chaining Header would result in a fragmented packet, the classifier should include a Service Chaining Header in each fragment. Doing so would prevent SF Nodes to dedicate resource to handle fragments.

7. Differentiated Services

When encapsulating an IP packet, the Ingress Node and each SF Node SHOULD use its Diffserv Codepoint (DSCP, [RFC2474]) to derive the DSCP (or MPLS Traffic-Class Field) of the encapsulated packet.

Generic considerations related to Differentiated Services and tunnels are further detailed in [RFC2983].

8. ECN (Explicit Congestion Notification) Considerations

When encapsulating an IP packet, the Ingress Node and each SF Node SHOULD follow [RFC6040] for ECN re-marking purposes.

9. Design Considerations

This section discusses two main protocol issues to be handled in order to deploy SFC.

A detailed design analysis is documented in [I-D.boucadair-sfc-design-analysis].

9.1. Transmit A SFC Map Index In A Packet

9.1.1. SFC Map Index

A SFC Map Index is an integer that points to a SF Map.

In order to avoid all nodes of a SFC-enabled domain to be SF-aware, this specification recommends to undertake classifiers at boundary nodes while intermediate nodes forward the packets according to the SF Map Index conveyed in the packet (SF Node) or according to typical forwarding policies (any SF-unaware node).

An 8-bit field would be sufficient to accommodate deployment contexts that assume a reasonable set of SF Maps. A 16-bit (or 32-bit) field would be more flexible (e.g., to accommodate the requirement discussed in Section 10.3).

9.1.2. Where To Store SFC Map Indexes In A Packet?

SF Map Indexes can be conveyed in various locations of a packet:

- o At L2 level
- o Define a new IP option or a new IPv6 extension header
- o Use IPv6 Flow Label
- o Use MPLS Label
- o Re-use an existing field (e.g., DS field)
- o TCP option
- o GRE Key
- o Define a new shim
- o Etc.

9.2. Steer Paths To Cross Specific SF Nodes

A SFC Ingress Node or a SF Node MUST be able to forward a packet that matches an existing SF Map to the relevant next hop SF Node. The locator of the next SF is retrieved from the SFC Policy Table. In case the next SF Node in the list is not an immediate (L3) neighbor, a solution to force the packet to cross that SF Node MUST be supported.

10. Deployment Considerations

10.1. Generic Requirements

The following deployment considerations should be taken into account:

- o Avoid inducing severe path stretch compared to the path followed if no SF is involved.

- o Minimize path computation delays: due to the enforcement of classification rules in all participating nodes, misconception of Service function chaining, inappropriate choice of nodes elected to embed Service functions, etc., must be avoided.
- o Avoid SF invocation loops: the design of SF chainings should minimize as much as possible SF invocation loops. In any case, forwarding loops must be avoided.

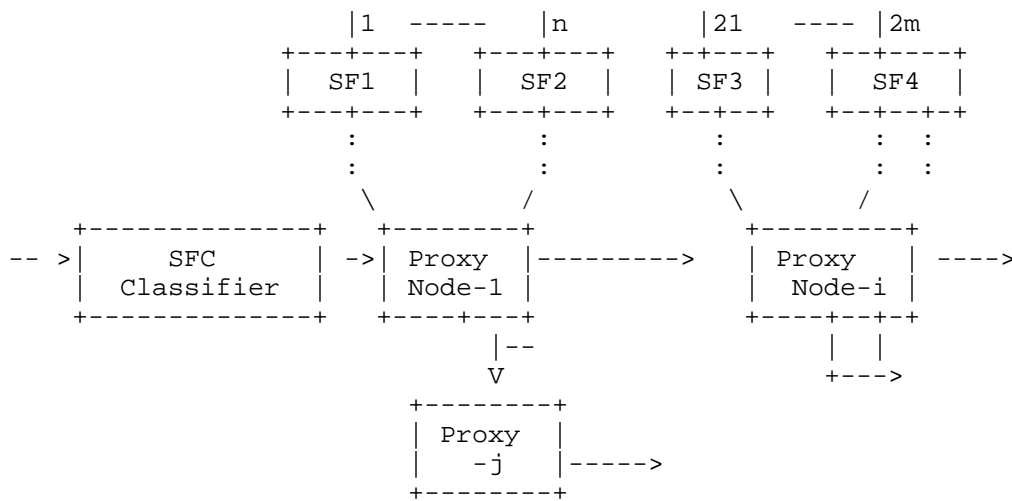
10.2. Deployment Models

Below are listed some deployment model examples:

1. A full marking mechanism: Ingress nodes perform the classification and marking functions. Then, involved SF Nodes process received packets according to their marking.
2. SF node mechanism, in which every SF Node embeds also a classifier, and the ingress node only decides the first node to forward to. Packets are forwarded at each node according to local policies. No marking is required when all SFs are co-located with a classifier. This model suffers from some limitations (see Section 10.4).
3. A router-based mechanism: All SF Nodes forward packets once processed to their default router. This default router is responsible for deciding how the packet should be progressed at each step in the chain. One or multiple routers can be involved in the same Service Function Chain.
4. A combination thereof.

10.2.1. 1.1. Proxy Node for Legacy Service Functions

It is not uncommon to have multiple legacy service nodes located in close vicinity with a service chain proxy node, or one to two links away. The following figure depicts typical network architecture for chaining those service nodes that are not aware of service layer encapsulation.



Various deployment options can be envisaged:

1. Upgrade legacy service nodes to support required SFC functionalities.
2. Enable Proxy Service Nodes to involve these legacy nodes in instantiated SFCs.
3. Exclude legacy service nodes from a SFC domain.

It is up to the responsibility of each Service Provider to decide which option to deploy within its networks.

10.3. On Service Function Profiles (a.k.a., Contexts)

Service Functions may often enforce multiple differentiated policy sets. These policy sets may be coarsely-grained or fine-grained. An example of coarsely-grained policy sets would be an entity that performs HTTP content filtering where one policy set may be appropriate for child users whereas another is appropriate for adult users. An example of finely-grained policy sets would be PCEF (3GPP Policy Control Enforcement Function) that has a large number of differentiated QoS and charging profiles that are mapped on a per-subscriber basis.

The Service Function Chaining mechanism directly support coarsely-grained differentiated policy sets and indirectly support finely-grained differentiated policy sets.

From a Service Function Chaining perspective, each coarsely-grained policy set for a Service Function will be considered as a distinct logical instance of that Service Function. Consider the HTTP content filtering example where one physical or virtual entity provides both child and adult content filtering. The single entity is represented as two distinct logical Service Functions, each with their own Service Function Identifier from a chaining perspective. The two (logical) Service Functions may share the same IP address or may have distinct IP addresses.

Finely-grained policy sets, on the other hand, would unacceptably explode the number of distinct Service Chains that were required with an administrative domain. For this reason, Service Functions that utilize finely-grained policy sets are represented as a single Service Function that has its own internal classification mechanism in order to determine which of its differentiated policy sets to apply. Doing so avoids from increasing the size of the SFC Policy Table.

The threshold, in terms of number of policies, between choosing the coarsely-grained policy or finely-grained policy technique is left to the administrative entity managing a given domain.

[DISCUSSION NOTE: This section will be updated to reflect the conclusions of the discussions from the design analysis draft.]

10.4. SF Node is also a Classifier

If SF Nodes are also configured to behave as Classifiers, the SF Map Index is not required to be explicitly signalled in each packet. Concretely, the SFC Policy Table maintained by the SF Node includes classification rules. These classification rules are enforced to determine whether the local SF must be involved. If an incoming packet matches at least one classification rule pointing to an SF Map in which the SF Identifier is listed, the SF Node retrieves the next hop SF from the SF Map indicated in the classification rule.

The packet is then handled by the local SF, and the SF Node subsequently forwards the packet to the next hop SF. If not, the packet is forwarded to the next hop according to a typical IP forwarding policy.

Let us consider the example shown in Figure 2. The local SF Node embeds SFa. Once classification rules and the SF Maps are checked, the SF Node concludes SFa must be invoked only when a packet matches Rules 1 and 3. If a packet matches Rule 1, the next SF is SFC. If a packet matches Rule 3, the next SF is SFh.

+-----+ SFC Policy Table +-----+	
Local SF Identifier: SFa +-----+	
Classification Rules Rule 1: If DEST=IP1; then SFC_MAP_INDEX1 Rule 2: If DEST=IP2; then SFC_MAP_INDEX2 Rule 3: IF DEST=IP3; then SFC_MAP_INDEX3 +-----+	
SF Maps SFC_MAP_INDEX1: {SFa, SFc} SFC_MAP_INDEX2: {SFd, SFb} SFC_MAP_INDEX3: {SFa, SFh} +-----+	

Figure 2: SFC Policy Table Example.

10.5. SFs within the Same Subnet

SF Nodes may be enabled in a SFC-enabled domain so that each of them has a direct L3 adjacency with other SF Nodes. In such configuration, no encapsulation scheme is required to exchange traffic between these nodes.

10.6. Service Function Loops

SF Nodes use the SFC Policy Table to detect whether the local SF was already applied to the received packet (i.e., detect SF Loop). The SF Node MUST invoke the local SF only if the packet is received from a SFC Boundary Node or a SF Node having an identifier listed before the local SF in the SF Map matched by the packet. SF Loop detection SHOULD be a configurable feature.

Figure 3 shows an example of a SFC Policy Table of a SF Node embedding SFa. Assume a packet received from Locb that matches Rule 2. SFa must not be invoked because SFb is listed after SFa (see the SF Map list). That packet will be forwarded without invoking SFa.

SFC Policy Table
Local SF Identifier: SFa
SF Maps
SFC_MAP_INDEX1: {SFa, SFc}
SFC_MAP_INDEX2: {SFd, SFa, SFb, SFh}
SFC Locators
Locator_SFb: Locb
Locator_SFC: Locc
Locator_SFd: Locd
Locator_SFh: Loch

Figure 3: Dealing With SF Loops.

10.7. Lightweight SFC Policy Table

If SF loop detection is not activated in an SFC-enabled domain, the PDP may provision SF nodes with a "lightweight" SFC Policy Table. A lightweight SFC Policy Table is a subset of the full SFC Policy Table that includes:

- o Only the SF Maps in which the local SF is involved.
- o Only the next hop SF instead of the full SF chain.

An example of a lightweight SFC Policy Table is shown in Figure 4.

SFC Policy Table
Local SF Identifier: SFa
Lite SF Maps
SFC_MAP_INDEX1, Next_Hop_SF = SFc
SFC_MAP_INDEX2, Next_Hop_SF = SFb
SFC Locators
Locator_SFb: Locb
Locator_SFC: Locc

Figure 4: Lightweight SFC Policy Table.

10.8. Liveness Detection Of SFs By The PDP

The ability of the PDP to check the liveness of each SF invoked in a service chain has several advantages, including:

- o Enhanced status reporting by the PDP (i.e., an operational status for any given service chain derived from liveness state of its SFs).
- o Ability to support various resiliency policies (i.e., bypass SF Node, use alternate SF Node, use alternate chain, drop traffic, etc.) .
- o Ability to support load balancing capabilities to solicit multiple SF instances that provide equivalent functions.

In order to determine the liveness of any particular SF Node, standard protocols such as ICMP or BFD (both single-hop [RFC5881] and multi-hop [RFC5883]) may be utilized between the PDP and the SF Nodes.

Because an SF Node can be responsive from a reachability standpoint (e.g., IP level) while the function it provides may be broken (e.g., a NAT module may be down), additional means to assess whether an SF is up and running are required. These means may be service-specific (e.g., [RFC6849], [I-D.tsou-softwire-bfd-ds-lite]).

For more sophisticated load-balancing support, protocols that allow for both liveness determination and the transfer of application-specific data, such as SNMP and NETCONF may be utilized between the PDP and the SF Nodes.

11. IANA Considerations

This document does not require any IANA actions.

12. Security Considerations

Means to protect SFC Boundary Nodes and SF Nodes against various forms of DDoS attacks MUST be supported. For example, mutual PDP and SF node authentication should be supported. Means to protect SF nodes against malformed, poorly configured (deliberately or not) SFC Policy Tables should be supported.

SFC Boundary Nodes MUST strip any existing SF Map Index when handling an incoming packet. A list of authorized SF Map Indexes are configured in the SFC elements.

NETCONF-related security considerations are discussed in [RFC6146].

Means to prevent SF loops should be supported.

Nodes involved in the same SFC-enabled domain MUST be provisioned with the same SFC Policy Table. Possible table inconsistencies may result in forwarding errors.

13. Contributors

The following individuals contributed to the document:

Parviz Yegani
Juniper Networks
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
USA
Email: pyegani@juniper.net

Paul Quinn
Cisco Systems, Inc.
USA
Email: paulq@cisco.com

Linda Dunbar
Huawei Technologies
1700 Alma Drive, Suite 500
Plano, TX 75075, USA
Phone: (469) 277 5840
Email: ldunbar@huawei.com

14. Acknowledgments

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

15. References

15.1. Normative References

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

15.2. Informative References

- [I-D.boucadair-sfc-design-analysis]
Boucadair, M., Jacquenet, C., Parker, R., and L. Dunbar,
"Service Function Chaining: Design Considerations,
Analysis & Recommendations", draft-boucadair-sfc-design-
analysis-02 (work in progress), February 2014.
- [I-D.boucadair-sfc-requirements]
Boucadair, M., Jacquenet, C., Jiang, Y., Parker, R.,
Pignataro, C., and K. Kengo, "Requirements for Service
Function Chaining", draft-boucadair-sfc-requirements-03
(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-00
(work in progress), January 2014.
- [I-D.sin-sdnrg-sdn-approach]
Boucadair, M. and C. Jacquenet, "Software-Defined
Networking: A Perspective From Within A Service Provider",
draft-sin-sdnrg-sdn-approach-09 (work in progress),
January 2014.
- [I-D.tsou-softwire-bfd-ds-lite]
Tsou, T., Li, B., Zhou, C., Schoenwaelder, J., Penno, R.,
and M. Boucadair, "DS-Lite Failure Detection and
Failover", draft-tsou-softwire-bfd-ds-lite-06 (work in
progress), February 2014.
- [RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black,
"Definition of the Differentiated Services Field (DS
Field) in the IPv4 and IPv6 Headers", RFC 2474, December
1998.
- [RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z.,
and W. Weiss, "An Architecture for Differentiated
Services", RFC 2475, December 1998.
- [RFC2753] Yavatkar, R., Pendarakis, D., and R. Guerin, "A Framework
for Policy-based Admission Control", RFC 2753, January
2000.
- [RFC2983] Black, D., "Differentiated Services and Tunnels", RFC
2983, October 2000.
- [RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network
Address Translator (Traditional NAT)", RFC 3022, January
2001.

- [RFC5881] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, June 2010.
- [RFC5883] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD) for Multihop Paths", RFC 5883, June 2010.
- [RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, November 2010.
- [RFC6092] Woodyatt, J., "Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service", RFC 6092, January 2011.
- [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.
- [RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.
- [RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, June 2011.
- [RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, August 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.
- [RFC6849] Kaplan, H., Hedayat, K., Venna, N., Jones, P., and N. Stratton, "An Extension to the Session Description Protocol (SDP) and Real-time Transport Protocol (RTP) for Media Loopback", RFC 6849, February 2013.

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

Ron Parker
Affirmed Networks
Acton, MA
USA

EMail: Ron_Parker@affirmednetworks.com

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

Phone: +34 913 129 041
EMail: diego@tid.es

Jim Guichard
Cisco Systems, Inc.
USA

EMail: jguichar@cisco.com

Carlos Pignataro
Cisco Systems, Inc.
USA

EMail: cpignata@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

Network working group
Internet Draft
Intended status: Informational
Expires: January 2015

L. Dunbar
Huawei
Ron Parker
Affirmed Networks
I. Smith; S. Majee
F5 Networks
N. So
Vinci Systems
Donald Eastlake
Huawei
July 4, 2014

Architecture for Chaining Legacy Layer 4-7 Service Functions
draft-dunbar-sfc-legacy-l4-l7-chain-architecture-05.txt

Status of this Memo

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

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, except to publish it as an RFC and to translate it into languages other than English.

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 November 4, 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.

Abstract

This draft describes the architecture for chaining existing Layer 4-7 service functions that are not aware of newly defined SFC header. The intent is to identify optimal architecture for flexibly chaining existing Layer 4-7 functions to meet various service needs.

Table of Contents

1. Introduction.....	3
2. Conventions used in this document.....	3
3. Legacy Layer 4-7 Service Functions and Chaining.....	4
3.1. Layer 4-7 Service Functions.....	4
3.2. Metadata to Layer 4-7 Service Functions.....	4
3.2.1. Metadata at different OSI Layers.....	5
3.2.2. Framework of carrying the metadata.....	5
4. Architecture for Chaining Legacy Layer 4-7 Service Functions...	6
4.1. Service Function Forwarder for Layer 4-7 Service Functions	7
4.2. Layer 4-7 nodes connection to SFF Nodes.....	9
4.3. Traffic Steering at SFF Nodes.....	10
5. Control Plane for Layer 4-7 Service Function Chain.....	11
5.1. Multiple Instances of a Service Function.....	11
5.2. Service Chain Re-Classification.....	13
5.3. Layer 4-7 traffic Steering.....	14
6. Service Chain from the Layer 7 Perspective.....	16
7. Conclusion and Recommendation.....	16
8. Manageability Considerations.....	17
9. Security Considerations.....	17
10. IANA Considerations.....	17
11. References.....	17
11.1. Normative References.....	17

11.2. Informative References.....	17
12. Acknowledgments.....	18

1. Introduction

This draft describes the architecture for chaining existing Layer 4-7 service functions that are not aware of newly defined SFC header. The intent is to identify optimal architecture for flexibly chaining existing Layer 4-7 functions to meet various service needs.

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].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying RFC-2119 significance.

Chain Classifier: A component that performs traffic classification and potentially encodes a unique identifier or the SF MAP Index introduced by [SFC-Framework] to the packets. The unique identifier in the packets can be used by other nodes to associate the packets to a specific service chain and/or steer the packets to the designated service functions.

DPI: Deep Packet Inspection

FW: Firewall

Legacy Layer 4-7 Service Function: Same as the Service Functions defined in [SFC-Problem] except that they may not be aware of the new service function chain header encapsulations. Many of existing Layer 4-7 service functions fall into this category. Exemplary functional modules include Firewall, Deep Packet Inspection (DPI), Encryption, Packet De-duplication, Compression, TCP Acceleration, NAT, and etc

Service Function Instance: One instantiation of a service function.

One service function could have multiple identical instances. For a service function with different functional instantiations, e.g. one instantiation applies policy-set-A (NAT44-A) and other applies policy-set-B (NAT44-B), they are considered as two different service functions."

Some Service Function Instances are visible to Service Chain Path. Sometimes a collection of service function instances can appear as one single entity to the Service Chain Path, leaving the instance selection to local nodes.

3. Legacy Layer 4-7 Service Functions and Chaining

Legacy Layer 4-7 service functions are the existing service functions that may not be aware of any new service encapsulation layers being proposed in SFC WG.

3.1. Layer 4-7 Service Functions

A Layer 4-7 service function performs certain action to the packets traversed through. By Layer 4-7, it means that those functions don't participate in network layer routing protocols. The implementation of such service function can be either Proxy based or Packet Based, or a hybrid of both when more than one function is performed to the same packet flow. Multiple service functions can be instantiated on a single service node as defined by [SFC-ARCH], or embedded in a L2/L3 network node.

- o Proxy based service functions: these service functions terminate original packets, may reassemble multiple packets, reopen a new connection, or formulate new packets based on the received packets.
- o Packet based service functions: these service functions maintain original packets, i.e. they don't make changes to packets traversed through except possibly making changes to metadata attached to the packet or the packet's outer header fields.

Some Layer 4-7 service functions might have intelligence to choose the subsequent service functions on a service chain and pass data packets directly to the selected service functions. However, most existing Layer 4-7 service functions don't have this capability.

3.2. Metadata to Layer 4-7 Service Functions

Strictly speaking, everything carrying the information that is not in the payload data is metadata. IETF has standardized many types of

metadata exchanged among L2/L3 nodes, e.g. QoS bits, MPLS labels, etc. Those metadata are out of the scope of SFC.

Metadata in the SFC sense must mean something more specific such as "the information added to the packet to be carried along with the packet for the consumption of the service function nodes along the chain".

This section classifies the metadata that are meaningful to SFC.

3.2.1. Metadata at different OSI Layers

o Application Layer metadata:

Some Layer 4-7 service functions, especially the proxy based service functions, exchange metadata among themselves by changing the payload of the data packets, e.g. attaching a cookie to the payload or initiating a new TCP session.

Those metadata, especially the metadata among L7 Service Functions, are considered as part of payload. Most likely they are proprietary to application layer. Therefore, they should be out of the scope of SFC.

o Layer 4-7 Service Function Layer Metadata

Some service functions exchange information among themselves. Today, most of those metadata exchanges between legacy Layer 4-7 service functions are vendor specific.

o Network Layer metadata

Some Layer 4-7 service functions exchange metadata with L2/L3 nodes to achieve desired network forwarding behavior.

3.2.2. Framework of carrying the metadata

o Message based metadata:

Some service functions receive metadata from external entities (e.g. policy engines, controller, etc). In Mobile environment, some service functions receive metadata from PCRF via Diameter interfaces. Those metadata are normally flow based, e.g. applying

a specific QoS priority for data packets with specific Source/Destination Address(es), TCP port number, etc. Those metadata don't have to be attached to every data packet.

o Data Packet attached Metadata:

Some metadata has to be attached to packets to facilitate proper treatment by service functions.

o Hybrid Method:

Attaching extra metadata to every packet increases the likelihood of packet size exceeding MTU, which lead to packet fragmentation. Therefore, the metadata attached to packets have to be compact.

To reduce the metadata size attached to data packets, it is worth considering combining the "message based metadata" and the "Packet attached Metadata". I.e. attaching compact index to packets that can correlate to complete metadata passed down from separate messages from external systems.

4. Architecture for Chaining Legacy Layer 4-7 Service Functions

Chaining Layer 4-7 Service Functions not only needs the network that steers data flows to their designated service functions, but also needs an Service Chain Controller that can update the steering policies to the relevant forwarding nodes when changes occurs.

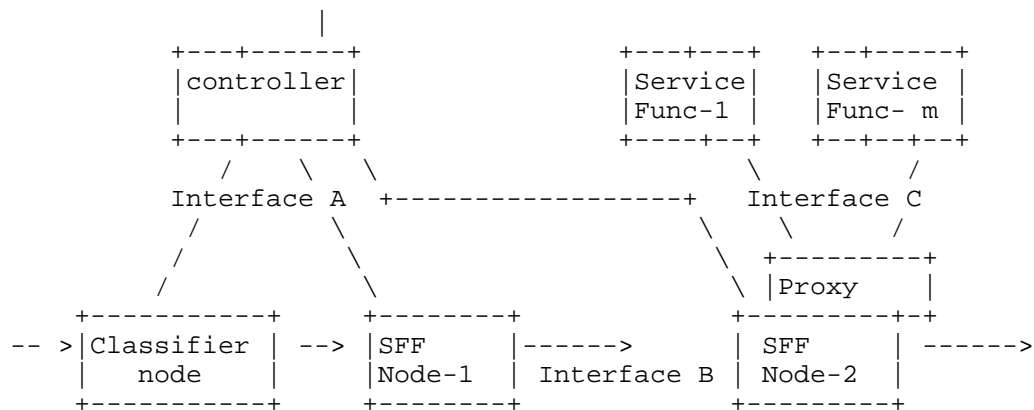


Figure 1 Interfaces needed for Chaining Service Functions

There are 3 types of interfaces to be addressed by the architecture:

- o Interface A: this is the interface between the Service Chain controller and the relevant classifier/steering nodes to exchange the steering policies or/and other information for the service chains.
- o Interface B: this is the network layer that transports the packets among SFF nodes. Proper tunnels might be needed among SFF nodes so that traffic can traverse the legacy network segments.
- o Interface C: this is the interconnection between SFF function and Service Functions. Since some legacy SFs can't recognize the SFC header, a proxy entity is needed to convert the information extracted from SFC header to existing header or tags (e.g. VLANs) recognizable by the SFs for packets traversed on this interface.

4.1. Service Function Forwarder for Layer 4-7 Service Functions

For chaining together legacy Layer 4-7 service functions, the Service Function Forwarder (SFF) defined by [SFC-Arch] may need to terminate the service layer encapsulation on behalf of service functions/nodes that are not aware of the SFC header. There can be multiple SFF nodes in the Service Chain domains [SFC-Framework].

Even though Layer 4-7 Service functions can be instantiated anywhere, it is not uncommon to have multiple service functions instantiated on nodes in close vicinity to a Service Function Forwarder node. The following figure depicts the architecture for chaining those Layer 4-7 service nodes that are not aware of service layer encapsulation. Each SFF is responsible for steering the traffic to their designated local service functions and for forwarding the traffic to the next hop SFF after the local service functions treatment.

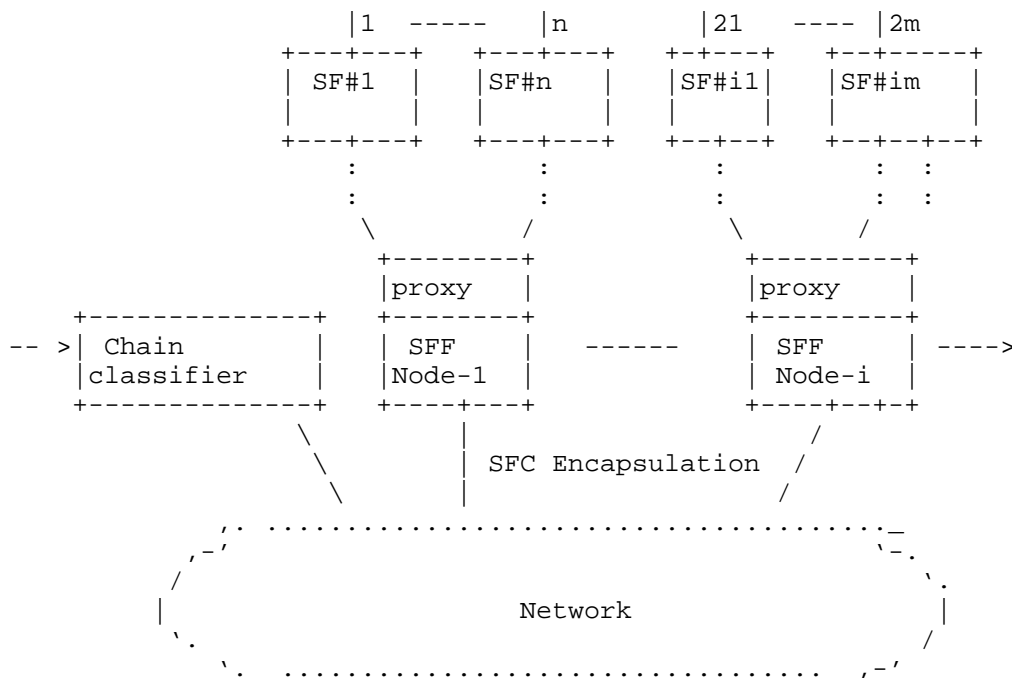


Figure 2 Chaining existing Layer 4-7 service nodes

The "Chain Classifier" node in the figure is to classify the incoming packets/frames into different service flows based on their service characteristics or policies from service chain orchestration or controller. Different service flows can be differentiated by some fields in the packets or can be encapsulated with the corresponding SFC header.

The steering policies for flows arriving at SFF Nodes can be carried by the SFC header in the data packets, separate out-of-band messages from Chain Classifier or external controllers, or combination of both.

The SFF nodes can be standalone devices, or can be embedded within network forwarding nodes. Overlay tunnels are expected to connect the "SFF nodes" together.

4.2. Layer 4-7 nodes connection to SFF Nodes

Since the legacy SFs can't terminate the newly defined SFC header, there has to be a proxy entity either attached to or embedded in a SFF node. Here are the major responsibilities of the proxy entity:

- SFF-> SF direction:

The proxy entity is needed to decapsulate the SFC header from the packets if the SFC header is not recognizable by the SF, extract the service chain identifier from the SFC header, map the service chain identifier to a locally significant tag or header that is recognizable by the legacy SF, and encapsulate the tag or the header to the data packets before sending the packets to the SF.

By locally significant, it means that the tag or the header is only local to the link/path between the SFF Proxy entity and the SF, and is capable of differentiating packets from different service chains that traverse the link/path.

Examples of locally significant tags include VLANs, GRE key, etc. Examples of locally significant header include encapsulating additional IP, MAC, or GRE header, etc.

If there are metadata carried by the SFC header that are needed by the SF, the proxy entity is responsible for extracting the metadata from the SFC header and passing them to the Service Functions via a method that is supported by the Service Function.

- SF -> SFF direction:

The proxy entity is responsible for constructing the SFC header expected by next SFF nodes from the locally significant tag/header when packets come back from the SF, encapsulating the SFC header back to the data packets before passing to the next SFF nodes.

Layer 4-7 Service nodes can be connected to SFF nodes in various ways. The topology could be bump in a wire or one armed topology.

- o A service function can be embedded in a SFF node (i.e. embedded in a router or a switch). In this case, the combined entity forms the SF node described in the [SFC-ARCH].
- o A service node can be one hop away from a SFF node

The one hop between the SFF node and the service node can be a physical link (e.g. Ethernet link). Under this scenario, there would be a Link Header, i.e. an outer MAC header, added to the data packets that meet the steering criteria.

The one hop link can be a transparent link, i.e. no link address is added to the data packets on the link between the SFF node and Service node. I.e. the service nodes can apply treatment to data frames arrived at the ingress port regardless of the Link Destination address.

- o A service node can be multiple hops away, such as when a service function is deployed in an on-net or private *aaS offering. Under this scenario, a tunnel is needed between the service node and the SFF node.

4.3. Traffic Steering at SFF Nodes

The forwarding (or steering) policies for data packets received by the SFF Nodes can be carried by the SFC header in the data packets or combined with separate out-of-band messages from external controller(s) or the Chain Classifier. When using the out-of-band messages to carry the steering policies to SFF nodes, the steering policies have to be correlated with some fields in the data packets. Those fields of the data packets play the role of differentiating packets belong to different service chains.

It worth noting that when one SFF node have multiple Service Functions (SF) attached, there could be two different Chains going through one common SF#1, but the Chain #1 needs to go to SF#4 after SF#1, and the Chain #2 needs to go to another SFF node after the SF#1. The SFF node has to re-classify traffic coming back from a port connected to a SF if the Chain identifier is not carried by the data packets.

The policies to steer traffic to their corresponding service functions or service function instances can change. Those steering policies can be dynamically updated by SFC Header or the out-of-band messages.

There are many types of policies for SFF to steer data packets to their designated service functions, for example:

- o Fixed header based forwarding: traffic steering based on header fields that have fixed position in the data packets:

- o Forwarding based on Layer 2-3 header fields, such as MAC or IP Destination Address, Source Addresses, MPLS label, VLAN ID, or combination of multiple Layer 2-3 header fields.
- o Forwarding based on Layer 4 header (TCP or UDP).
- o QoS header based forwarding.
- o Layer 7 based forwarding: traffic steering (or forwarding) based on the payload (L7) of data packets.

Multiple data packets may carry some meaningful data, like one HTTP message. Under this scenario, multiple data packets have to be examined before meaningful data can be extracted for making Layer 7 based forwarding decision.

- o Metadata based steering: traffic steering (or forwarding) based on the identity of the initiating user, the UE model or type, the site name or FQDN, or network conditions (congestion, utilization, etc.).

However those metadata might not necessarily be carried by each data packet due to extended bits required that can cause high probability of packet fragmentation. Those metadata can be dynamically passed down to steering nodes in some forms of steering policies from network controller(s).

5. Control Plane for Layer 4-7 Service Function Chain

5.1. Multiple Instances of a Service Function

One service function could have multiple identical instances, potentially attached to different SFF nodes. It is also possible to have multiple identical service function instances attached to one Service Function Forwarder (SFF) node, especially in an environment where service function instances are running on virtual machines with each having limited capacity.

At functional level, the order of service functions, e.g. Chain#1 {s1, s4, s6}, Chain#2 {s4, s7}, is important, but very often which instance of the Service Function "s1" is selected for the Chain #1 is not. It is also possible that multiple instances of one service function can be reached by different network nodes. The actual instance selected for a service chain is called "Service Chain Instance Path".

There are various policies that could be employed to select instances for service chain instance path. Some Service Function Instances are visible to Service Chain Path. Sometimes a collection of service function instances can appear as one single entity to the Service Chain Path, leaving the instance selection to local nodes.

When there is change to the instances selected for a Service Chain Instance Path, either in-band or out-of-band messages can be sent to the SFF nodes to update the steering policies dynamically.

The downside with out-of-band messages is synchronization and race conditions. For a newly recognized flow, it is not scalable to expect the classifier node to queue the packets until the out-of-band notification is acknowledged by every Service Function Forwarder node. On the other hand, it is reasonable to use out-of-band messages to inform steering policies on SFF nodes if the steering policies can be pre-established before traffic arrives at the Classifier nodes, e.g. subscriber profile basis service chain instance path.

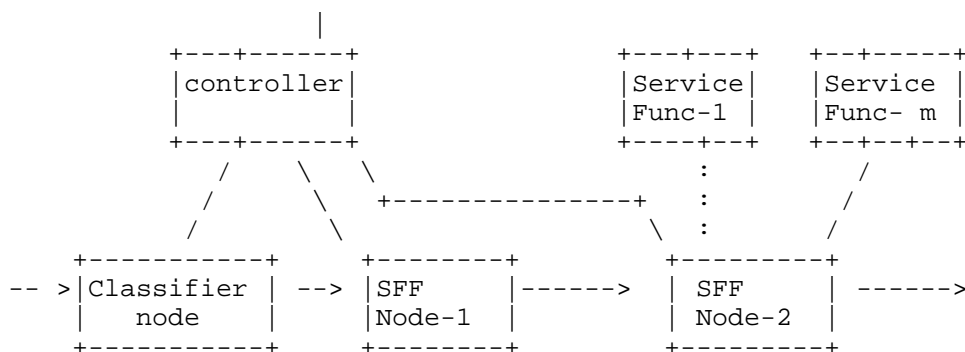


Figure 3 Controller for Service Chain Instance Path

Some service functions make changes to data packets, such as NAT changing the address fields. If any of those fields are used in traffic steering along the service chain, the criteria can be different before and after those the service functions.

5.2. Service Chain Re-Classification

The policy for associating flows with their service chains can be complicated and could be dynamic due to different behavior associated with chains.

For a chain of {FW, Header_enrichment, smart_node, Video_opt, Parental Control}, the video optimizer really needs to work on the response path. It may also use completely different encapsulation e.g. ICAP for example. There could be Smart-Node to further classify a particular part of the flow and bypass something, say the "video_opt". Therefore, the classification done by the service chain classification nodes at the network entrance can't completely dictate the exact sequence of service functions.

Basically, some service functions, especially Layer 7 service functions, can re-classify the service chain. So a chain could be constructed explicitly like below:

```
Classifier -> (SF-A) -> (SF-B) -> (SF-L7 Classifier) -- Chain -X
                                     |
                                     +-- Chain Y
```

Essentially SF-L7 is more like deep classification engine that might analyze individual http transaction and classify them differently. In reality SF-L7 can be a reverse proxy that is then capable of handling individual http transaction and select appropriate chain.

For Chain Re-classification, it is necessary to have message level coordination among those SFs and Service Chain Orchestration or/and Controller entities, as shown in the following figure:

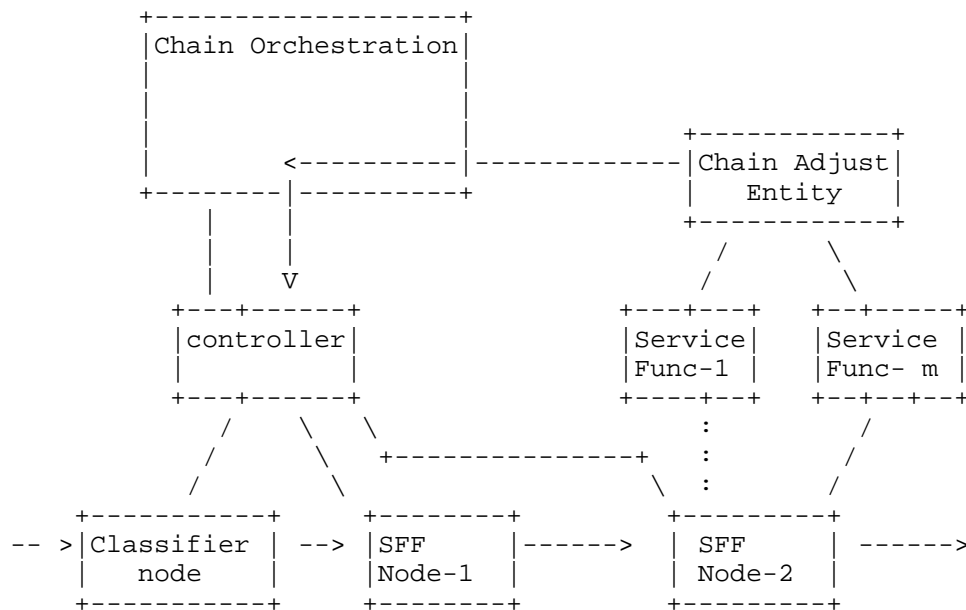


Figure 4 Service Chain Re-classification

The Service Chain Classification node can encounter flows that don't match with any policies. There should be a default policy that applies all statutorily required policies to the unknown flows.

Multiple flows can share one service chain. The criteria to select flows to be associated with their service chain could be different. For example, for one service chain "A" shared by Flow X, Y, Z:

- o Criteria for Flow X to the Service Chain "A" are TCP port
- o Criteria for Flow Y to the Service Chain "A" are Destination Address
- o Criteria for Flow Z to the Service Chain "A" are MPLS label.

5.3. Layer 4-7 traffic Steering

Very often the criteria for steering flows to service functions are based on higher layer headers, such as TCP header, HTTP header, etc.

Most of deployed switches/routers are very efficient in forwarding packets based on Layer 2 or Layer 3 headers, such as MAC/IP destination addresses, or VLAN/MPLS labels but have limited capacity for forwarding data packets based on higher layer header. As of today, differentiating data packets based on higher layer headers depends on ACLs (Access Control List field matching) or DPI, both of which are relatively expensive and extensive use of such facilities may limit the bandwidth of switches/routers.

The Service Chain classification node introduced by [Boucadair-framework] and [SFC-ARCH] can alleviate the workload on large number of nodes in the network, including SFF nodes, from steering traffic based on higher layer fields.

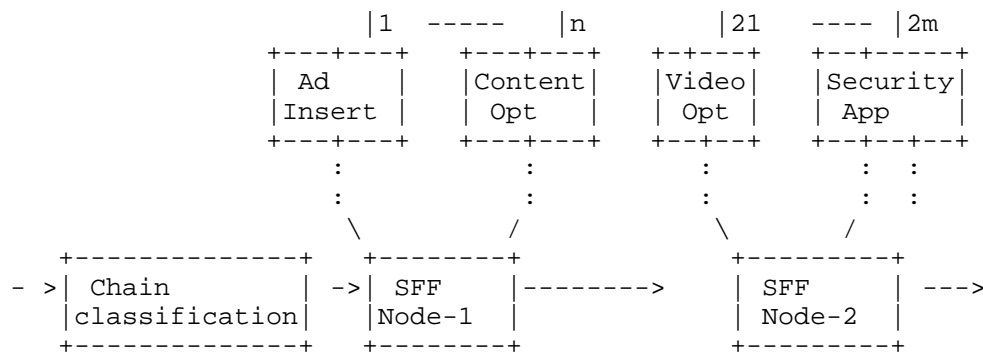


Figure 5 Service Chain Marking At Ingress

A Service Chain Classification node can associate a unique Service Chain Label (e.g. Layer 2 or 3 Label) or SF MAP Index to the packets in the flow. Such a Layer 2 or 3 Label makes it easier for subsequent nodes along the flow path to steer the flow to the service functions specified by the flow's service chain.

The Service Chain Classification Function usually resides on the ingress edge nodes of the service chain domain, such as Wireless Packet Gateway, Broadband Network Gateways, Cell Site Gateways, etc.

In some situations, like service chain for wireless subscribers, many flows (i.e. subscribers) have common service chain requirements. Under those situations, the Service Chain classification Functional can mark multiple flows with the same service chain requirement using the same Layer 2 or 3 Label, which effectively aggregates those flows into one service chain.

For service chains that are shared by a great number of flows, they can be pre-provisioned. For example, if VLAN ID=10 is the service chain that need to traverse "Service-1" at SFF Node #1 and "Service-3" at SFF Node #2, the steering policy for VLAN ID=10 can be dynamically changed by controllers.

6. Service Chain from the Layer 7 Perspective

From the Layer 7 perspective, the service chain can be much more complex. As shown in the figure below, the service functions to be chained can depend on the HTTP message request and reply. The service chain classification nodes may have to examine the whole HTTP message to determine the specific sequence of service functions for the flows. The HTTP message might have to be extracted from multiple data packets. Sometimes, the logic to steer traffic to chain of service functions might depend on the data retrieved from a database based on messages constructed from packets. The decision may depend on the HTTP response rather than the request, or it may depend on a particular sequence of request-response messages. The message handler may also alter the Layer 7 service chain based on hints or modification done by previous service function. HTTP based service function may insert HTTP header to add further criterion for service selection in the next round of classification.

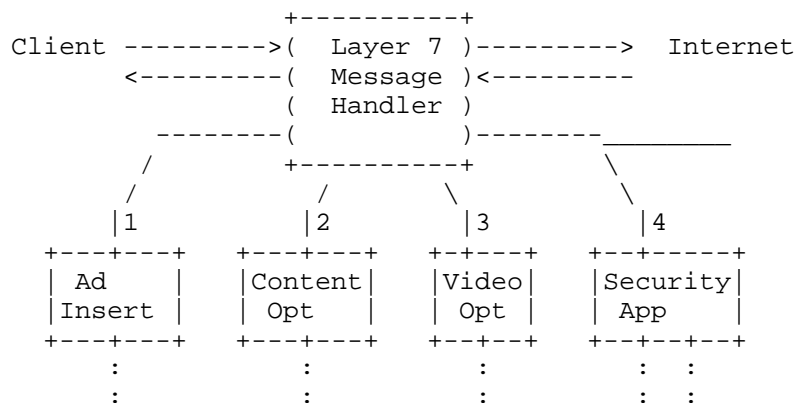


Figure 6 Layer 7 Service Chain Complexity

7. Conclusion and Recommendation

There are many Layer 4-7 service functions being deployed in the network. Many of them are not capable to adapt to new service chain encapsulation layer.

This document provides architecture framework for chaining those Layer 4-7 service functions that are not aware of new service layer encapsulation.

8. Manageability Considerations

There currently exists no single management methodology to control the L2-4 packet-based forwarding device, the L4-7 service delivery device, and the L7+ application server. Such unified management of configuration state is required for service function chaining to be a practical solution.

9. Security Considerations

TBD

10. IANA Considerations

This document requires no IANA actions. RFC Editor: Please remove this section before publication.

11. References

11.1. Normative References

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

11.2. Informative References

[Boucadair-framework] M. Boucadair, et al, "Differentiated Service Function Chaining Framework", <draft-boucadair-service-chaining-framework-00>; Aug 2013

[SFC-Problem] P. Quinn, et al, "Service Function Chaining Problem statement", <draft-quinn-sfc-problem-statement-02>, Dec 9, 2013

[SFC-Framework] M. Boucadair, et al, "Service Function Chaining: Framework & Architecture", <draft-boucadair-sfc-framework-00>; Oct 2013

[SFC-Arch] J. Halpern, et al, "Service Function Chaining (SFC) Architecture", <draft-merged-sfc-architecture-00>, July 2014.

[NSH-Header] P. Quinn, et al, "Network Service Header", < draft-quinn-nsh-01>, July 12, 2013

[SC-MobileNetwork] W. Haeffner, N. Leymann, "Network Based Services in Mobile Network", IETF87 Berlin, July 29 2013

[Application-SDN] J. Giacomoni, "Application Layer SDN", Layer 123 ONF Presentation, Singapore, June 2013

[SC-Use-Case] Liu, et, al., "Service Chaining Use Cases", < draft-liu-service-chaining-use-cases-00>, Sept, 2013

12. Acknowledgments

This draft has merged some sections from
<http://datatracker.ietf.org/doc/draft-parker-sfc-chain-to-path/>.

This draft has taken input from "Application Layer SDN" presentation given by John Giacomoni of F5 at Layer 123 conference. Thanks to Huang Shi Bi and Li Hong Yu for the valuable comments and suggestions.

This document was prepared using 2-Word-v2.0.template.dot.

Authors' Addresses

Linda Dunbar
Huawei Technologies
5340 Legacy Drive, Suite 175
Plano, TX 75024, USA
Phone: (469) 277 5840
Email: ldunbar@huawei.com

Ron Parker
Affirmed Networks
Acton, MA 01720
USA
Email: ron_parker@affirmednetworks.com

Ian Smith
F5 Networks
Email: I.Smith@F5.com

Sumandra Majee
F5 Networks
Email: S.Majee@F5.com

Ning So
Vinci Systems
Email: ning.so@vinci-systems.com

Donald Eastlake
Huawei Technologies
155 Beaver Street
Milford, MA 01757 USA
Phone: 1-508-333-2270
Email: d3e3e3@gmail.com

Internet Working Group

Y. Jiang

Internet Draft

H. Li

Huawei

Intended status: Informational

Expires: August 2014

February 14, 2014

An Architecture of Service Function Chaining
draft-jiang-sfc-arch-01.txt

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 August 14, 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.

Abstract

As network virtualization is opening the gate to much more innovative services for service providers, Service Function Chaining (SFC) provides a flexible way of service provisioning and facilitates their deployments.

This document provides a general abstract architecture for service chaining. It is a flexible and scalable architecture which can fulfill requirements of SFC. Some solutions based on this architecture are also discussed and compared. This architecture can be used as a guideline and also a criterion for the design of SFC.

Table of Contents

1.	Conventions used in this document	3
2.	Terminology	3
3.	Introduction	3
4.	Service Chaining Architecture	5
5.	Service Chaining Topology	7
6.	Service Chaining Construction	7
6.1.	Service chaining Controller	8
7.	Availability and Scalability of Service Chaining.....	8
8.	Service Chaining Solution Considerations	9
8.1.	Ethernet compatible solution	9
8.2.	IP/MPLS compatible solution	10
8.3.	Solution of Network-Located Function Chaining (NLFC) ..	10
8.4.	Solution with a standalone service header	10
9.	Security Considerations	10
10.	IANA Considerations	11
11.	References	11
11.1.	Normative References	11
11.2.	Informative References	11
12.	Acknowledgments	11
Appendix	Change Log	11
Authors' Addresses	12

1. 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 [RFC2119].

2. Terminology

Service Flow: packets/frames with specific service characteristics (e.g., packets matching a specific tuple of fields in Ethernet, IP, TCP, HTTP headers and etc.) or determined by some service policies (such as access port and etc.)

Service Classifier (SCLA), an entity which can classify incoming packets/frames into different service flows based on their service characteristics or some policies.

Service Function (SF): a logical entity which can provide one or more service processing functions for packets/frames such as firewall, DPI (Deep Packet Inspection), LI (Lawful Intercept) and etc. Usually these processing functions are computation intensive. This entity may also provide packet/frame encapsulation/decapsulation capability.

Service Chain (SC): one or more service processing functions in a specific order which are chained together to provide a composite service, and packets/frames from one or more service flow should follow and be processed in sequence.

Service Chaining Domain: a domain where packet is forwarded using service chaining mechanism.

Service Forwarding Entity (SFE): a logical entity which forwards packets/frames to SFs attached to this SFE or other SFEs in the same service chain. Optionally, it provides mapping, insertion and removal of header(s) in packets/frames. Note service forwarding path may not be the shortest path to its destination.

Service Function Chaining (SFC): a mechanism of building service chains and forwarding packets/frames of service flows through them.

3. Introduction

With the maturity of hardware/software in network virtualization, it is possible for service providers to provide more innovative services.

Service chaining provides a flexible way to construct services, e.g., it is easy to insert/remove, and upgrade service processing functions for a service in this framework. It is thus possible to define very complex services over heterogeneous networks in a consistent way with the help of service chaining.

Several drafts have already been proposed for service chaining with different approaches ([NLFC], [NSH] and [L3VPNSC]) and more solutions are expected to emerge in the near future.

This document provides architecture and abstraction of the critical components for service function chaining, and it is hoped to shed some lights on the design of service chaining. Possible solutions of service chaining are also outlined and compared in this document so that they can be gauged under the same criterion.

4. Service Chaining Architecture

In order to decouple the service from the networks, three layers of protocols are envisioned in the stack of SFC:

- Service layer;
- SFC forwarding layer;
- Network under layer.

In the Service Layer, service functions may be provided with metadata capability, but they may also behave like the traditional network devices such as firewall or DPI. Therefore, a null or service layer encapsulation/decapsulation may be needed.

In the SFC forwarding layer, a Service Chain (SC) can be abstracted to consist of the following components:

- At least one Service Classifier (SCLA), which classifies incoming packets/frames into different service flows based on their service characteristics or some policies;
- One or more Service Forwarding Entities (SFEs), which make up the forwarding path of a SC, and each SFE is attached with one or more Service Functions.
- One or more Service Functions (SFs), which are connected with SFEs and provide various service processing functions for a SC.

Service chaining architecture with a single SFE is demonstrated in Figure 1. Upon their entry into a service chaining domain, packets/frames are classified by SCLA into different service flows. Then packets/frames from a specific service flow are forwarded by SFE into one or more SFs in a service chain in correct order. Upon completion of all service processing, they exit the service chaining domain. It should be noted that SCLA, SFE, and even SFs can be implemented in a single network element.

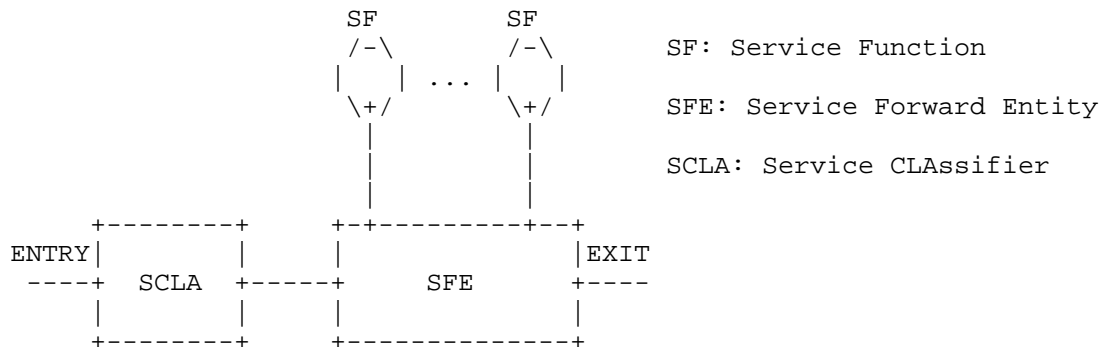


Figure 1 Service Chain with a single SFE

Service chaining architecture with multiple SFEs is further demonstrated in Figure 2.

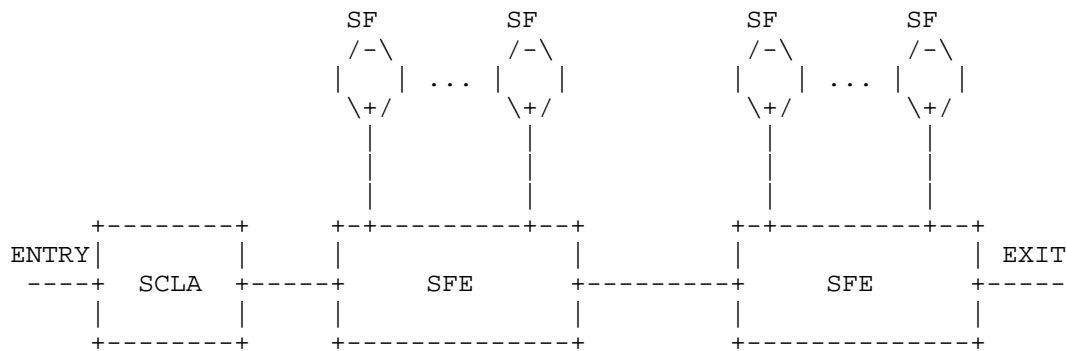


Figure 2 Service Chain with Multiple SFEs

The Network under layer such as IP, MPLS, and Ethernet provides the traditional transport (routing and switching) capability for the SCs. Interconnections between SFs and their SFEs, and between SFE peers can be a physical/logical link or a network path which may be an Ethernet or IP/MPLS underlay network. Ethernet, IP/MPLS or other

tunneling technologies such as those being in progress in NV03 workgroup can be used for this underlay network.

5. Service Chaining Topology

SCLA, SF and SFE can be implemented as a standalone network entity, or in a network entity combined with other functions (such as one SFE plus one or more SFs, one SCLA plus one SFE, or one SCLA plus one SFE and one or more SFs). Furthermore, they may be implemented as software modules running in a DC, in clouds or as standalone physical equipments.

Following service chaining topologies can be supported by this architecture:

-Daisy chain

This is a type of service chain in the shape of a daisy, as demonstrated in Figure 1, where multiple SFs are attached to a single SFE, and SFs cannot send packets/frames directly to each other but via one shared SFE.

-Lily chain

This is a type of service chain in the shape of a lily. It can be regarded as a simplified form of Figure 2, where only a single SF is attached to each SFE, and the SFEs are connected from one to another in sequence.

-Hybrid chain

Part of a hybrid chain may be Daisy chain and other parts may be Lily chain, thus it can be readily combined by the previous two.

More complex service chains can be constructed by reclassification of a SC (for example, by providing multiple SCLAs in a SC); or by change of the metadata in a SF (thus enforce a change to its forwarding paradigm).

6. Service Chaining Construction

Attachment of SFs onto an SFE could be pre-configured on the SFE; or updated by some auto discovery and registering procedure when an SF first attaches to it.

6.1. Service chaining Controller

A service chaining controller which centrally manages service chains (e.g., set up, remove, and monitor service chains) can be implemented together with an SDN controller or a network orchestrator. OSS may also be used as a platform to provide this kind of control function.

In service layer, a Service Graph (SG) shall be defined for each Service Chain (SC), which consists of a list of Service Functions and their order, and parameters for the service flow over the SC.

In service forwarding layer, a Service Forwarding Graph (SFG) shall be defined for each Service Chain, which consists of a list of Service Functions and SFEs between them.

The service chaining controller is responsible for mapping a SG to a SFG, installing/updating/removing classification tables on/from the SCLAs, and installing/updating/removing service forwarding tables on/from the SFEs.

A northbound RESTful API can be specified for the SC controller to manage and control SCs over the web.

7. Availability and Scalability of Service Chaining

Two options for load balancing are possible:

-Load balancing on flows

It is very convenient to provide load balancing and make it scalable in this architecture: construct multiple service chains which provide the same set of service processing functions; the SCLA classifies a service flow into sub-flows, and each sub-flow is directed into a different service chain.

-Load balancing on SFs/paths

It is also possible to provide load balancing in finer granularity. Such as, provide multiple SFs with the same service processing function if a compute bottleneck is expected or found for this service processing function; provide multiple paths between a pair of SFEs if shortage of bandwidth is expected or found for the single path between them. The SFE should be able to load balance a service flow over these SFs or paths with some pre-determined algorithms.

High availability of a service chain will be discussed in a next version.

8. Service Chaining Solution Considerations

A solution must be able to classify packets/frames into different service flows. This is done by the Service Classifier (a packet/frame is classified with a tuple of fields in one or more headers or local policy). After the classification, a Service Identification (SID) may be applied to the packet/frame so that no further classification is needed in subsequent SFEs. SID can be mapped to traditional header fields such as a VLAN or an MPLS label (hence VLAN mapping or MPLS label swap is needed), or carried in a new service header (hence an extra header is inserted).

A solution must be able to forward packets/frames across all service processing entities in a service chain in a correct sequence. This is done by the service forwarding entity with the help of SID.

A solution should support the exchange of information between service processing entity and the service forwarding entity. For example, the processing result of a packet in a service processing entity can be encoded in a service header in the packets. But for long term service states (which can influence the processing of multiple packets/frames), they can be locally stored and/or signaled to service forwarding entity or other service processing entities by some specific control channels (e.g., OpenFlow protocol).

[REQ] outlines a suite of requirements for service function chaining.

Possible solutions for service chaining are outlined in the following sub-sections.

8.1. Ethernet compatible solution

Ethernet technology (IEEE 802.1Q, 802.1ad, and etc.) or DC technology can be used to support service chaining. SCLA maps service ID of a service flow to a VLAN (e.g., C-VLAN and S-VLAN) ID or a VXLAN ID in a frame. SFE then forwards the frame with the VLAN ID to appropriate SFEs for service processing (may need to change VLAN ID for each SF).

The issue with this solution: its service chaining layer is entangled with its transport layer, thus planning and provisioning of service chains are complex, and it may not be a scalable solution to be used

in a large network since VLAN ID or VXLAN ID space is shared by both service ID and tunnel ID.

8.2. IP/MPLS compatible solution

MPLS can be used to support service chaining - only an MPLS label is needed to represent SID of a service flow. Service ID of a service flow is mapped to an MPLS label in the packets by the SCLA. SFE then forwards frames with the label to appropriate SFs for service processing (may need to swap the label for each SF).

[L3VPNSC] demonstrates how service topology specific Route Targets can be introduced into BGP/MPLS VPN to support automatic signaling of service chaining.

Similarly, L2VPN may be crafted to support service chaining.

8.3. Solution of Network-Located Function Chaining (NLFC)

[NLFC] proposes a solution of using NLFC Map (an ordered list of NLF identifiers) for service chaining, where a packet carries a map index of its service chain, then NLF nodes forward the packet by looking up the map index in locally stored NLFC Policy Table for its next NLF node.

8.4. Solution with a standalone service header

A standalone service header can provide an independent layer of protocol data unit, so that service specific information such as SID can be carried over all kinds of underlay networks with no need of mapping.

[NSH] proposes a new Network Service Header for service chaining, which consists of a Base Header and a Context Header.

[SCHM] also introduces a simple service chaining header and how to use it to build service chains.

9. Security Considerations

This document proposes architecture for the service chaining, outlines several types of its topology, and discusses possible approaches in this domain, thus no security issue is raised at present. It was expected that further solutions for these requirements will deal with security considerations specifically.

10. IANA Considerations

No IANA action is needed for this document.

11. References

11.1. Normative References

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

11.2. Informative References

[NLFC] Boucadair, M., and Jacquenet, C., "Service Function Chaining: Framework & Architecture", draft-boucadair-sfc-framework-02, February 2014.

[NSH] Quinn, P., and Fernando, R., and et al, "Network Service Header", draft-quinn-sfc-nsh-01, February 2014.

[SCHM] Niu, L., Li, H., and Jiang, Y., "A Service Function Chaining Header and its Mechanism", draft-niu-sfc-mechanism-00, January 2014.

[L3VPNSC] Fernando, R., Rao, D., and et al, "Virtual Topologies for Service Chaining in BGP IP VPNs", draft-rfernando-l3vpn-service-chaining-01, June 2013.

[REQ] Boucadair, M., and et al, "Requirements for Service Function Chaining", draft-boucadair-sfc-requirements-03, February 2014.

[PS] Quinn, P., and et al, "Network Service Header", draft-ietf-sfc-problem-statement

12. Acknowledgments

TBD.

Appendix Change Log

A.1 draft-jiang-service-chaining-arch-00, June 2013

A.2 draft-jiang-sfc-arch-00, October 2013

A.2 draft-jiang-sfc-arch-01, February 2014

Authors' Addresses

Yuanlong Jiang
Huawei Technologies Co., Ltd.
Bantian, Longgang district
Shenzhen 518129, China
Email: jiangyuanlong@huawei.com

Hongyu Li
Huawei Technologies Co., Ltd.
Bantian, Longgang district
Shenzhen 518129, China
Email: hongyu.lihongyu@huawei.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: March 29, 2014

W. Liu
H. Li
O. Huang
Huawei Technologies
M. Boucadair
France Telecom
N. Leymann
Deutsche Telekom AG
Z. Cao
China Mobile
J. Hu
China Telecom
September 25, 2013

Service Function Chaining Use Cases
draft-liu-service-chaining-use-cases-02

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.

This document presents a set of 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 29, 2014.

Copyright Notice

Copyright (c) 2013 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	4
3. Service Function Chaining Deployment Scenarios	4
3.1. Use Case of Service Function Chain in Broadband Network .	4
3.2. Use Case of Service Function Chain in Mobile Core Network (a.k.a., Gi-LAN)	5
3.3. Use Case for Distributed Service Function Chain	7
3.4. Use Case of Service Function Chain in Data Center	7
4. Use Cases of Service Function Chaining Technical Scenario . .	8
4.1. General Use Case for Service Function Chaining	8
4.2. Use Case for Service Function Chain with NAT Function . .	9
4.3. Use Case for Multiple Underlay Networks	9
4.4. Use Case of Service Path Forking	10
4.5. Use Case of Multiple Service Paths Share one Service Function	11
4.6. Use Case of Service Layer Traffic Optimization	11
5. Security Considerations	12
6. Acknowledgements	12
7. Informative References	12
Authors' Addresses	13

1. Introduction

The delivery of value-added services relies on the invocation of various Service Functions (SFs). Indeed, 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 (e.g., HTTP header enrichment) with the identity of the subscriber or the UE (User Equipment).

This document describes some use cases of Service Function Chaining (SFC). The overall SFC Framework is defined in [I-D.boucadair-service-chaining-framework].

For most of the use cases presented in this document,

- o SFC are not per-subscriber. In other words, this document assumes per-subscriber SFCs are not instantiated in the network. Deployments cases that would require per-subscriber SFCs are out of scope.
- o Instantiated SFC are driven by service creation and offering 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.

Figure 1 illustrates a possible deployment position for Service Function Chaining: between BNG and CR (Core Router). The Service Function Chain shown 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.

3.2. Use Case of Service Function Chain in Mobile Core Network (a.k.a., Gi-IAN)

Gi interface is a reference point between the GGSN (Gateway GPRS Support Node) and an external PDN (Packet Domain Network) [RFC6459]. In 4G networks, this reference point is called SGi. The following notes are made:

- o There is no standard specification of such reference points (i.e., Gi and SGi) in term of Service Functions to be located in that segment.
- o Several (S)Gi Interfaces can be deployed within the same PLMN (Public Land Mobile Network); this depends on the number of PDNs and other factors

Traffic is directed to/from Internet traversing one or more Service Functions. Note, these Service Functions are called "enablers" by some operators.

Plenty of Service Functions are enabled in that segment. Some of these functions are co-located on the same device while other standalone boxes can be deployed to provide some other Service Functions. The number of these SFs is still growing due to the deployment of new value-added services. Some of these functions are not needed to be invoked for all services/UEs, e.g.,:

- o TCP optimization function only for TCP flows
- o HTTP header enrichment only of HTTP traffic
- o Video optimization function for video flows
- o IPv6 firewall + NAT64 function for outgoing IPv6 packets
- o IPv4 firewall + NAT64 function for incoming IPv4 packets
- o Etc.

Figure 2 illustrates a use case of Gi Interface scenario. Figure 2 involves many Service Functions that are enabled in the Gi Interface: WAP GW, TCP Optimizer, Video Optimizer, Content Caching, FW, NAT (44,

64), etc. This list is not exhaustive but it is provided for illustration purposes.

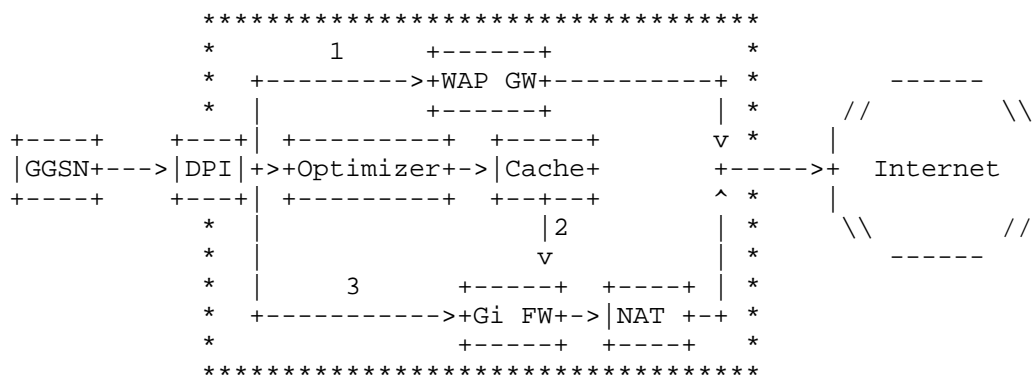


Figure 2: An example of Service Function Chain scenario in the Gi Interface

For example, the traffic from GGSN/PGW to Internet can be categorized and directed into the following Service Function Chains by DPI:

- o Chain 1: WAP GW. DPI performs traffic classification function, recognizes WAP protocol traffic, and directs these traffic to the WAP GW through Service Function Chain 1.
- o Chain 2: Optimizer + cache + Firewall + NAT. DPI recognizes and directs the HTTP traffic to the Optimizer, Cache, Firewall and NAT in order, to perform HTTP video optimization, HTTP content cache, firewall and NAT function, respectively.
- o Chain 3: Firewall +NAT. For other traffic to the Internet, DPI directs these traffic by Service Function Chain 3, the traffic would travel the firewall and NAT in order.

Access to internal services is subject to dedicated policies. For example, a dedicated function to update HTTP flow with a UE identifier may be needed to avoid explicit identification when accessing some service platforms operated by the mobile operator.

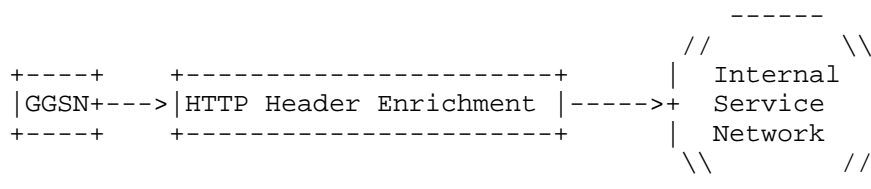


Figure 3: HTTP Header Enrichment

Figure 3 illustrates a use case of HHE (HTTP Header Enrichment). The HHE SF is able to inject the UE identifier to Internal Service Network for identification purpose.

Note, some mobile networks rely on regional-based service platforms (including interconnection links); while some of service functions are serviced in a centralized fashion.

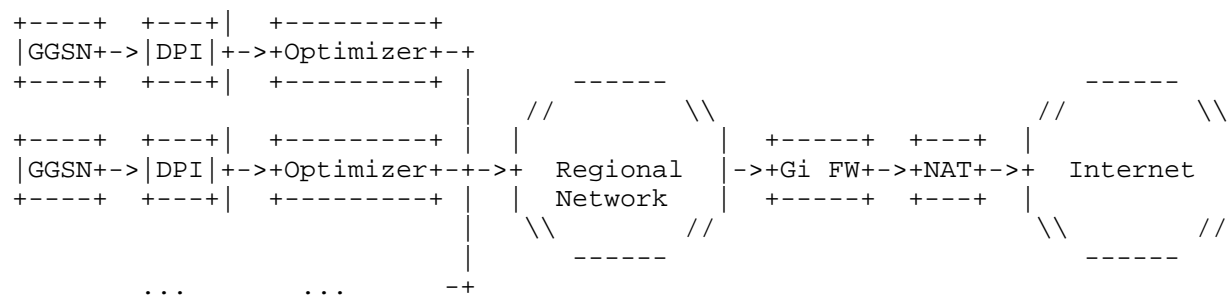


Figure 4: Cross-region services

Figure 4 illustrates a use case of cross-region services, in which the functions that consist of the SFC are located at different regions and flows cross a regional network to go through the Service Function Chains.

3.3. Use Case for 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 an network element close to the customer (e.g. certain security functions).

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

3.4. Use Case of 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 5 illustrates a possible scenario for Service Function Chain in Data Center: SFs are located between the DC Router (access router)

+-----+

Figure 6: 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. Use Case for Service Function Chain with NAT Function

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 their 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.

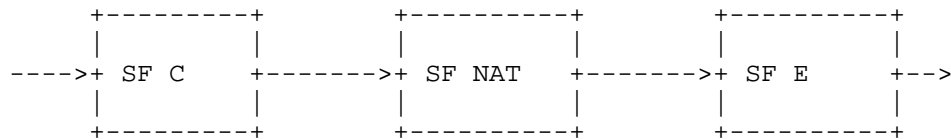


Figure 7: Service Function Chain with NAT64 function

Figure 7 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.

4.3. Use Case for 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.

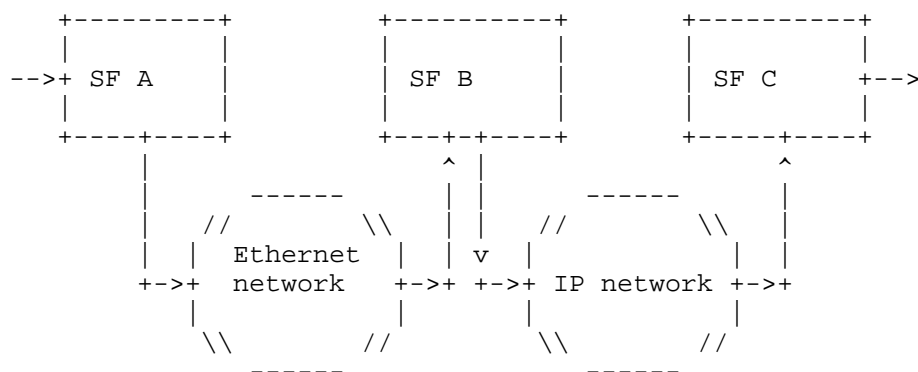


Figure 8: multiple underlay networks: Ethernet and IP

Figure 8 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.

4.4. Use Case of 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.

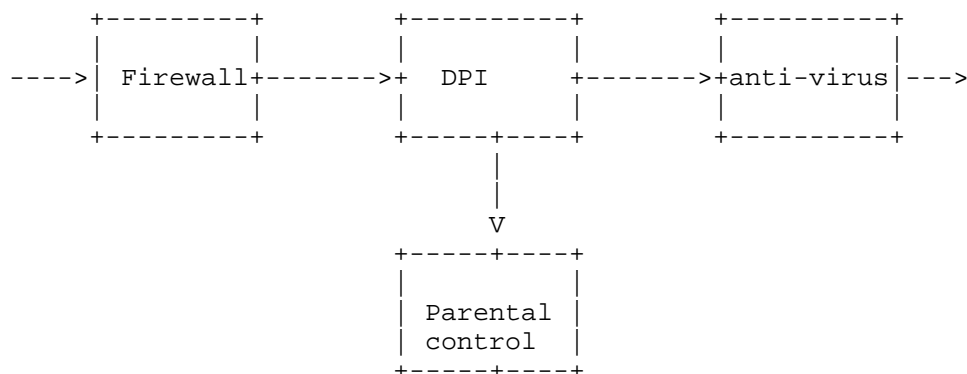




Figure 9: a forking service path

Figure 9 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.

4.5. Use Case of Multiple Service Paths Share one Service Function

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.

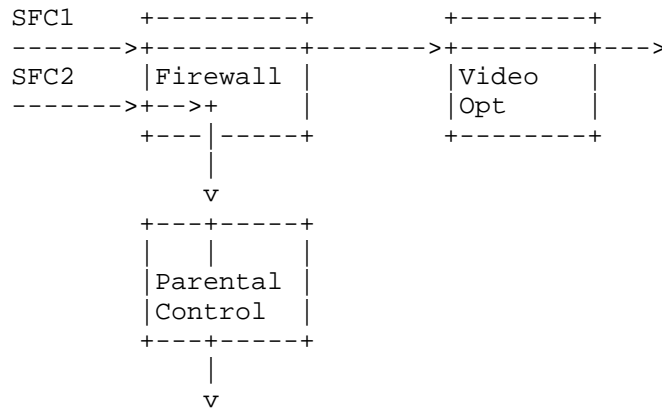


Figure 10: Two Service Function Chains share one Service Function

In Figure 10, 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.

4.6. Use Case of Service Layer Traffic Optimization

In Figure 11, one SF has two instances SF_A1 and SF_A2 on different networking paths. When data traffic hits the first SF_0, it will be forwarded to SF_A1 or SF_A2 depending on the traffic load on different paths. Such service layer traffic optimization is the essential requirement for many computing-intensive service process functions.

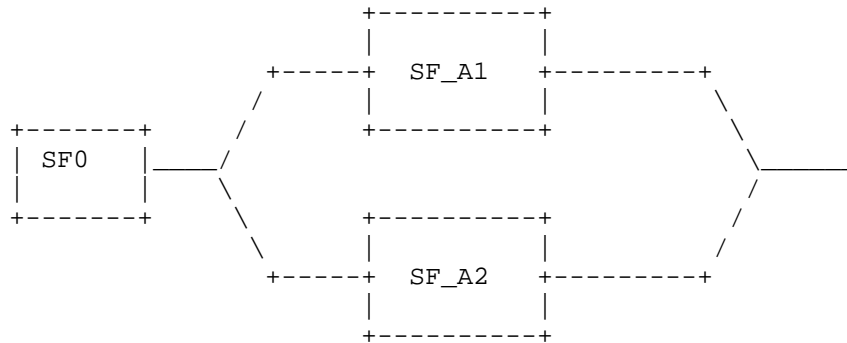


Figure 11: service layer traffic optimization

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.boucadair-service-chaining-framework].

6. Acknowledgements

N/A.

7. Informative References

- [I-D.boucadair-service-chaining-framework]
Boucadair, M., Jacquenet, C., Parker, R., Lopez, D., Guichard, J., and C. Pignataro, "Differentiated Service Function Chaining Framework", draft-boucadair-service-chaining-framework-00 (work in progress), August 2013.
- [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.

Authors' Addresses

Will(Shucheng) Liu
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
France Telecom
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Nicolai Leymann
Deutsche Telekom AG

Email: n.leymann@telekom.de

Zhen Cao
China Mobile

Email: caozhen@chinamobile.com

Jie Hu
China Telecom
No.118 Xizhimennei street, Xicheng District
Beijing 100035
P.R. China

Email: huj@ctbri.com.cn

Network Working Group
Internet-Draft
Intended status: Informational
Expires: February 27, 2014

P. Quinn
J. Guichard
S. Kumar
Cisco Systems, Inc.
P. Agarwal
R. Manur
Broadcom
A. Chauhan
Citrix
N. Leymann
Deutsche Telekom
M. Boucadair
C. Jacquenet
France Telecom
M. Smith
N. Yadav
Insieme Networks
T. Nadeau
K. Gray
Juniper Networks
B. McConnell
Rackspace
K. Glavin
Riverbed
August 26, 2013

Network Service Chaining Problem Statement
draft-quinn-nsc-problem-statement-03.txt

Abstract

This document provides an overview of the issues associated with the deployment of services functions (such as firewalls, load balancers) in large-scale environments. The term service function chaining is used to describe the deployment of such service functions, and the ability of a network operator to specify an ordered list of service functions that should be applied to a deterministic set of traffic flows. Such service function chains require integration of service policy alongside the deployment of applications, while allowing for the optimal utilization of network resources.

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 27, 2014.

Copyright Notice

Copyright (c) 2013 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	4
1.1. Definition of Terms	4
2. Problem Areas	6
3. Service Function Chaining	9
4. Service Function Chaining Use Cases	11
4.1. Enterprise Data Center Service Chaining	11
4.2. Mobility Service Chaining	11
5. Related IETF Work	12
6. Summary	13
7. Security Considerations	14
8. Acknowledgments	15
9. References	16
9.1. Normative References	16
9.2. Informative References	16
Authors' Addresses	17

1. Introduction

Services that are composed of service functions require more flexible service function deployment models than those typically available in networks today. Such services may utilize traditional network service functions (for example firewalls and server load balancers), as well as higher layer applications and features. Services may be delivered within a specific context so that isolated user groups attached to a common network may be formed. Such user groups may require unique capabilities with the ability to tailor service characteristics on a per-tenant/per-subscriber/per-VPN basis that must not affect other user groups

Current service function deployment models are relatively static in that they are bound to fixed network topologies and resources. At present, these deployments are not easily manipulated (i.e.: moved, created or destroyed) even when virtualized elements are deployed. This poses a problem in highly elastic service environments that require relatively rapid creation, destruction or movement of real or virtual service functions or network elements. Additionally, the transition to virtual platforms requires an agile service insertion model that supports elastic and very granular service delivery, and post-facto modification; supports the movement of service functions and application workloads in the existing network, all the while retaining the network and service policies and the ability to easily bind service policy to granular information such as per-subscriber state.

This document outlines the problems encountered with existing service deployment models for service function chaining (often referred to simply as service chaining; in this document the terms will be used interchangeably), as well as the problems of service chain creation/deletion, policy integration with service chains, and policy enforcement within the network infrastructure.

1.1. Definition of Terms

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

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

Service Chain: A service chain defines the required functions and associated order (service-function1 --> service-function 2) that must be applied to packets and/or frames. A service chain does not specify the network location or specific instance of service functions (e.g. firewall1 vs. firewall2).

Service Function: A network or application based packet treatment, application, compute or storage resource, used singularly or in concert with other service functions within a service chain to enable a service offered by a network operator.

A non-exhaustive list of Service Functions includes: firewalls, WAN and application acceleration, Deep Packet Inspection (DPI), server load balancers, NAT44 [RFC3022], NAT64 [RFC6146], HOST_ID injection, HTTP Header Enrichment functions, TCP optimizer, etc.

The generic term "L4-L7 services" is often used to describe many service functions.

Service Node: Physical or virtual element providing one or more service functions.

Network Service: An externally visible service offered by a network operator; a service may consist of a single service function or a composite built from several service functions executed in one or more pre-determined sequences and delivered by one or more service nodes.

2. Problem Areas

The following points describe aspects of existing service deployment that are problematic, and are being addressed by the network service chaining effort.

1. **Topological Dependencies:** Network service deployments are often coupled to the physical network topology creating constraints on service delivery and potentially inhibiting the network operator from optimally utilizing service resources. This limits scale, capacity, and redundancy across network resources.

These topologies serve only to "insert" the service function (i.e. ensure that traffic traverse a service function); they are not required from a native packet delivery perspective. For example, firewalls often require an "in" and "out" layer-2 segment and adding a new firewall requires changing the topology (i.e. adding new L2 segments).

As more service functions are required - often with strict ordering - topology changes are needed before and after each service function resulting in complex network changes and device configuration. In such topologies, all traffic, whether a service function needs to be applied or not, often passes through the same strict order.

A common example is web servers using a server load balancer as the default gateway. When the web service responds to non-load balanced traffic (e.g. administrative or backup operations) all traffic from the server must traverse the load balancer forcing network administrators to create complex routing schemes or create additional interfaces to provide an alternate topology.

2. **Configuration complexity:** A direct consequence of topological dependencies is the complexity of the entire configuration, specifically in deploying service chains. Simple actions such as changing the order of the service functions in a service chain require changes to the topology. Changes to the topology are avoided by the network operator once installed, configured and deployed in production environments fearing misconfiguration and downtime. All of this leads to very static service delivery models. Furthermore, the speed at which these topological changes can be made is not rapid or dynamic enough as it often requires manual intervention, or use of slow provisioning systems.

The service itself can contribute to complexity: it may require an intricate combination of very different capabilities,

regardless of the underlying topology. QoS-based, resilient VPN service offerings are a typical example of such complexity.

3. **Constrained High Availability:** An effect of topological dependency is constrained service function high availability. Worse, when modified, inadvertent non-high availability can result.

Since traffic reaches service functions based on network topology, alternate, or redundant service functions must be placed in the same topology as the primary service.

4. **Consistent Ordering of Service Functions:** Service functions are typically independent; service function_1 (SF1)...service function_n (SFn) are unrelated and there is no notion at the service layer that SF1 occurs before SF2. However, to an administrator many service functions have a strict ordering that must be in place, yet the administrator has no consistent way to impose and verify the ordering of the functions that used to deliver a given service.
5. **Service Chain Construction:** Service 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 will provide not only connectivity state, but will also be increasingly utilized for the formation of services. Such a control / management plane could be centrally controlled and managed, or be distributed between a subset of end-systems.
6. **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. Per-service function packet classification is inefficient and prone to errors, duplicating functionality across service functions. Furthermore packet classification is often too coarse lacking the ability to determine class of traffic with enough detail.
7. **Transport Dependence:** Service functions can and will be deployed in networks with a range of transports, including under and overlays. The coupling of service functions to topology requires service functions to support many transports or for a transport gateway function to be present.

8. Elastic Service Delivery: Given the current state of the art for adding/removing service functions largely centers around VLANs and routing changes, rapid changes to the service layer can be hard to realize due to the risk and complexity of such changes.
9. Traffic Selection Criteria: Traffic selection is coarse, that is, all traffic on a particular segment traverse service functions whether the traffic requires service enforcement or not. This lack of traffic selection is largely due to the topological nature of service deployment since the forwarding topology dictates how (and what) data traverses service function(s). In some deployments, more granular traffic selection is achieved using policy routing or access control filtering. This results in operationally complex configurations and is still relatively inflexible.
10. Limited End-to-End Service Visibility: Troubleshooting service related issues is a complex process that involve network and service expertise. This is especially the case when service chains span multiple DCs, or across administrative boundaries such as externally consumable service chain components. 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.
11. Per-Service (re)Classification: Classification occurs at each service, independent from previously applied service functions. These unrelated classification events consume resources per service. More importantly, the classification functionality often differs per service function and service function cannot leverage the results from other deployed network or service.
12. Symmetric Traffic Flows: Service chains may be unidirectional or bidirectional; unidirectional is one where traffic is passed through a set of service functions in one forwarding direction only. Bidirectional is one where traffic is passed through a set of service functions in both forwarding directions. Existing service deployment models provide a static approach to realizing forward and reverse service chain association most often requiring complex configuration of each network device throughout the forwarding path.
13. Multi-vendor Service Functions: Deploying service functions from multiple vendors often requires per-vendor expertise: insertion models differ, there are limited common attributes and inter-vendor service functions do not share information.

3. Service Function Chaining

Service chaining provides a framework to address the aforementioned problems associated with service deployments:

1. **Service Overlay:** Service chaining utilizes a service specific overlay that creates the service topology: the overlay creates a path between service nodes. The service overlay is independent of the network topology and allows operators to use whatever overlay or underlay they prefer and to locate service functions in the network as needed.

Within the service topology, service functions can be viewed as resources for consumption and an arbitrary topology constructed to connect those resources in a required order. Adding new service functions to the topology is easily accomplished, and no underlying network changes are required. Furthermore, additional service instances, for redundancy or load distribution, can be added or removed to the service topology as required.

Lastly, the service overlay can provide service specific information needed for troubleshooting service-related issues.

2. **Generic Service Control Plane (GSCP):** GSCP provides information about the available service functions on a network. The information provided by the control plane includes service network location (for topology creation), service type (e.g. firewall, load balancer, etc.) and, optionally, administrative information about the service functions such as load, capacity and operating status. GSCP allows for the formulation of service chains and disseminates the service chains to the network.
3. **Service Classification:** Classification is used to select which traffic enters a service overlay. The granularity of the classification varies based on device capabilities, customer requirements, and service functionality. Initial classification is used to start the service chain. Subsequent classification can be used within a given service chain to alter the sequence of service functions applied. Symmetric classification ensures that forward and reverse chains are in place.
4. **Dataplane Metadata:** Dataplane metadata provides the ability to exchange information between the network and service functions, service functions and service functions and service functions and the network. Metadata can include the result of antecedent classification, information from external sources or forwarding related data. For example, service functions utilize metadata, as required, for localized policy decision. A common approach to

service metadata creates a common foundation for interoperability between service functions, regardless of vendor.

4. Service Function Chaining Use Cases

The following sections provide high level overviews of several common service chaining deployments.

4.1. Enterprise Data Center Service Chaining

TBD

4.2. Mobility Service Chaining

TBD

5. Related IETF Work

The following subsections discuss related IETF work and are provided for reference. This section is not exhaustive, rather it provides an overview of the various initiatives and how they relate to network service chaining.

1. L3VPN[L3VPN]: The L3VPN working group is responsible for defining, specifying and extending BGP/MPLS IP VPNs solutions. Although BGP/MPLS IP VPNs can be used as transport for service chaining deployments, the service chaining WG focuses on the service specific protocols, not the general case of VPNs. Furthermore, BGP/MPLS IP VPNs do not address the requirements for service chaining.
2. LISP[LISP]: LISP provides locator and ID separation. LISP can be used as an L3 overlay to transport service chaining data but does not address the specific service chaining problems highlighted in this document.
3. NVO3[NVO3]: The NVO3 working group is chartered with creation of problem statement and requirements documents for multi-tenant network overlays. NVO3 WG does not address service chaining protocols.
4. ALTO[ALTO]: The Application Layer Traffic Optimization Working Group is chartered to provide topological information at a higher abstraction layer, which can be based upon network policy, and with application-relevant service functions located in it. The mechanism for ALTO obtaining the topology can vary and policy can apply to what is provided or abstracted. This work could be leveraged and extended to address the need for services discovery.
5. I2RS[I2RS]: The Interface to the Routing System Working Group is chartered to investigate the rapid programming of a device's routing system, as well as the service of a generalized, multi-layered network topology. This work could be leveraged and extended to address some of the needs for service chaining in the topology and device programming areas.

6. Summary

This document highlights problems associated with network service deployment today and identifies several key areas that will be addressed by the service chaining working group. Furthermore, this document identifies four components that are the basis for service chaining. These components will form the areas of focus for the working group.

7. Security Considerations

Security considerations are not addressed in this problem statement only document. Given the scope of service chaining, and the implications on data and control planes, security considerations are clearly important and will be addressed in the specific protocol and deployment documents created by the service chaining working group.

8. Acknowledgments

The authors would like to thank David Ward, Rex Fernando and Jim French for their contributions.

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

- [ALTO] "Application-Layer Traffic Optimization (alto)",
<<http://datatracker.ietf.org/wg/alto/>>.
- [I2RS] "Interface to the Routing System (i2rs)",
<<http://datatracker.ietf.org/wg/i2rs/>>.
- [L3VPN] "Layer 3 Virtual Private Networks (l3vpn)",
<<http://datatracker.ietf.org/wg/l3vpn/>>.
- [LISP] "Locator/ID Separation Protocol (lisp)",
<<http://datatracker.ietf.org/wg/lisp/>>.
- [NVO3] "Network Virtualization Overlays (nvo3)",
<<http://datatracker.ietf.org/wg/nvo3/>>.
- [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.

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

Puneet Agarwal
Broadcom

Email: pagarwal@broadcom.com

Rajeev Manur
Broadcom

Email: rmanur@broadcom.com

Abhishek Chauhan
Citrix

Email: Abhishek.Chauhan@citrix.com

Nic Leymann
Deutsche Telekom

Email: n.leymann@telekom.de

Mohamed Boucadair
France Telecom

Email: mohamed.boucadair@orange.com

Christian Jacquenet
France Telecom

Email: christian.jacquenet@orange.com

Michael Smith
Insieme Networks

Email: michsmit@insiemenetworks.com

Navindra Yadav
Insieme Networks

Email: nyadav@insiemenetworks.com

Thomas Nadeau
Juniper Networks

Email: tnadeau@juniper.net

Ken Gray
Juniper Networks

Email: kgray@juniper.net

Brad McConnell
Rackspace

Email: bmcconne@rackspace.com

Kevin Glavin
Riverbed

Email: Kevin.Glavin@riverbed.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: November 6, 2014

P. Quinn, Ed.
Cisco Systems, Inc.
J. Halpern, Ed.
Ericsson
May 5, 2014

Service Function Chaining (SFC) Architecture
draft-quinn-sfc-arch-05.txt

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 November 6, 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	3
1.1. Scope	3
1.2. Definition of Terms	3
2. Architectural Concepts	5
2.1. Service Function Chains	5
2.2. Service Function Chain Symmetry	6
2.3. Service Function Paths	6
3. Architecture Principles	7
4. Core SFC Architecture Components	8
4.1. SFC Encapsulation	9
4.2. Service Function (SF)	9
4.3. Service Function Forwarder (SFF)	9
4.3.1. Transport Derived SFF	11
4.4. Network Forwarder (NF)	11
4.5. Classification/Re-classification	11
4.6. SFC Control Plane	12
4.7. Shared Metadata	13
4.8. Resource Control	13
5. The Role of Policy	14
6. Load Balancing Considerations	15
7. SFC Proxy	18
8. MTU Considerations	19
9. SFC OAM	20
10. Summary	21
11. Security Considerations	22
12. Contributors	23
13. Acknowledgments	25
14. IANA Considerations	26
15. References	27
15.1. Normative References	27
15.2. Informative References	27
Appendix A. Existing Service Deployments	28
Appendix B. Issues with Existing Deployments	29
Appendix C. SFC Encapsulation Requirements	30
Authors' Addresses	31

1. Introduction

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

Service Function Chaining 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. Each SF is referenced using an identifier that is unique within an administrative 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.

1.1. Scope

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. 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 function forwarders (SFF) and service functions (SF).

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

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.

A non-exhaustive list of Service Functions includes: firewalls, WAN and application acceleration, Deep Packet Inspection (DPI), server load balancers, NAT44 [RFC3022], NAT64 [RFC6146], HOST_ID injection, HTTP Header Enrichment functions, TCP optimizer, etc.

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 service does not contain the SFC encapsulation.

Service Function Identity (SFID): A unique identifier that represents a service function. SFIDs are unique for each SF within an SFC domain.

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.

Note: The term "service" is overloaded with varying definitions. For example, to some a service is an offering composed of several elements within the operators network whereas for others a service, or more specifically a network service, is a discrete element such as a firewall. Traditionally, these network services host a set of service functions and have a network locator where the service is hosted.

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.

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

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

2. Architectural Concepts

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

2.1. Service Function Chains

In most networks services are constructed as a sequence of SFs that represent an SFC. At a high level, an SFC creates an abstracted view of a service and 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. SFs 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 SF 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. SFCs can have more than one terminus.

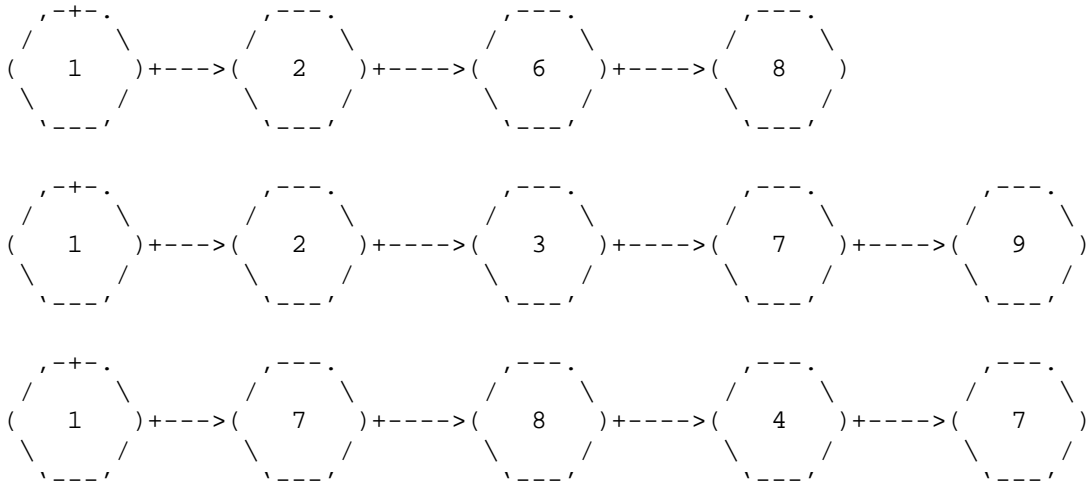


Figure 1: Service Function Chain Graphs

The architecture allows for two or more SFs to be co-resident on the

same service node. In these cases, some implementations 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. Implementation details of such mechanisms are considered out-of-scope for this document.

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). 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.

SFCs may contain cycles; that is traffic may need to traverse more than once one or more SFs within an SFC.

2.3. Service Function Paths

When an SFC is instantiated into the network it is necessary to select the specific instances of SFs that will be used, and to create the service topology for that SFC using SF's network locator. Thus, instantiation of the SFC results in the creation of a Service Function Path (SFP) and is used for forwarding packets through the SFC. In other words, an SFP is the instantiation of the defined SFC.

This abstraction enables the binding of SFCs to specific instances, or set of like instances of SFs based on a range of policy attributes defined by the operator. For example, an SFC definition might specify that one of the SF elements is a firewall. However, on the network, there might exist a number of instances of the same firewall (that is to say they enforce the same policy) and only when the SFP is created is one of those firewall instances selected. The selection can be based on a range of policy attributes, ranging from simple to more elaborate criteria.

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. Consistent policy identifiers: a common identifier is used for SF policy selection.
3. Classification: traffic that satisfies classification rules is forwarded according to a specific SFC. For example, classification can be as simple as an explicit forwarding entry that forwards all traffic from one address into the SFC. Multiple classification points are possible within an SFC (i.e. forming a service graph) thus enabling changes/update to the SFC by SFs.
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, the metadata can be thought of as providing, and sharing the result of classification (that occurs with the SFC 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 in turn imposes that information in the SFC encapsulation for delivery to the requisite SFs. The SFs in turn utilize the user/subscriber information for local policy decisions.

5. Heterogeneous control/policy points: allowing SFs to use independent mechanisms (out of scope for this document) like IF-MAP or Diameter to populate and resolve local policy and (if needed) local classification criteria.

4. Core SFC Architecture Components

The following sub-sections provide details on each logical component that form the basis of the SFC architecture. An overview of how each of these components interact is provided in the following figure.

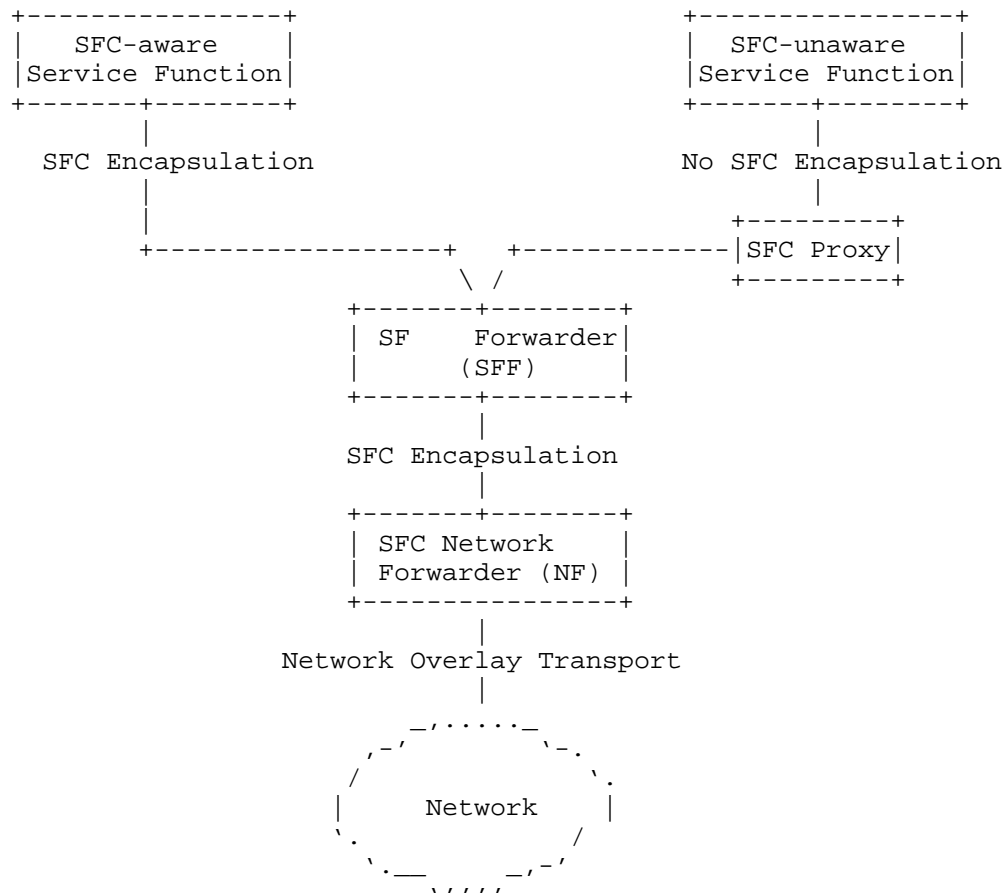


Figure 2: Service Function Chain Architecture Components

4.1. SFC Encapsulation

The SFC encapsulation enables service function path selection and the sharing of metadata/context information.

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. The SFC encapsulation therefore, relies on an outer network transport. Transit nodes -- 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 and as such any network transport protocol may be used to carry the SFC encapsulation.

4.2. Service Function (SF)

The concept of a SF evolves; rather than being viewed as a bump in the wire, a SF becomes a resource within a specified administrative domain that is available for consumption as part of a composite service. As such, SFs have one or more network locators through which they are reachable, and a variable set of attributes that describe the function offered. The combination of network locator and attributes are used to construct an SFP. SFs send/receive SFC encapsulated data from one or more SFFs.

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 a logical SFC proxy function is defined. The SFC proxy acts a gateway between the SFC encapsulation and SFC unaware SFs. The integration of SFC-unaware service function is discussed in more detail in the SFC proxy section.

4.3. Service Function Forwarder (SFF)

The SFF is responsible for forwarding packets and/or frames received from an NF to one or more SFs associated with a given SFF using information conveyed in the SFC encapsulation.

The collection of SFFs creates a service plane using an 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 SF of a given SFC should be applied as traffic flows through the associated SFP. Each SFF need only maintain SFC forwarding information that is relevant locally. The SFC forwarding state at all SFFs collectively represents the SFPs associated with each SFC in the SFC domain.

SFP	Ordered Service Functions				
ID	order1	order2	order3	...	
SFP1	SFID1	SFID5	SFID20		
SFP4	SFID100	SFID3	SFID4	SFID9	
...					

Figure 3: SFF Table

Figure 3 depicts a view of the service forwarding state for two SFPs - SFP1 and SFP4. The SF columns of this table may come from different SFFs.

The SFF component has the following primary responsibilities:

1. SFP forwarding : Traffic arrives at an SFF from one or more NFs. 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 hop in the SFP, the SFF, encapsulates the traffic in the appropriate network transport and delivers it to the NF for delivery to the next SFF along the path.
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 servicing it, SFF fails to find the next SF or knows from the service forwarding state that the SFC is complete. SFF removes the SFC encapsulation and delivers the packet to an NF for forwarding.

3. Maintaining flow state: In some cases, the SFF may be stateful. It creates flows and stores flow-centric information. When traffic arrives after being steered through an SFC-unaware SF, the SFF must perform re-classification of traffic to determine the SFP. A state-full SFF simplifies such classification to a flow lookup.

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. Existing implementations may not be able to act on the SFC encapsulation. These platforms MAY opt to use a transport mechanism which carries the service path information from the SFC encapsulation, and information derived from the SFC encapsulation, to build transport 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. Network Forwarder (NF)

This component is responsible for performing the overlay encapsulation/de-capsulation and forwarding of packets on the overlay network. NF forwarding may consult the SFC encapsulation or the inner payload of an incoming packet only in the necessary cases to achieve optimal forwarding in the network.

4.5. Classification/Re-classification

Traffic that satisfies classification criteria is directed into an SFP and forwarded to the requisite service function(s). Classification is handled by a logical service classification function, and initial classification occurs at the edge of 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 directed into SFP A, or quite granular: all packets matching this 5-tuple are subject to SFP B.

As a consequence of the classification decision, the appropriate SFC encapsulation is imposed on the data prior to forwarding along the SFP.

The SFC architecture supports reclassification (or non-initial

classification) as well. As packets traverse an SFP, reclassification 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.

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 reclassifies 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. In this simple example, the DPI service function reclassified the traffic based on local application layer classification capabilities (that were not available during the initial classification step).

4.6. SFC Control Plane

The SFC control plane is responsible for constructing the SFPs; translating the SFCs to the forwarding paths and propagating path information to participating nodes - network and service - to achieve requisite forwarding behavior to construct the service overlay. For instance, a SFC construction may be static - using specific SF instances, or dynamic - choosing service explicit SF instances at the time of delivering traffic to the SF. In SFC, SFs are resources; the control plane advertises their capabilities, availability and location. 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 administrative domain wide view of all available service function resources as well as the network locator through which they are reachable.
2. Uses SFC policy to construct service function chains, and associated service function paths.
3. Selection of specific SF instances for a requested SFC, either statically (using specific SF instances) or dynamically (using service explicit SF instances at the time of delivering traffic to the SF).
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 metadata syntax to relevant SF instances and/or SFC encapsulation-proxies or their respective policy planes.

4.7. 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. This component is optional. 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.
- o Context information can be derived in several ways:
 - * External sources
 - * Network node classification
 - * Service function classification

4.8. 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 the network path(s) between service nodes, characteristics of the nodes themselves such as memory, number of virtual interfaces, routes, etc..., and configuration of the SFs running on the service nodes.

5. The Role of Policy

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

Specifically, it is assumed that 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 which direct the control logic to change the service chain placement or the 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. Depending upon operator preferences, these policies may also determine which sequences of functions are valid and 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.

6. Load Balancing Considerations

Supporting function elasticity and high-availability shouldn't overly complicate SFC or lead to unnecessary scalability problems.

In the simplest case, where there is only a single function in the chain (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 case where the classifier is separate from the single function or a function at the terminal address may need sub-prefix or per subscriber metadata, we would have a single chain (the metadata changes but the SFC chain does not), regardless of the number of potential terminal addresses for the flow. This is the case of the simple load balancer.

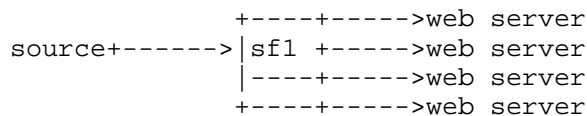


Figure 4: Simple Load Balancing

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 the following figure, we have a chain of five service functions between the traffic source and its destination.

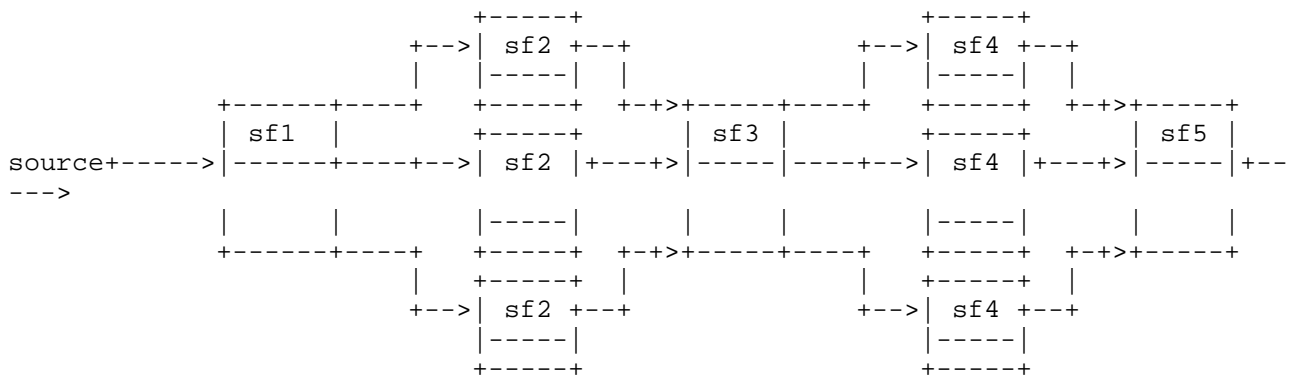


Figure 5: Load Balancing

Either through an imbedded action in sf1 and sf3, or through external control, the service functions sf2 and sf4 are elastically expanded and contracted dynamically. This would be represented as one chain: s1->s2->s3->s4->s5, but with multiple paths (not as a number of chains equal to the factorial combination of potential end-to-end paths). The load distribution decision will be localized (in general, although there might be macro policy controlling that - which is out of scope for the sake of a simple example). In this case, the control entity will push to the sf1 nodes, a table of sorts: sf2 with a series of next hops, and if needed some weighted or other metrics (these could also be decided locally by some policy, but sf1 would need to be aware of expand/contract triggers and actions). sf1 would use local logic -- hash, state table, etc. -- to distribute the chained packets to sf2.

The addition of high availability should likewise not require a multitude of new chains.

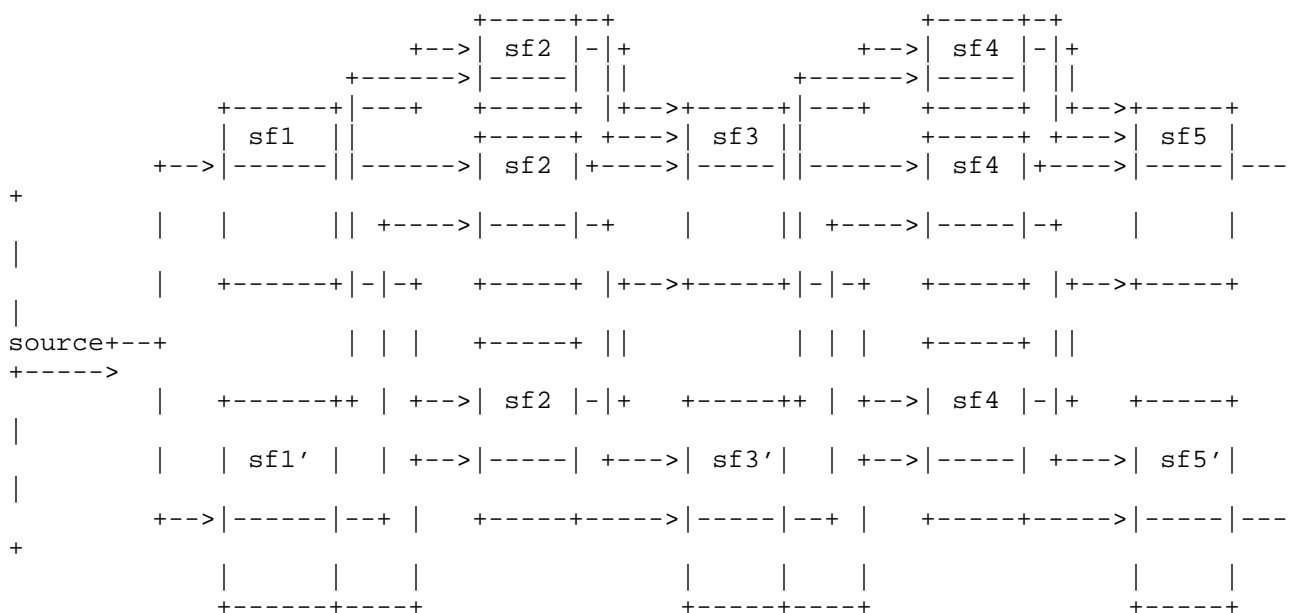


Figure 6: Load Balancing and HA

In the figure, sf1, sf3 and sf5 have a redundant counterpart for high availability purposes (typical of stateful appliance/function redundancy strategies, these entities may have private connections for transferring state not shown). Note that the elasticity of sf2 and sf4 provide a separate high availability strategy for those functions. In the case where sf1', sf3' and sf5' provide transparent dynamic replacement (they assert the addressing characteristics of their counterparts via an internal or external trigger), there is still a single chain (again, not a factorial explosion).

7. SFC Proxy

In order for the SFC architecture to support SFC-unaware SF's, an optional, logical SFC proxy function may be used. This proxy removes the SFC encapsulation and then uses a local attachment circuit to deliver packets to SFC unaware SFs. More specifically:

For traffic received from a NF or SFF, destined to an SF, the SFC proxy:

- o Removes the SFC encapsulation from SFC encapsulated packets and/or frames.
- o Identifies the required SF to be applied based on information carried in the SFC encapsulation.
- o Selects the appropriate outbound local attachment circuit through which the next SF for this SFP is reachable. This information is derived from the SFC encapsulation or from local configuration. Examples of a local attachment circuit include, but are not limited to, VLANs, IP-in-IP, GRE, VXLAN.
- o Forwards the original payload via a 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 Imposes the appropriate SFC encapsulation based on the identification of the SFC to be applied.

8. MTU Considerations

Modern systems are expected to be able to cope gracefully with MTU issues that may arise from the application of additional headers to a packet. Adopting the recommendations of other WG's who have recently tackled this issue (e.g. [RFC6830]), there are several mechanisms for dealing with packets that are too large to transit the path from the point of service classification to the last function (SFN) in the SFC.

In the "stateful" approach, the classifier keeps a per-path record of the maximum size allowed, and sends an ICMP Too Big message to the original source when a packet which is too large is seen (where "too large" implies after the imposition of the appropriate SFC encapsulation).

In the "stateless" approach, for IPv4, packets without the 'DF' bit set, too-large packets are fragmented, and then the fragments are forwarded; all other packets are discarded and an ICMP Too Big message returned.

A recommendation of a specific mechanism and/or its implementation is beyond the scope of this document.

9. 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. 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 run in-band fate-sharing with 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.
- o Out-of-band: reporting beyond the actual dataplane. An additional layer beyond the data-plane OAM, allows for additional alerting and measurements.

Some of the detailed functions performed by SFC OAM include fault detection, continuity checks, connectivity verification, service path tracing, diagnostic and fault isolation, alarm reporting, performance measurement, locking and testing of service functions, and also allow for vendor-specific as well as experimental functions. SFC should leverage, and if needed extend relevant existing OAM mechanisms.

10. Summary

Service function chains enable composite services that are constructed from one or more service functions. This document provides a standard architecture, including architectural concepts, principles, and components, for the creation of Service function chains.

11. Security Considerations

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

12. Contributors

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

Puneet Agarwal
Broadcom
Email: pagarwal@broadcom.com

Andre Beliveau
Ericsson
Email: andre.beliveau@ericsson.com

Kevin Glavin
Riverbed
Email: Kevin.Glavin@riverbed.com

Ken Gray
Cisco Systems, Inc.
Email: kegray@cisco.com

Jim Guichard
Cisco Systems, Inc.
Email: jguichar@cisco.com

Surendra Kumar
Cisco Systems, Inc.
Email: smkumar@cisco.com

Darrel Lewis
Cisco Systems, Inc.
Email: darlewis@cisco.com

Nic Leymann
Deutsche Telekom
Email: n.leymann@telekom.de

Rajeev Manur
Broadcom
Email: rmanur@broadcom.com

Thomas Nadeau
Brocade
Email: tnadeau@lucidvision.com

Carlos Pignataro
Cisco Systems, Inc.

Email: cpignata@cisco.com

Michael Smith
Cisco Systems, Inc.
Email: michsmit@cisco.com

Navindra Yadav
Cisco Systems, Inc.
Email: nyadav@cisco.com

13. Acknowledgments

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.

14. IANA Considerations

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

15. References

15.1. Normative References

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

15.2. Informative References

- [NSCprob] "Network Service Chaining Problem Statement", <<http://datatracker.ietf.org/doc/draft-ietf-sfc-problem-statement/>>.
- [RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.
- [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.
- [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.
- [RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, January 2013.

Appendix A. Existing Service Deployments

Existing service insertion and deployment techniques fail to address new challenging requirements raised by modern network architectures and evolving technologies such as multi-tenancy, virtualization, elasticity, and orchestration. Networks, servers, storage technologies, and applications, have all undergone significant change in recent years: virtualization, network overlays, and orchestration have increasingly become adopted techniques. All of these have profound effects on network and services design.

As network service functions evolve, operators are faced with an array of form factors - virtual and physical - as well as with a range of insertion methods that often vary by vendor and type of service.

Such existing services are deployed using a range of techniques, most often associated with topology or forwarding modifications. For example, firewalls often rely on layer-2 network changes for deployment: a VLAN is created for the "inside" interface, and another for the "outside" interface. In other words, a new L2 segment was created simply to add a service function. In the case of server load balancers, policy routing is often used to ensure traffic from server's returns to the load balancer. As with the firewall example, the policy routing serves only to ensure that the network traffic ultimately flows to the service function(s).

The network-centric information (e.g. VLAN) is not limited to insertion; this information is often used as a policy identifier on the service itself. So, on a firewall, the layer-2 segment identifies the local policy to be selected. If more granular policy discrimination is required, more network identifiers must be created either per-hop, or communicated consistently to all services.

Appendix B. Issues with Existing Deployments

Due to the tight coupling of network and service function resources in existing networks, adding or removing service functions is a complex task that is fraught with risk and is tied to operationalizing topological changes leading to massively static configuration procedures for network service delivery or update purposes. The inflexibility of such deployments limits (and in many cases precludes) dynamic service scaling (both horizontal and vertical) and requires hop-by-hop configuration to ensure that the correct service functions, and sequence of service functions are traversed.

A non-exhaustive list of existing service deployment and insertion techniques as well as the issues associated with each may be found in [NSCprob].

Appendix C. SFC Encapsulation Requirements

TBD

Authors' Addresses

Paul Quinn (editor)
Cisco Systems, Inc.

Email: paulq@cisco.com

Joel Halpern (editor)
Ericsson

Email: jmh@joelhalpern.com

