Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 7, 2014

P. Quinn
J. Guichard
S. Kumar
C. Pignataro
Cisco Systems, Inc.
N. Leymann
Deutsche Telekom
M. Smith
N. Yadav
Insieme Networks
K. Gray
T. Nadeau
Juniper Networks

October 4, 2013

Service Function Chaining (SFC) Architecture
draft-quinn-sfc-arch-00.txt

Abstract

This document describes an architecture for the creation of Service Function Chains. It includes architectural concepts, principles, and components used for the application of services in a network. This document does not propose solutions or 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 April 7, 2014.

Table of Contents

1. Introduction

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

1.1. Scope

The architecture described herein is assumed to be applicable to a single network administrative domain. While it is possible for the principals and architectural components to be applied to inter-domain service function chains, 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. Classification is performed by a classifier.

Network Locator (NL) : A name or address that uniquely identifies a Service Node. Some examples are IPv4, IPv6 or MAC addresses.

Service Function (SF): 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 an 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.

This document does not make any assumption in the OSI Layer on which the Service Function acts; the exact definition of each Service Function is deployment-specific. Further dissection into greater granularity (e.g. for example, sub-functions of the firewall service function) are out of scope for this document.

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 delivered.

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): The combination of a set of service functions that are to be applied to selected traffic in a specific order. The implied order may not be a linear progression as the architecture will allow for nodes that copy to more than one branch.

Service Chain (SC): A short form of Service Function Chain.

Service Traffic Recirculation (STR) : When traffic from a forward SN is forwarded back through a previous SF within the SFC in order to facilitate additional processing. Traffic may pass back and forth between SNs many times before being ultimately forwarded. Network operators or SNs must ensure that such configurations can handle infinite packet looping either via explicit configuration, or by enabling protocols that will.

1.3. Service Function Chaining

Service Function Chaining is a concept that implies more than just an ordered set of service functions, rather it describes a method for deploying service functions that enables not only ordering but topological independence of those service functions. A basic service function chain might simply utilize an existing overlay technology along with service specific forwarding in the network to steer traffic to the necessary service functions. However, additional information that is shared across a subset of service functions enables value added service functions and a richer service function chain. For example, shared information, such as the results of a classification function, may be passed to downstream service functions to enable the offloading of service function processing. As another example, sharing the information derived at one service function to the rest in the service chain would not only obviate the need to re-derive the same information but also simplifies the service as re-deriving may be impractical (for example when the packet may have been encrypted by an intermediate service function.)

2. Architectural Concepts

The following sections describe the core principles of a service function chaining infrastructure.

2.1. Service Function Chains

In most networks services are constructed as a sequence of service functions that represent a Service Function Chain. The collection of available service functions within an administrative domain forms a directed graph where the vertices represent an individual service function and the edges form the network connecting those vertices as partially represented in figure 1.

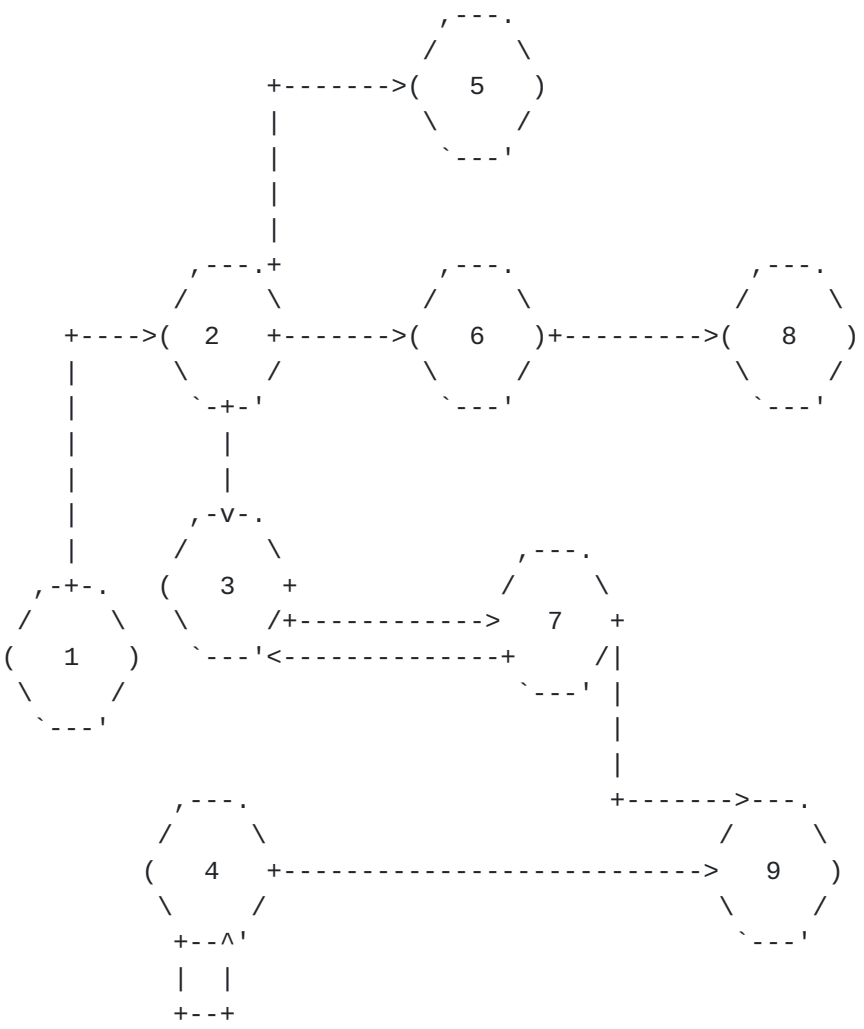

Figure 1: Service Function Chain Directed Graph

At a high level, service function chaining creates an abstracted view of a service and specifies the set of required service functions as well as the order in which they must be executed. Sub-graphs, from the overall directed graph, define each Service Function Chain. Service functions can be part of zero, one, or many service function chains.

Service function chains can start from the origination point of the service (i.e.: SN1 in Figure 1), or from any subsequent SN in the graph. SNs can therefore become branches in the graph, with those SFs performing forwarding decisions that move traffic to one or more branches. It is important to understand that multiple branches between nodes may exist, as with any network, and thus multicast as well as unicast forwarding paradigms are valid. Service function chains can have more than one terminus.

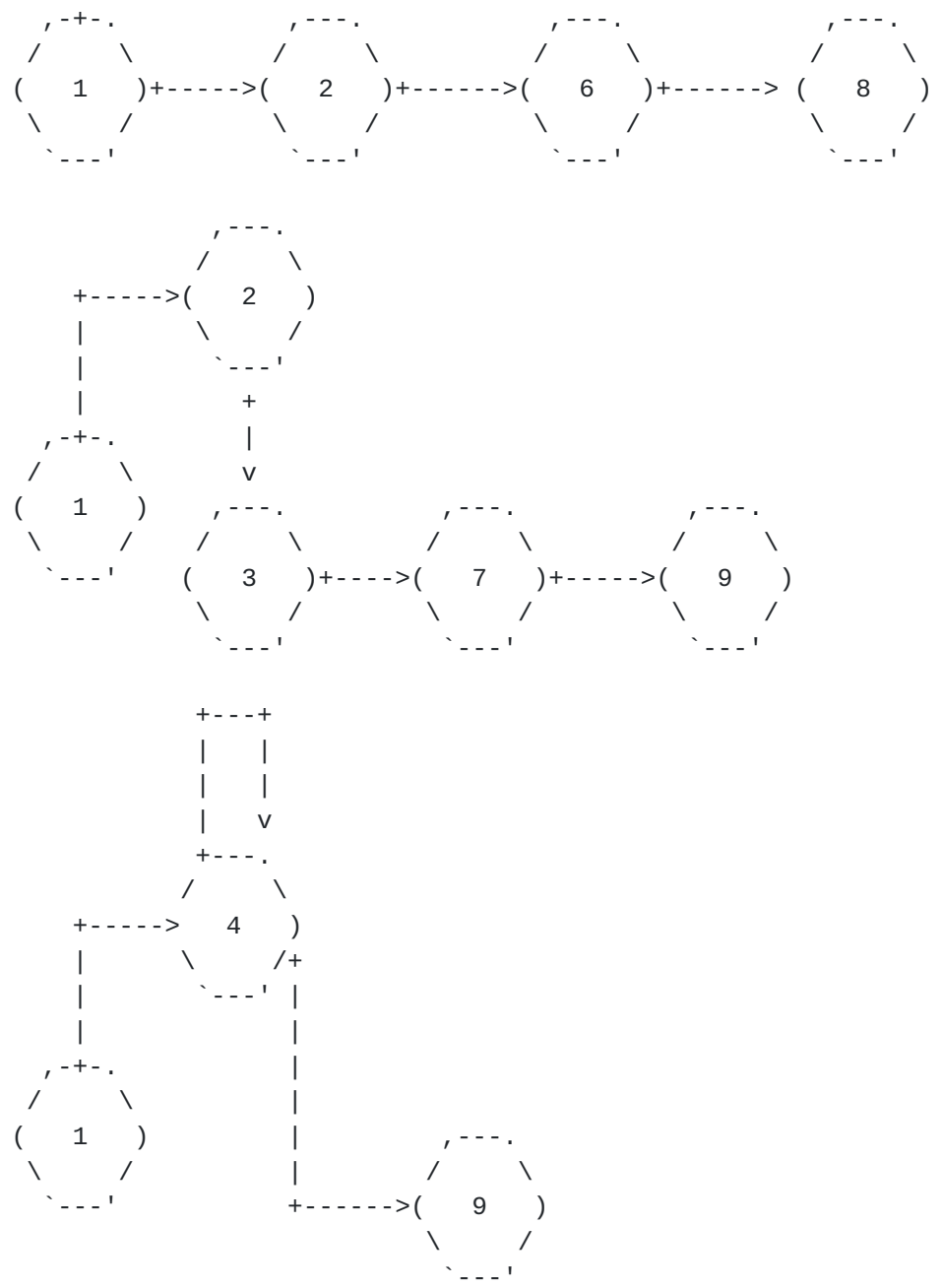

Figure 2: Service Function Chain Sub-Graphs

The cases in which two or more service functions are co-resident on the same service node and wish to implement some sort of internal, inter-process or inter-VM messaging (communication behind the virtual switching element) that is optimized for such an environment are out-of-scope for this document. Together, these entities are expected to handle the externalities of the chain (ingress header and egress

header handling) of the service function chain.

2.2. Service Function Chain Symmetry

Service Function Chains may be unidirectional or bidirectional. A unidirectional service function chain requires traffic to be forwarded through the ordered service functions in one direction (SF1 -> SF2 -> SF3), whereas a bidirectional service function chain requires a symmetric path (SF1 -> SF2 -> SF3 and SF3 -> SF2 -> SF1). A hybrid service function chain has attributes of both bidirectional and unidirectional service function chains: some service functions require symmetric traffic, other service functions do not process reverse traffic.

It is important to note that service function chains may point to themselves, effectively forming recursive paths. This is called Service Traffic Recirculation (STR). This therefore allows for loops within the topology. As with normal networks, it is important that nodes participating in the service path be able to detect and mitigate loops.

2.3. Service Function Paths

Service function chains, when instantiated in the network, lead to the selection of specific instances of service functions at various SNs as well as the creation of the service topology using the network locator of each individual SN. Thus, instantiation of the service function chain results in the creation of a Service Function Path and is used for forwarding packets through the service function chain. In other words, Service Function Path is the instantiation of the defined service function chain.

This abstraction enables the binding of service function chains to specific service function instances based on a range of policy attributes defined by the operator. For example, a service function chain definition might specify that one of the service function elements of the chain is a firewall. However, on the network, there might exist a number of instances of the same firewall service function (that is to say they enforce the same policy) and only when the service function path 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.

Classifiers can select the instances in the data path, offloading/ distributing the service instance load distribution functionality and improving the convergence time if there is a service instance failure. criteria.

3. Service Function Chaining Architecture

3.1. 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 service functions.

2. Consistent policy identifiers: per-service policy leverages policy identifier(s) that are consistent for the service function chain.

3. Classification: traffic that satisfies some classification rules may then be forwarded according to a specific SF chain. For example, classification can be as simple as an explicit forwarding entry that forwards all traffic from one address into the service function chain that starts on some interface of a forwarding entity. Multiple classification points are possible within a service function chain (i.e. forming a service graph) thus enabling changes/update to the path by functions.

4. Sharing of metadata/context: the network and service functions no longer exist in separate silos. Metadata/context data can be shared amongst all participating nodes.

5. No presumption of control point location. For example, the original chain may be described by an external control point to be *.* ---> SF1, where SF1 is a DPI device. This device can, in turn, use an independent mechanism (out of scope for this document) like IF-MAP or Diameter, to "pull the chain" based on the classifier or some integral policy, and select an existing or define a new branch (as described earlier in section 2.1).

3.2. Fundamental Components

Service function chaining can be divided into several components that together form the basis of the architecture:

1. Service Node: This is the embodiment of a service function, and can be instantiated within a physical or virtual element that hosts one or more service functions. These entities may have one or more network locators associated with them for reachability and service delivery.

2. Service Functions as Resources: The concept of a service function evolves: rather than being viewed as a bump in the wire, a service function becomes a resource within a specified administrative domain that is available for consumption. As such, service functions have a network locator and a variable set of attributes that describe the function offered. The combination of locator and attributes are used to construct a service function chain.

3. Classifier: A component that performs traffic classification. Classification is the precursor to the start of a service function path: traffic that matches classification criteria is forwarded along a given service function path to realize the specifications of a service function chain. The granularity of classification varies based on operator requirements and device capabilities. While initial classification at a network node starts a service function path, subsequent classifications may occur along the service function chain and further alter the service function path. This re-classification may also update the context information (see below).

4. Overlay Service Topology: A service topology is created to interconnect the elements used to form the service function path. This overlay topology is specific to the service function path: it is created for the express purpose of steering the service packets through the service functions and optionally passing context data. The overlay topology can be constructed using any existing transport, for example IP, MPLS, etc.

5. Control plane: The service function chaining control plane is responsible for constructing the service function paths: translating the service function chains 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 service function chain construction may be static - using specific service function instances, or dynamic - choosing service explicit function instances at the time of delivering traffic to the service function. In service function chaining, service functions 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 exist within distributed routing elements as in traditional networks, or in a centralized configuration.

6. Shared context data: Sharing context data allows the network to provide network-derived information to the service functions, service function to service function information exchange and the

sharing of service-derived information to the network. This component is optional. Service function chaining infrastructure enables the exchange of this shared context along the service function path. The shared context serves several key functions within the 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 service functions

Context information can be derived in several ways:

* External sources

* Network node classification

* Service function classification

7. 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 service functions running on the service nodes.

The figure below provides a high level view of the components:

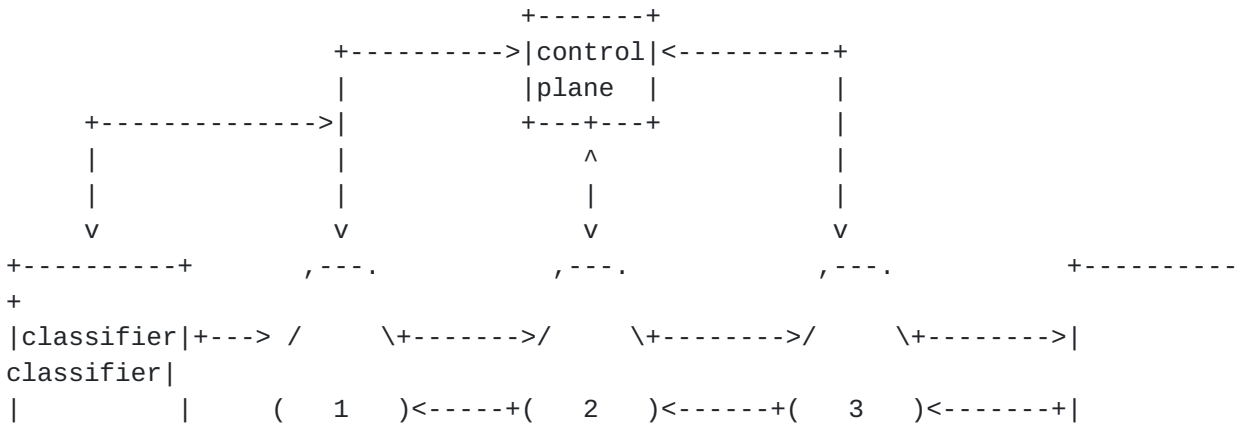

Figure 3: Service Function Chaining Architecture

4. 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.

5. Security Considerations

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

6. Acknowledgments

The authors would like to thank David Ward, Abhijit Patra, Nagaraj Bagepalli, Christian Jacquenet for their review and comments.

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.

7.2. Informative References

[NSCprob] "Network Service Chaining Problem Statement", <http://datatracker.ietf.org/doc/draft-quinn-nsc-problem-statement/>.

[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.

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].

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

Carlos Pignataro
Cisco Systems, Inc.

Email: cpignata@cisco.com

Nic Leymann
Deutsche Telekom

Email: n.leymann@telekom.de

Michael Smith
Insieme Networks

Email: michsmit@insiemenetworks.com

Navindra Yadav
Insieme Networks

Email: nyadav@insiemenetworks.com

Ken Gray
Juniper Networks

Email: kgray@juniper.net

Thomas Nadeau
Juniper Networks

Email: tnadeau@juniper.net