Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 4, 2015

J. Halpern, Ed.
Ericsson
C. Pignataro, Ed.
Cisco
July 3, 2014

Service Function Chaining (SFC) Architecture
draft-merged-sfc-architecture-00

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 January 4, 2015.

Table of Contents

1. Introduction

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

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

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

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

1.1. 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.

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

1.2. Assumptions

The following assumptions are made:

- Not all SFs can be characterized with a standard definition in terms of technical description, detailed specification, configuration, etc.

- 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.

- 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.

- 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).

- SF chaining logic and related policies should not be exposed outside a given administrative domain.

- Several SF chaining logics can be simultaneously enforced within an administrative domain to meet various business requirements.

- No assumption is made on how FIBs and RIBs of involved nodes are populated.

- How to bind the traffic to a given SF chaining is policy-based.

1.3. Definition of Terms

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

Note: 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.

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

Classifier: An element that performs Classification.

Service Function (SF): A function that is responsible for specific treatment of received packets. A Service Function can act at 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.

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

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 Node (SN): Element within an SFC-enabled domain that hosts one or more service functions and has one or more network locators associated with it for reachability and service delivery.

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

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

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

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

2. Architectural Concepts

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

Service Function Chaining enables the creation of composite 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.

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.

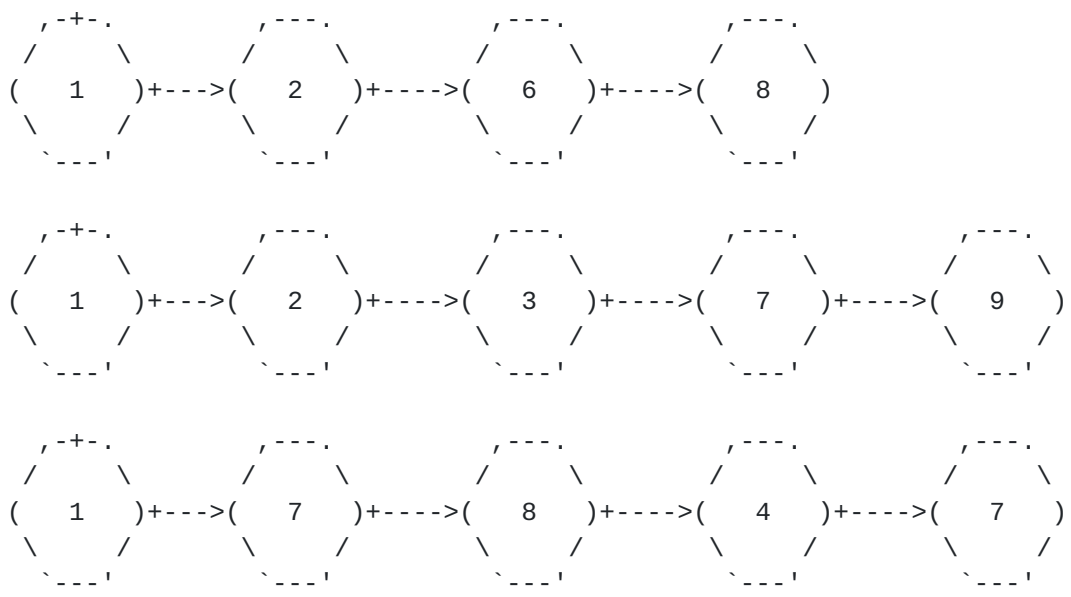

Figure 1: Service Function Chain Graphs

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. Solutions will need to ensure suitable disambiguation for such situations.

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. Plane separation: dynamic provisioning of SFPs is separated from packet handling operations (e.g., packet forwarding).

3. Classification: traffic that satisfies classification rules is forwarded according to a specific 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. Service definition independence: the technical characterization of each SF is not required to design the SFC architecture and SFC

data plane operations. Consequently, no IANA registry is required to store the list of SFs.

6. Service function chain independence: The creation, modification, or deletion of a service chain have no impact on other service chains.

7. 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

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

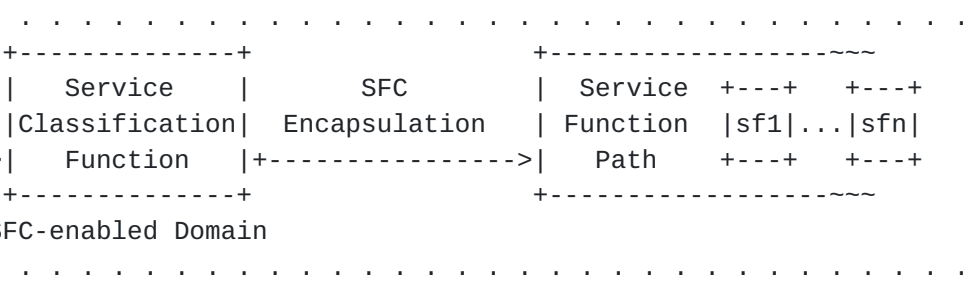

Figure 2: Service Function Chain Architecture

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

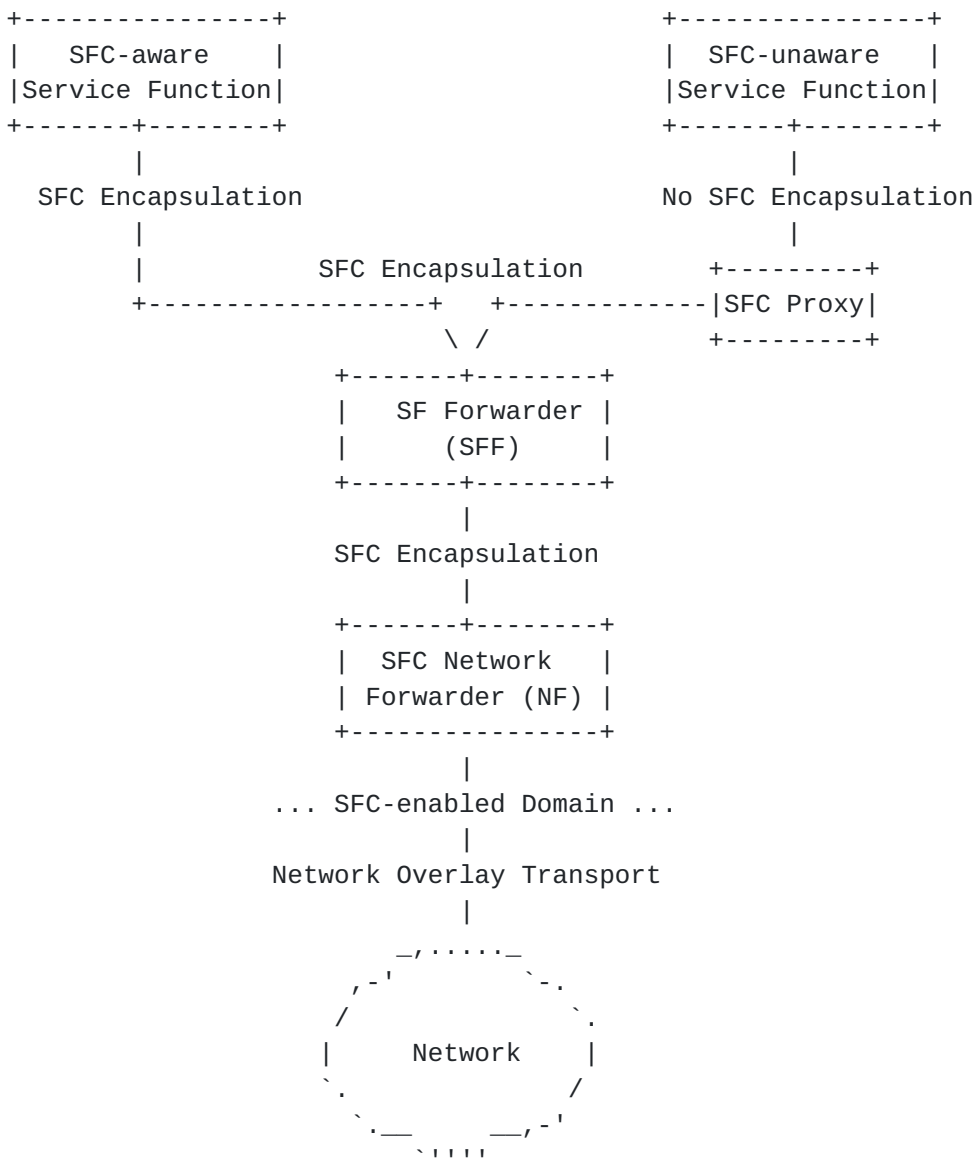

Figure 3: Service Function Chain Architecture Components

4.1. SFC Encapsulation

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

While the SFC architecture defines a new encapsulation - the SFC encapsulation - and several logical components for the construction of SFCs, existing SF implementations may not have the capabilities to act upon or fully integrate with the new SFC encapsulation. In order to provide a mechanism for such SFs to participate in the architecture 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.

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. SFC Proxy

In order for the SFC architecture to support SFC-unaware SF's (e.g., legacy service nodes), 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.

Architecturally, the SFC Proxy along with an SFC-unaware Service Function make up an SF. More specifically:

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

- Removes the SFC encapsulation from SFC encapsulated packets and/or frames.
- Identifies the required SF to be applied based on information carried in the SFC encapsulation.
- 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, L2TPv3, GRE, VXLAN.
- Forwards the original payload via a local attachment circuit to the appropriate SF.

When traffic is returned from the SF:

- 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.
- Imposes the appropriate SFC encapsulation based on the identification of the SFC to be applied.

Alternatively, a service provider may decide to exclude legacy nodes from an SDC domain.

4.6. 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.

4.7. Re-Classification and Branching

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. This is referred to as "branching".

For example, an initial classification results in the selection of SFP A: DPI_1 --> SLB_8. However, when the DPI service function is executed "attack" traffic is detected at the application layer. DPI_1 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).

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

The SFC control plane provides the following functionality:

1. An 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.

4.9. Shared Metadata

Sharing metadata allows the network to provide network-derived information to the SFs, SF-to-SF information exchange and the sharing of service-derived information to the network. 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:

- Allows elements that typically operate as ships-in-the-night to exchange information.
- Encodes information about the network and/or data for post-service forwarding.
- Creates an identifier used for policy binding by SFs.
- Context information can be derived in several ways:
  - External sources
  - Network node classification
  - Service function classification

4.10. 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. Additional Architectural Concepts

6.1. Loop Prevention

This SFC architecture is predicated on topological independence from the underlying forwarding topology. Consequently, a service topology is created by Service Function Paths. Within this service topology, this methods need to support intentioanl, limited loops as described above while detecting, and either resolving or preferably preventing indefinite loops.

6.2. 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. See Figure 4.

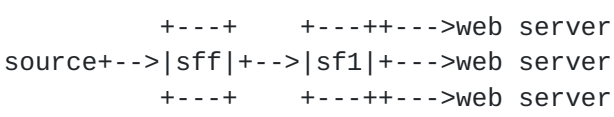

Figure 4: Simple Load Balancing

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

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

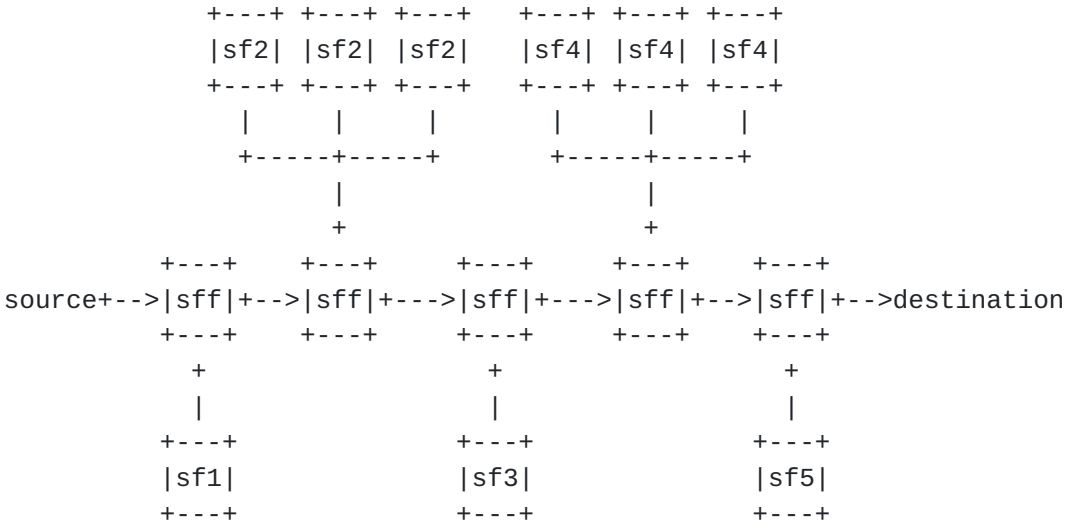

Figure 5: Load Balancing

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

6.3. MTU and Fragmentation 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 SFP.

The solution to these issues should result not adversely affect service nodes. A recommendation of a specific mechanism and/or its implementation is beyond the scope of this document.

6.4. SFC OAM

Operations, Administration, and Maintenance (OAM) tools are an integral part of the architecture. These serve various purposes, including fault detection and isolation, and performance management. For example, there are many advantages of SFP liveness detection, including status reporting, support for resiliency operations and policies, and an enhanced ability to load balance.

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:

- 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.

- 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.

6.5. Operational (and Manageability) Considerations

To be provided.

7. Security Considerations

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

Security considerations apply to the realization of this architecture. Such realization ought to provide means to protect the SFC-enabled domain and its borders against various forms of attacks, including DDoS attacks. Additionally, Service Nodes need to provide means of security against malformed, poorly configured (deliberate or not) protocol constructs and loops.

8. Contributors and Acknowledgments

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

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

Authors:

Mohamed Boucadair

Christian Jacquenet

Ron Parker

Diego R. Lopez

Jim Guichard

Carlos Pignataro

Contributors:

Parviz Yegani

Paul Quinn

Linda Dunbar

Acknowledgements:

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

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

Authors:

Paul Quinn (editor)

Joel Halpern (editor)

Contributors:

Puneet Agarwal

Andre Beliveau

Kevin Glavin

Ken Gray

Jim Guichard

Surendra Kumar

Darrel Lewis

Nic Leymann

Rajeev Manur

Thomas Nadeau

Carlos Pignataro

Michael Smith

Navindra Yadav

Acknowledgements:

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

9. IANA Considerations

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

10. References

10.1. Normative References

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

10.2. Informative References

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

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

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

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

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

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

[RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, January 2013.

Authors' Addresses

Joel Halpern (editor)
Ericsson

Email: jmh@joelhalpern.com

Carlos Pignataro (editor)
Cisco Systems, Inc.

Email: cpignata@cisco.com