INTERNET-DRAFT
Intended Status: Informational
Expires: January 6, 2016

R. Fernando
Cisco
S. Mackie
Juniper
D. Rao
Cisco
B. Rijsman
Juniper
M. Napierala
AT&T

July 5, 2015

Service Chaining using Virtual Networks with BGP

draft-fm-bess-service-chaining-01

Abstract

This document describes how service function chains (SFC) can be applied to traffic flows using routing in a virtual (overlay) network to steer traffic between service nodes. Chains can include services running in routers, on physical appliances or in virtual machines. Service chains have applicability at the subscriber edge, business edge and in multi-tenant datacenters. The routing function into SFCs and between service functions within an SFC can be performed by physical devices (routers), be virtualized inside hypervisors, or run as part of a host OS.

A BGP control plane for route distribution is used to create virtual networks implemented using IP MPLS, VXLAN or other suitable encapsulation, where the routes within the virtual networks cause traffic to flow through a sequence of service nodes that apply packet processing functions to the flows. Two techniques are described: in one the service chain is implemented as a sequence of distinct VPNs between sets of service nodes that apply each service function; in the other, the routes within a VPN are modified through the use of special route targets and modified next-hop resolution to achieve the desired result.

In both techniques, service chains can be created by manual configuration of routes and route targets in routing systems, or through the use of a controller which contains a topological model of the desired service chains.

This document also contains discussion of load balancing between

network functions, symmetric forward and reverse paths when stateful services are involved, and use of classifiers to direct traffic into a service chain.

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/1id-abstracts.html

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

Copyright and License Notice

Copyright (c) 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

The purpose of networks is to allow computing systems to communicate with each other. Requests are usually made from the client or customer side of a network, and responses are generated by applications residing in a datacenter. Over time, the network between the client and the application has become more complex, and traffic between the client and the application is acted on by intermediate systems that apply network services. Some of these activities, like firewall filtering, subscriber attachment and network address translation are generally carried out in network devices along the traffic path, while others are carried out by dedicated appliances, such as media proxy and deep packet inspection (DPI). Deployment of these in-network services is complex, time-consuming and costly, since they require configuration of devices with vendor-specific operating systems, sometimes with co-processing cards, or deployment of physical devices in the network, which requires cabling and configuration of the devices that they connect to. Additionally, other devices in the network need to be configured to ensure that traffic is correctly steered through the systems that services are running on. The current mode of operations does not easily allow common operational processes to be applied to the lifecycle of services in the network, or for steering of traffic through them. The recent emergence of Network Functions Virtualization (NFV) [NFVE2E] to provide a standard deployment model for network services as software appliances, combined with Software Defined Networking (SDN) for more dynamic traffic steering can provide foundational elements that will allow network services to be deployed and managed far more efficiently and with more agility than is possible today. This document describes how the combination of several existing technologies can be used to create chains of functions, while preserving the requirements of scale, performance and reliability for service provider networks. The technologies employed are:

- Traffic flow between service functions described by routing and

network policies rather than by static physical or logical connectivity

o Packet header encapsulation in order to create virtual private networks using network overlays

o VRFs on both physical devices and in hypervisors to implement forwarding policies that are specific to each virtual network

o Optional use of a controller to calculate routes to be installed in routing systems to form a service chain. The controller uses a topological model that stores service function instance connectivity to network devices and intended connectivity between service functions.

o MPLS or other labeling to facilitate identification of the next interface to send packets to in a service function chain

o BGP or BGP-style signaling to distribute routes in order to create service function chains

o Distributed load balancing between service functions performed in the VRFs that service function instance connect to.

Virtualized environments can be supported without necessarily running BGP or MPLS natively. Messaging protocols such as NC/YANG, XMPP or OpenFlow may be used to signal forwarding information. Encapsulation mechanisms such as VXLAN or GRE may be used for overlay transport. The term "BGP-style", above, refers to this type of signaling.

Traffic can be directed into service function chains using IP routing at each end of the service function chain, or be directed into the chain by a classifier function that can determine which service chain a traffic flow should pass through based on deep packet inspection (DPI) and/or subscriber identity.

The techniques can support an evolution from services implemented in physical devices attached to physical forwarding systems (routers) to fully virtualized implementations as well as intermediate hybrid implementations.

1.1 Terminology

This document uses the following acronyms and terms.

Terms	Meaning
AS	Autonomous System
ASBR	Autonomous System Border Router
CE	Customer Edge
FW	Firewall
I2RS	Interface to the Routing System
L3VPN	Layer 3 VPN
LB	Load Balancer
NLRI	Network Layer Reachability Information [RFC4271]
P	Provider backbone router
proxy-arp	proxy-Address Resolution Protocol
RR	Route Reflector
RT	Route Target
SDN	Software Defined Network
vCE	virtual Customer Edge router
vFW	virtual Firewall
vLB	virtual Load Balancer
VM	Virtual Machine
vPC	virtual Private Cloud
vPE	virtual Provider Edge router
VPN	Virtual Private Network
VRF	VPN Routing and Forwarding table [RFC4364]
vRR	virtual Route Reflector

This document follows some of the terminology used in [draft-ietf-sfc-architecture] and adds some new terminology:

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 software executing in physical or virtual devices

Classification:
: Customer/network/service policy used to identify and select traffic flow(s) requiring certain outbound forwarding actions, in particular, to direct specific traffic flows into the ingress of a particular service function chain, or causing branching within a service function chain.

Virtual Network:

A logical overlay network built using virtual links or packet encapsulation, over an existing network (the underlay).

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. An SFC may be either a linear chain or a complex service graph with multiple branches. The term "Service Chain" is often used in place of "Service Function Chain".

SFC Set:

The pair of SFCs through which the forward and reverse directions of a given classified flow will pass.

Service Function (SF):
A logical function that is applied to packets. A service function can act at the network layer or other OSI layers. A service function can be embedded in one or more physical network elements, or can be implemented in one or more software instances running on physical or virtual hosts. One or multiple service functions can be embedded in the same network element or run on the same host. Multiple instances of a service function can be enabled in the same administrative domain. We will also refer to "Service Function" as, simply, "Service" for simplicity.

A non-exhaustive list of services includes: firewalls, DDOS protection, anti-malware/ant-virus systems, WAN and application acceleration, Deep Packet Inspection (DPI), server load balancers, network address translation, HTTP Header Enrichment functions, video optimization, TCP optimization, etc.

SF Instance:
An instance of software that implements the packet processing of a service function

SF Instance Set:
A group of SF instances that, in parallel, implement a service function in an SFC. Routing System: A hardware or software system that performs layer 3 routing and/or forwarding functions. The term includes physical routers as well as hypervisor or Host OS implementations of the forwarding plane of a conventional router.

VRF:
A subsystem within a routing system as defined in [RFC4364] that contains private routing and forwarding tables and has physical and/or logical interfaces associated with it. In the case of

hypervisor/Host OS implementations, the term refers only to the forwarding function of a VRF, and this will be referred to as a "VPN forwarder."

Ingress VRF:
A VRF containing an ingress interface of a SF instance

Egress VRF:
A VRF containing an egress interface of a SF instance

2 Service Function Chain Architecture Using Virtual Networking

The techniques described in this document use virtual networks to implement service function chains. Service function chains can be implemented on devices that support existing MPLS VPN and BGP standards [RFC4364, RFC4271, RFC4760], but other encapsulations, such as VXLAN [RFC7348], can be used. Similarly, equivalent control plane protocols such as BGP-EVPN can also be used where supported.

The following sections detail the building blocks of the SFC architecture, and outline the processes of route installation and subsequent route exchange to create an SFC.

2.1 High Level Architecture

Service function chains can be deployed with or without a classifier. Use cases where SFCs may be deployed without a classifier include multi-tenant data centers, private and public cloud and virtual CPE for business services. Classifiers will primarily be used in mobile and wireline subscriber edge use cases. Use of a classifier is discussed in Section 4.

A high-level architecture diagram of an SFC without a classifier, where traffic is routed into and out of the SFC, is shown in Figure 1, below. An optional controller is shown that contains a topological model of the SFC and which configures the network resources to implement the SFC.

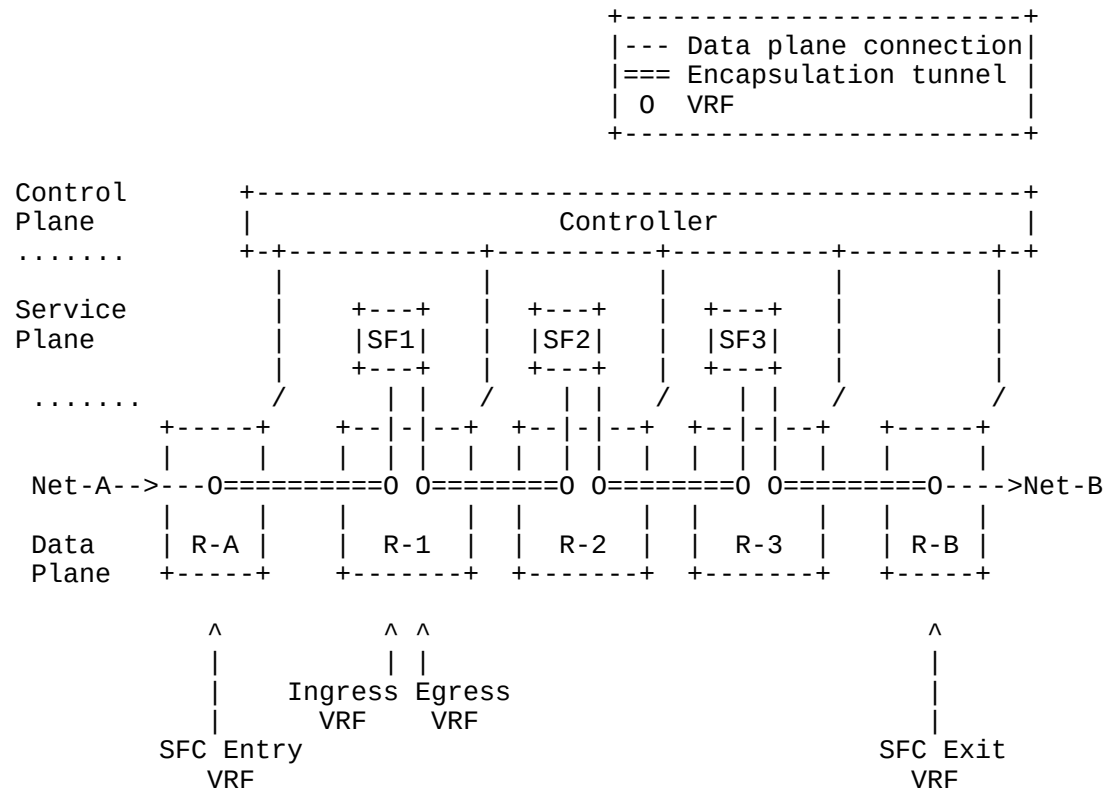


Figure 1 - High level SFC Architecture

Traffic from Network-A destined for Network-B will pass through the SFC composed of SF instances, SF1, SF2 and SF3. Routing system R-A contains a VRF (shown as "O" symbol) that is the SFC entry point. This VRF will advertise a route to reach Network-B into Network-A causing any traffic from a source in Network-A with a destination in Network-B to arrive in this VRF. The forwarding table in the VRF in R-A will direct traffic destined for Network-B into an encapsulation tunnel with destination R-1 and a label that identifies the ingress (left) interface of SF1 that R-1 should send the packets out on. The packets are processed by service instance SF-1 and arrive in the egress (right) VRF in R-1. The forwarding entries in the egress VRF direct traffic to the next ingress VRF using encapsulation tunneling. The process is repeated for each service instance in the SFC until packets arrive at the SFC exit VRF (in R B). This VRF is peered with Network-B and routes packets towards their destinations in the user data plane.

In the example, each pair of ingress and egress VRFs are configured

in separate routing systems, but such pairs could be collocated in the same routing system, and it is possible for the ingress and egress VRFs for a given SF instance to be in different routing systems. The SFC entry and exit VRFs can be collocated in the same routing system, and the service instances can be local or remote from either or both of the routing systems containing the entry and exit VRFs, and from each other.

The controller is responsible for configuring the VRFs in each routing system, installing the routes in each of the VRFs to implement the SFC, and, in the case of virtualized services, may instantiate the service instances.

2.2 Service Function Chain Logical Model

A service function chain is a set of logically connected service functions through which traffic can flow. Each egress interface of one service function is logically connected to an ingress interface of the next service function.

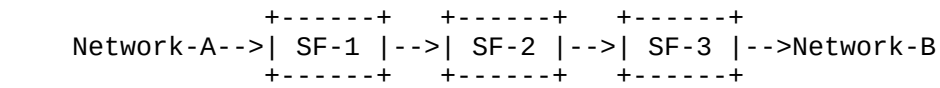


Figure 2 - A Chain of Service Functions

In Figure 2, above, a service function chain has been created that connects Network-A to Network-B, such that traffic from a host in Network-A to a host in Network-B will traverse the service function chain.

As defined in [draft-ietf-sfc-architecture], a service function chain can be uni-directional or bi-directional. In this document, in order to allow for the possibility that the forward and reverse paths may not be symmetrical, SFCs are defined as uni-directional, and the term "SFC set" is used to refer to a pair of forward and reverse direction SFCs for some set of routed or classified traffic.

2.3 Service Function Implemented in a Set of SF Instances

A service function instance is a software system that acts on packets that arrive on an ingress interface of that software system. Service function instances may run on a physical appliance or in a virtual machine. A service function instance may be transparent at layer 2 and/or 3, and may support branching across multiple egress interfaces and may support aggregation across ingress interfaces. For simplicity, the examples in this document have a single ingress and a single egress interface.

Each service function in a chain can be implemented by a single service function instance, or by a set of instances in order to provide scale and resilience.

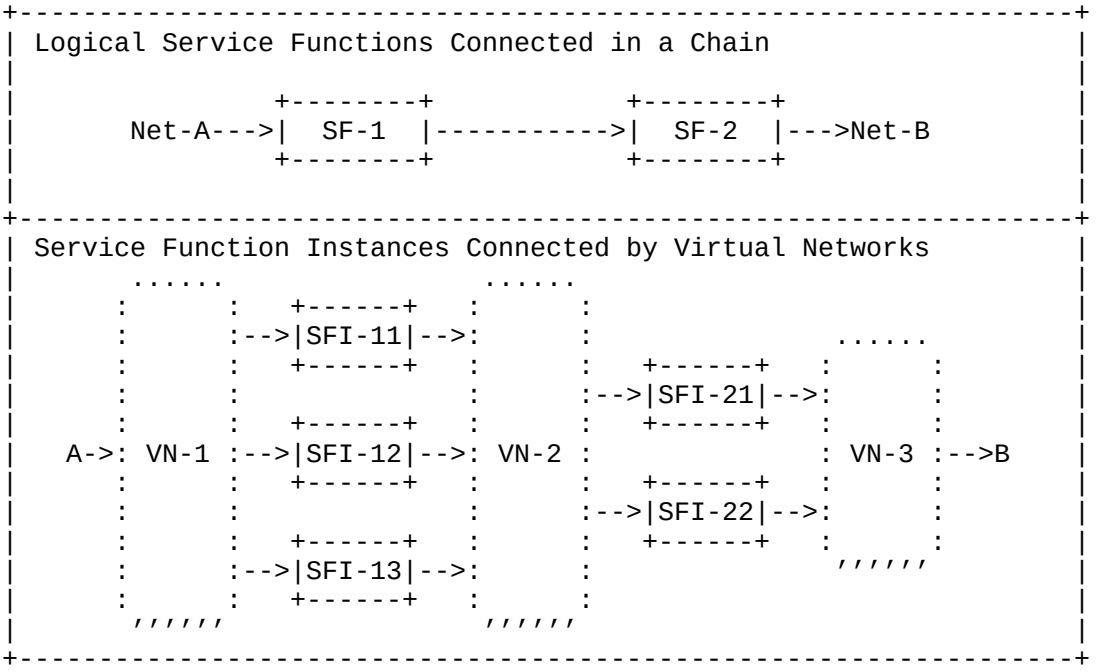


Figure 3 - Service Functions Are Composed of SF Instances Connected Via Virtual Networks

In Figure 3, service function SF-1 is implemented in three service function instances, SFI-11, SFI-12, and SFI-13. Service function SF-2 is implemented in two SF instances. The service function instances are connected to the next service function in the chain using a virtual network, VN-2. Additionally, a virtual network (VN-1) is used to enter the SFC and another (VN-3) is used at the exit.

The logical connection between two service functions is implemented using a virtual network that contains egress interfaces for instances of one service function, and ingress interfaces of instances of the next service function. Traffic is directed across the virtual network between the two sets of service function instances using layer 3 forwarding (e.g. an MPLS VPN) or layer 2 forwarding (e.g. a VXLAN).

The virtual networks could be described as "directed half-mesh", in

that the egress interface of each SF instance of one service function can reach any ingress interface of the SF instances of the connected service function.

Details on how routing across virtual networks is achieved, and requirements on load balancing across ingress interfaces are discussed in later sections of this document.

2.4 SF Instance Connections to VRFs

SF instances can be deployed as software running on physical appliances, or in virtual machines running on a hypervisor. These two options are described in more detail in the following sections.

2.4.1 SF Instance in Physical Appliance

The case of a SF instance running on a physical appliance is shown in Figure 4, below.

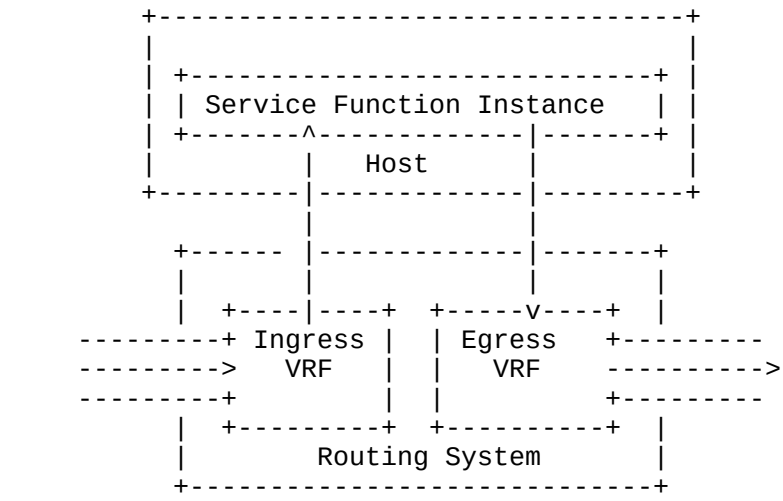


Figure 4 - Ingress and Egress VRFs for a Physical Routing System and Physical SF Instance

The routing system is a physical device and the service function instance is implemented as software running in a physical appliance (host) connected to it. Transport between VRFs on different routing systems that are connected to other SF instances in an SFC is via encapsulation tunnels, such as MPLS over GRE, or VXLAN.

2.4.2 SF Instance in a Virtualized Environment

In virtualized environments, a routing system with VRFs that act as VPN forwarders is resident in the hypervisor/Host OS, and is co-resident in the host with one or more SF instances that run in virtual machines. The egress VPN forwarder performs tunnel encapsulation to send packets to other physical or virtual routing systems with attached SF instances to form an SFC. The tunneled packets are sent through the physical interfaces of the host to the other hosts or physical routers. This is illustrated in Figure 5, below.

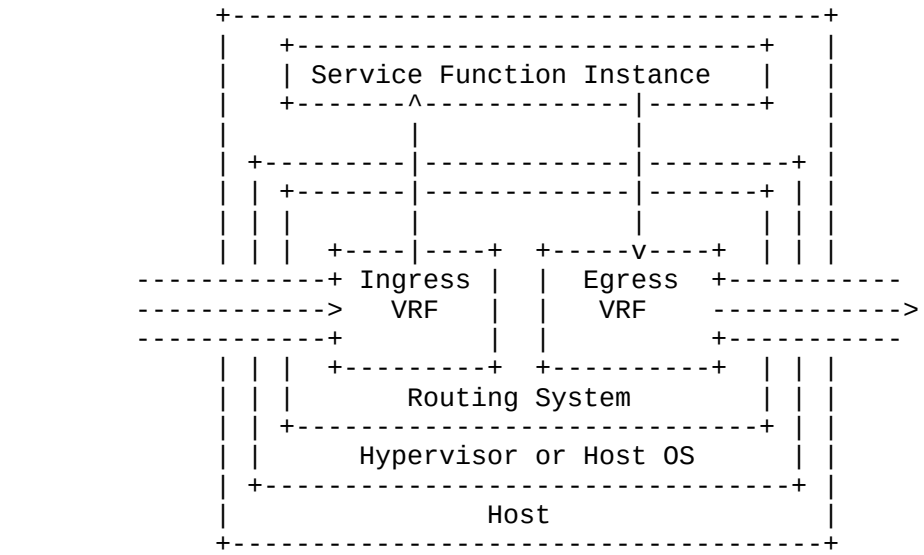


Figure 5 - Ingress and Egress VRFs for a Virtual Routing System and Virtualized SF Instance

When more than one instance of an SF is running on a hypervisor, they can be connected to the same VRF for scale out of an SF within an SFC.

The routing mechanisms in the VRFs into and between service function instances, and the encapsulation tunneling between routing systems are identical in the physical and virtual implementation of SFCs described in this document. Physical and virtual service functions can be mixed as needed with different combinations of physical and virtual routing systems.

2.5 Encapsulation Tunneling for Transport

Encapsulation tunneling is used to transport packets between SF

instances in the chain and, when a classifier is not used, from the originating network into the SFC and from the SFC into the destination network.

The tunnels can be MPLS over GRE [RFC4023], MPLS over UDP [draft-ietf-mpls-in-udp], MPLS over MPLS [RFC3031], VXLAN [RFC7348], or another suitable encapsulation method.

Tunneling may be enabled in each routing system as part of a base configuration or may be configured by the controller. Tunnel encapsulations may be configured by the controller or signaled using BGP.

2.6 SFC Creation Procedure

This section describes how service chains are created using two methods:

o Sequential VPNs - where a conventional VPN is created between each set of SF instances to create the links in the SFC

o Route Modification - where each routing system modifies advertised routes that it receives, to realize the links in an SFC on the basis of a special service topology RT and a route-policy that describes the service chain logical topology

In both cases the controller, when present, is responsible for creating ingress and egress VRFs, configuring the interfaces connected to SF instances in each VRF and configuring RTs for each VRF. Additionally, in the second method, the controller also sends the route-policy containing the service chain logical topology to each routing system. If a controller is not used, these procedures will require to be performed manually or through scripting, for instance.

The following sub-sections describe how RT configuration, local route installation and route distribution occurs in each of the methods.

2.6.1 SFC Provisioning Using Sequential VPNs

The task of the controller in this method of SFC provisioning is to create a set of VPNs that carry traffic to the destination network through instances of each service function in turn. This is achieved by configuring RTs such that the egress VRFs of one set of SF instances import an RT that is an export RT for the ingress VRFs of the next, logically connected, set of SF instances.

The process of SFC creation is as follows

1. Controller creates a VRF in each routing system that is connected to a service instance that will be used in the SFC

2. Controller configures each VRF to contain the logical interface that connects to a SF instance.

3. Controller implements route target import and export policies in the VRFs using the same route targets for the egress VRFs of a service function and the ingress VRFs of the next logically connected service function in the SFC.

4. Controller installs a static route in each ingress VRF whose next hop is the interface that a SF instance is connected to. The prefix for the route is the destination network to be reached by passing through the SFC.

5. Routing systems advertise the static routes via BGP as VPN routes with next hop being the IP address of the router, with an encapsulation specified and a label that identifies the service instance interface.

6. Routing systems containing VRFs with matching route targets receive the updates.

7. Routes are installed in egress VRFs with matching import targets. The egress VRFs of each SF instance will now contain VPN routes to one or more routers containing ingress VRFs for SF instances of the next service function in the SFC.

In the case of physical routers, the creation and configuration of VRFs, interfaces and local static routes can be performed programmatically using Netconf; and BGP route distribution can use a route reflector (which may be part of the controller). In the virtualized case, where a VPN forwarder is present, creation and configuration of VRFs, interfaces and installation of routes can be performed using a single protocol like XMPP, NC/YANG or an equivalent programmatic interface.

Also in the virtualized case, routes in the ingress and egress VRFs can be calculated by the controller based on its internal knowledge of the required SFC topology and the connectivity of SF instances to routing systems. In this case the routes are directly installed and no route advertisement is necessary.

As discussed further in Section 3, egress VRFs can load balance across the multiple next hops advertised from the next set of ingress VRFs.

Routes to the destination network via the first set of SF instances are advertised to the gateway router for the source network, and the egress VRFs of the last SF instance set have routes via the destination network gateway router.

2.6.2 Modified-Route SFC Creation

In this method of SFC configuration, all the VRFs connected to SF instances are configured with same import and export RT, so they form a VPN-connected mesh between the SF instance interfaces. This is termed the "Service VPN". A route is configured or learnt in each VRF with destination being the IP address of the connected SF instance via an interface configured in the VRF. The interface may be a physical or logical interface. The routing system that hosts such a VRF advertises a VPN route for each locally connected SF instance, with a forwarding label that enables it to forward incoming traffic from other routing systems to the connected SF instance. The VPN routes may be advertised via an RR or the controller, which then sends these updates to all the other routing systems that have VRFs with the service VPN RT. At this point all the VRFs have a route to reach every SF instance. The same IP address is used for each SF instance in a set, enabling load-balancing among multiple SF instances in the set.

The controller sends a route-policy to each routing system in the VPN, that describes the logical topology of each service chain that it belongs to. The route-policy contains entries in the form of a tuple for each service chain:

{Service-topology-name, Service-topology-RT, Service-node-sequence} where Service-node-sequence is simply an ordered list of the service function instance IP addresses that are in the chain.

Every service function chain has a single unique service-topology-RT that is provisioned on all participating routing systems in the relevant VRFs.

The VRF in the routing system that connects to the destination network is configured to attach the Service-topology-RT to exported routes, and the VRF in the gateway router of the source network will import routes using Service-topology-RT. A controller may also be used to originate the Service-topology-RT attached routes.

Route-policies may be described in a variety of formats in addition to that described above. For instance, it would be possible to use YANG as a modeling language.

Using Figure 1 for reference, when the gateway R-B advertises a VPN

route to Network-B, it attaches the Service-topology-RT. BGP route updates are sent to all the routing systems in the service VPN. The routing systems perform a modified set of actions for next-hop resolution and route installation in the ingress VRFs compared to normal BGP VPN behavior in routing systems, but no changes are required in the operation of the BGP protocol itself. The modification of behavior in the routing systems allows the automatic and constrained flow of traffic through the service chain.

Each routing system in the service VPN will process the VPN route to Network-B via R-B as follows:

1. If the routing system contains VRFs that import the Service-topology-RT, continue, otherwise ignore the route.

2. The routing system identifies the position and role (ingress/egress) of each of its VRFs in the SFC by comparing the IP address of the route in the VRF to the connected SF instance with those in the Service-node-sequence in the route-policy. Alternatively, the controller may provision the specific service node IP to be used as the next-hop in each VRF, in the route-policy.

3. The routing system modifies the next-hop of the imported route with the Service-topology-RT, to select the appropriate next-hop as per the route-policy. It ignores the next-hop and label in the received route. It resolves the selected next-hop in the VRF routing table.

   a. The imported route to Network-B in the ingress VRF is modified to have a next-hop of the IP address of the logically connected SF instance.

   b. The imported route to Network-B in the egress VRF is modified to have a next hop of the IP address of the next SF instance in the SFC.

4. The egress VRFs for the last service function install the VPN route via the gateway R-B unmodified.

Note that the modified routes are not re-advertised into the VPN by the various routing systems in the SFC.

Similar to the sequential VPN method, VRF configuration and creation, and routing-policy installation can be performed manually or via scripting, or a controller could be used to automate the process.

2.7 Controller Function

The purpose of the controller is to manage instantiation of SFCs in networks and datacenters. When an SFC is to be instantiated, a model of the desired topology (service functions, number of instances, connectivity) is built in the controller either via an API or GUI. The controller then selects resources in the infrastructure that will support the SFC and configures them. This can involve instantiation of SF instances to implement each service function, the instantiation of VRFs that will form virtual networks between SF instances, and installation of routes to cause traffic to flow into and between SF instances.

For simplicity, in this document, the controller is assumed to contain all the required features for management of SFCs. In actual implementations, these features may be distributed among multiple inter-connected systems. E.g. An overarching orchestrator might manage the overall SFC model, sending instructions to a separate virtual machine manager to instantiate service function instances, and to a virtual network manager to set up the service chain connections between them.

The controller can also perform necessary BGP signaling and route distribution actions as described throughout this document.

2.8 Variations on Setting Prefixes in an SFC

2.8.1 Variation 1

In the configuration methods described above, the network prefixes for each network (Network-A and Network-B in the example above) connected to the SFC are used in the routes that direct traffic through the SFC. This creates an operational linkage between the implementation of the SFC and the insertion of the SFC into a network.

For instance, subscriber network prefixes will normally be segmented across subscriber attachment points such as broadband or mobile gateways. This means that each SFC would have to be configured with the subscriber network prefixes whose traffic it is handling.

In a variation of the SFC configuration method described above, the prefixes used in each direction can be such that they include all possible addresses at each side of the SFC. For example, in Figure 1, the prefix for Network-A could include all subscriber IP addresses and the prefix for Network-B could be the default route, 0/0.

Using this technique, the same routes can be installed in all instances of an SFC that serve different groups of subscribers in different geographic locations.

The routes forwarding traffic into a SF instance and to the next SF instance are installed when an SFC is initially built, and each time a SF instance is connected into the SFC, but there is no requirement for VRFs to be reconfigured when traffic from different networks pass through the service chain, so long as their prefix is included in the prefixes in the VRFs along the SFC.

In this variation, it is assumed that no subscriber-originated traffic will enter the SFC destined for an IP address also in the subscriber network address range. This will not be a restriction in many cases.

2.8.2 Variation 2

As another slight variation of the above, a network prefix may be disaggregated and spread out among various gateway routers, for instance, in the case of virtual machines in a data-center. In order to reduce the scaling requirements on the routing systems along the SFC, the aggregate network prefix may be advertised with the Service-topology-RT and used in the traffic forwarding along the SFC.

Where there is a gateway router for the destination network that can aggregate the prefixes, none of the routing systems along the SFC need to receive the more-specific routes. If there is not, the service chain can be divided into two parts such that only the egress VRFs of the last SF instance import the more specific routes; and the rest of the VRFs only import the aggregate prefix. For instance, this may be done by using two different Service-topology-RTs for more-specific and aggregate routes.

In the simplest case, a default route is used to direct forwarding along the SFC upto the last SF instance, while the source network’s gateway routers and the egress VRF of the last SF instance use the destination network’s prefixes.

2.9 Header Transforming Service Functions

If a service function performs an action that changes the source address in the packet header (e.g., NAT), the routes that were installed as described above may not support reverse flow traffic. The solution to this is for the controller modify the routes in the reverse direction to direct traffic into instances of the transforming service function. The original routes with a source prefix (Network-A in Figure 2) are replaced with a route that has a

prefix that includes all the possible addresses that the source address could be mapped to. In the case of network address translation, this would correspond to the NAT pool.

3 Load Balancing Along a Service Function Chain

One of the key concepts driving NFV [NFVE2E]is the idea that each service function along an SFC can be separately scaled by changing the number of service function instances that implement it. This requires that load balancing be performed before entry into each service function. In this architecture, load balancing is performed in either or both of egress and ingress VRFs depending on the type of load balancing being performed, and if more than one service instance is connected to the same ingress VRF.

3.1 SF Instances Connected to Separate VRFs

If SF instances implementing a service in an SFC are each connected to separate VRFs(e.g. instances are connected to different routers or are running on different hosts), load balancing is performed in the egress VRFs of the previous service, or in the VRF that is the entry to the SFC. The controller distributes BGP multi-path routes to the egress VRFs. The destination prefix of each route is the ultimate destination network, or its representative aggregate or default. The next-hops in the ECMP set are BGP next-hops of the service instances attached to ingress VRFs of the next service in the SFC. The load balancing corresponds to BGP Multipath, which requires that the route distinguishers for each route are distinct in order to recognize that distinct paths should be used. Hence, each VRF in a distributed, SFC environment should have a unique route distinguisher.

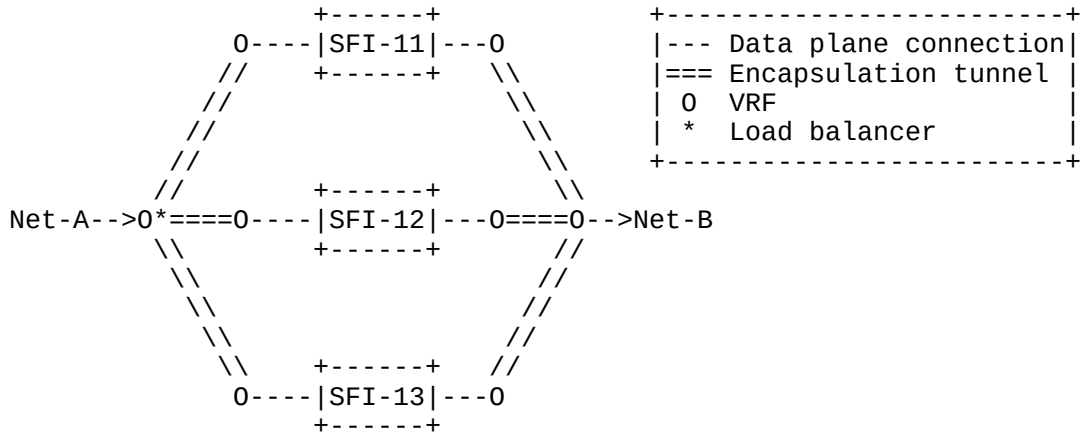


Figure 6 - Load Balancing across SF Instances Connected to Different VRFs

In the diagram, above, a service function is implemented in three service instances each connected to separate VRFs. Traffic from Network-A arrives at VRF at the start of the SFC, and is load balanced across the service instances using a set of ECMP routes with next hops being the addresses of the routing systems containing the ingress VRFs and with labels that identify the ingress interfaces of the service instances.

3.2 SF Instances Connected to the Same VRF

When SF instances implementing a service in an SFC are connected to the same ingress VRF, load balancing is performed in the ingress VRF across the service instances connected to it. The controller will install routes in the ingress VRF to the destination network with the interfaces connected to each service instance as next hops. The ingress VRF will then use ECMP to load balance across the service instances.

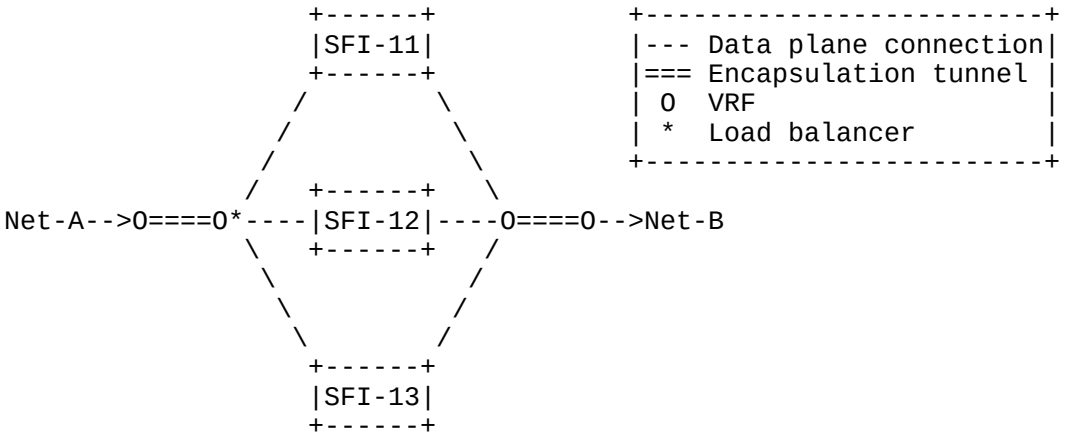


Figure 7 - Load Balancing across SF Instances Connected to the Same VRF

In the diagram, above, a service is implemented by three service instances that are connected to the same ingress and egress VRFs. The ingress VRF load balances across the ingress interfaces using ECMP, and the egress traffic is aggregated in the egress VRF.

3.3 Combination of Egress and Ingress VRF Load Balancing

In Figure 8, below, an example SFC is shown where load balancing is performed in both ingress and egress VRFs.

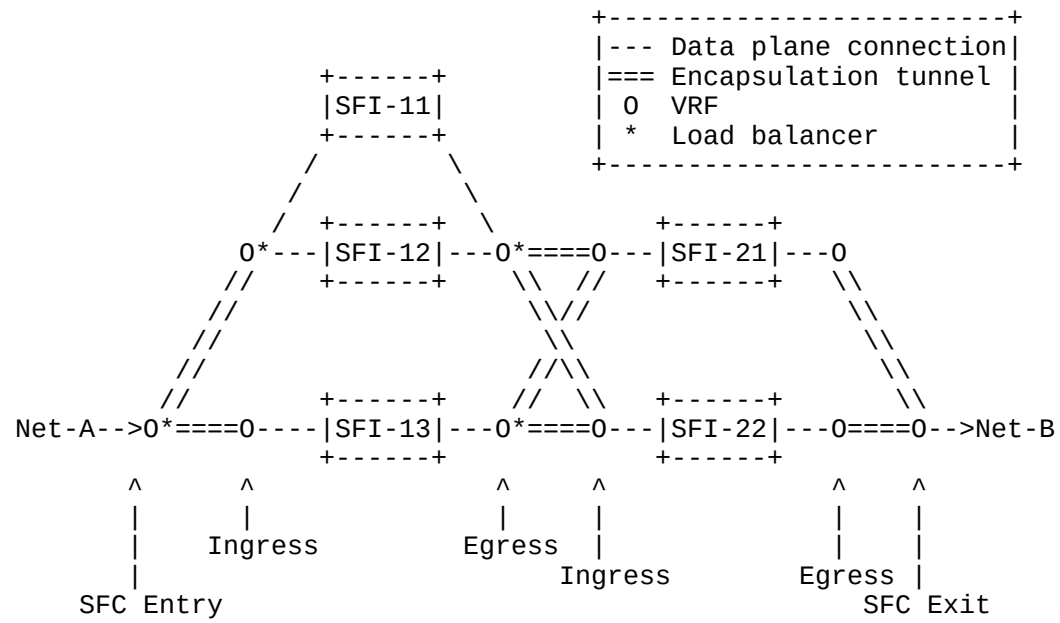


Figure 8 - Load Balancing across SF Instances

In Figure 8, above, an SFC is composed of two services implemented by three service instances and two service instances, respectively. The service instances SFI-11 and SFI-12 are connected to the same ingress and egress VRFs, and all the other service instances are connected to separate VRFs.

Traffic entering the SFC from Network-A is load balanced across the ingress VRFs of the first service function by the chain entry VRF, and then load balanced again across the ingress interfaces of SFI-11 and SFI-12 by the shared ingress VRF. Note that use of standard ECMP will lead to an uneven distribution of traffic between the three service instances (25% to SFI-11, 25% to SFI-12, and 50% to SFI-13). This issue can be mitigated through the use of BGP link bandwidth extended community [draft-ietf-idr-link-bandwidth].

After traffic passes through the first set of service instances, it is load balanced in each of the egress VRFs of the first set of service instances across the ingress VRFs of the next set of service

instances.

3.4 Forward and Reverse Flow Load Balancing

This section discusses requirements in load balancing for forward and reverse paths when stateful service functions are deployed.

3.4.1 Issues with Equal Cost Multi-Path Routing

As discussed in the previous sections, load balancing in the forward SFC in the above example can automatically occur with standard BGP, if multiple equal cost routes to Network-B are installed into all the ingress VRFs, and each route directs traffic through a different service function instance in the next set. The multiple BGP routes in the routing table will translate to Equal Cost Multi-Path in the forwarding table. The hash used in the load balancing algorithm (per packet, per flow or per prefix) is implementation specific.

If a service function is stateful, it is required that forward flows and reverse flows always pass through the same service function instance. ECMP does not provide this capability, since the hash calculation will see different input data for the same flow in the forward and reverse directions (since the source and destination fields are reversed).

Additionally, if the number of SF instances changes, either increasing to expand capacity, or decreases (planned, or due to a SF instance failure), the hash table in ECMP is recalculated, and most flows will be directed to a different SF instance and user sessions will be disrupted.

There are a number of ways to satisfy the requirements of symmetric forward/reverse paths for flows and minimal disruption when SF instances are added to or removed from a set. Two techniques that can be employed are described in the following sections.

3.4.2 Modified ECMP with Consistent Hash

Symmetric forwarding into each side of an SF instance set can be achieved with a small modification to ECMP if the packet headers are preserved after passing through a SF instance set. In this case, each packet’s 5-tuple data can be used in a hashing function, provided the source and destination IP address and port information are swapped in the reverse calculation and that the same or no hash salt is used for both directions. This method only requires that the list of available service function instances is consistently maintained in all the load balancers, rather than maintaining a distributed flow table.

In the SFC architecture described in this document, when SF instances are added or removed, the controller is required to configure (or remove) static routes to the SF instances. The controller could configure the load balancing function in VRFs that connect to each added (or removed) SF instance as part of the same network transaction as route updates to ensure that the load balancer configuration is synchronized with the set of SF instances.

The effect of rehashing when SF instances are added or removed can be minimized, or even eliminated using variations of the technique of consistent hashing [consistent-hash]. Details are outside the scope of this document.

3.4.3 ECMP with Flow Table

A second refinement that can ensure forward/reverse flow consistency, and also provides stability when the number of SF instances changes ("flow-stickiness"), is the use of dynamically configured IP flow tables in the VRFs. In this technique, flow tables are used to ensure that existing flows are unaffected if the number of ECMP routes changes, and that forward and reverse traffic passes through the same SF instance in each set of SF instances implementing a service function.

The flow tables are set up as follows:

1. User traffic with a new 5-tuple enters an egress VRF from a connected SF instance.

2. The VRF calculates the ECMP hash across available routes (i.e., ECMP group) to the ingress interfaces of the SF instances in the next SF instance set.

3. The VRF creates a new flow entry for the 5-tuple traffic with the next-hop being the chosen downstream ECMP group member (determined in the step 2. above) . All subsequent packets for the same flow will be forwarded using flow lookup and, hence, will use the same next-hop.

4. The encapsulated packet arrives in the routing system that hosts the ingress VRF for the selected SF instance.

5. The ingress VRF of the next service instance determines if the packet came from a routing system that is in an ECMP group in the reverse direction(i.e., from this ingress VRF back to the previous set of SF instances).

6. If an ECMP group is found, the ingress VRF creates a reverse flow entry for the 5-tuple with next-hop of the tunnel on which

traffic arrived.

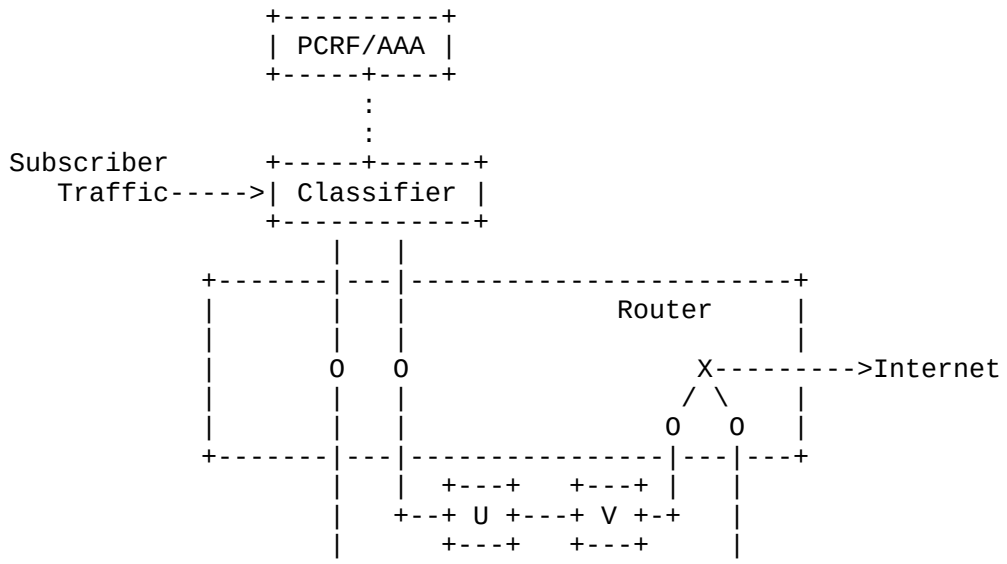
7. The packet is sent into the SF instance connected to the ingress VRF.

The above method ensures that forward and reverse flows pass through the same SF instances, and that if the number of ECMP routes changes when SF instances are added or removed, all existing flows will continue to flow through the same SF instances, but new flows will use the new ECMP hash. The only flows affected will be those that were passing through an SF instance that was removed, and those will be spread among the remaining SF instances using the updated ECMP hash.

4 Steering into SFCs Using a Classifier

In many applications of SFCs, a classifier will be used to direct traffic into SFCs. The classifier inspects the first or first few packets in a flow to determine which SFC the flow should be sent into. The decision criteria can include the IP 5-tuple of the header, and/or analysis of the payload of packets using deep packet inspection. Integration with a subscriber management system such as PCRF or AAA will usually be required in order to identify which SFC to send traffic to based on subscriber policy.

An example logical architecture is shown in Figure 9, below where a classifier is external to a physical router.



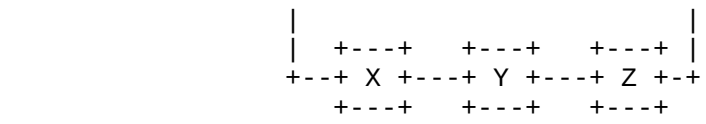


Figure 9 - Subscriber/Application-Aware Steering with a Classifier

In the diagram, the classifier receives subscriber traffic and sends the traffic out of one of two logical interfaces, depending on classification criteria. The logical interfaces of the classifier are connected to VRFs in a router that are entries to two SFCs (shown as O in the diagram).

In this scenario, the exit VRF for each SFC does not peer with a gateway or proxy node in the destination network and packets are forwarded using IP lookup in the main routing table or in a VRF that the exit traffic from the SFCs is directed into (shown as X in the diagram).

An alternative would be where the classifier is itself a distributed, virtualized service function, but with multiple egress interfaces. In that case, each virtual classifier instance could be attached to a set of VRFs that connect to different SFCs. Each chain entry VRF would load balance across the first SF instance set in its SFC. The reverse flow table mechanism described in Section 3.4.3 could be employed to ensure that flows return to the originating classifier instance which may maintain subscriber context and perform charging and accounting.

5 External Domain Co-ordination

It is likely that SFCs will be managed as a separate administrative domain from the networks that they receive traffic from, and send traffic to. If the connected networks use BGP for route distribution, the controller in the SFC domain can join the network domains by creating BGP peering sessions with routing systems or route reflectors in those network domains.

In order to steer traffic from the network domains into an SFC, the controller will advertise a destination network’s prefixes into the peering network domain with a BGP next-hop and label associated with the SFC entry point, that may be on a routing system attached to the first SF instance. This advertisement may be over regular MP-BGP/VPN peering which assumes existing standard VPN routing/forwarding behavior on the network domain’s routers (PEs/ASBRs).

An operational benefit of this approach is also that the SFC topology within a domain need not be exposed to other domains.

6 Fine-grained steering using BGP Flow-Spec

When steering traffic from a network domain's existing routing systems into an SFC is desired based on attributes of the packet flow, [FLOWSPEC] is a signaling option that can be used. In this case, the controller advertises a flow-spec route to the network domain's routing systems or route reflectors with the appropriate next-hop or Service-topology-RT for the SFC entry point.

7 BGP-EVPN signaling

In a DC environment, routing systems are likely to use VXLAN based overlays and a BGP EVPN control plane (DC-OVERLAY). For the solution designs described earlier in the document, the BGP VPN routes for both the SF instances and the destination networks are advertised via BGP-EVPN, using type-2 and type-5 route types.

8 Controller Federation

When SFCs are distributed geographically, or in very large-scale environments, there may be multiple SFC controllers present. If there is a requirement for SFCs to span controller domains there may be a requirement to exchange information between controllers. Again, a BGP session between controllers can be used to exchange route information as described in the previous sections and allow such domain spanning SFCs to be created.

9 Summary and Conclusion

The architecture for service function chains described in this document uses virtual networks implemented as overlays in order to create service function chains. The virtual networks use standards-based encapsulation tunneling, such as MPLS over GRE/UDP or VXLAN, to transport packets into an SFC and between service function instances without routing in the user address space. Two methods of installing routes to form service chains are described.

In environments with physical routers, a controller may operate in tandem with existing BGP route reflectors, and would contain the SFC topology model, and the ability to install the local static interface routes to SF instances. In a virtualized environment, the controller can emulate route refection internally and simply install required routes directly without advertisements occurring.

10 Security Considerations

The security considerations for SFCs are broadly similar to those concerning the data, control and management planes of any device placed in a network. Details are out of scope for this document.

11 IANA Considerations

There are no IANA considerations.

12 Acknowledgements

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

This document is a merged specification based on earlier drafts [draft-rfernando-bess-service-chaining] and [draft-mackie-sfc-using-virtual-networking].

The authors would like to thank D. Daino, D.R. Lopez, D. Bernier, W. Haeffner, A. Farrel, L. Fang, and N. So, for their contributions to the earlier drafts. The authors would also like to thank the following individuals for their review and feedback on the original proposals: E. Rosen, J. Guchard, P. Quinn, P. Bosch, D. Ward, A. Ganesan, T. Morin, N. Seth, G. Pildush and N. Bitar.

13 References

13.1 Normative References

None

13.2 Informative References

[NFVE2E] "Network Functions Virtualisation: End to End Architecture, http://docbox.etsi.org/ISG/NFV/70-DRAFT/0010/NFV-0010v016.zip".

[RFC2328] J. Moy, "OSPF Version 2", RFC 2328, April, 1998.

[draft-merged-sfc-architecture] Halpern, J. and Pignataro, C., "Service Function Chaining (SFC) Architecture", draft-ietf-sfc-architecture-09 June 2015.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, February 2006.

[RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.

[RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, January 2007.

[RFC7348] Mahalingam, M., et al. "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks.", RFC 7348, August 2014.

[draft-ietf-l3vpn-end-system] Marques, P., et al., "BGP-signaled end-system IP/VPNs", draft-ietf-l3vpn-end-system-04, October 2, 2014.

[FLOWSPEC] Marques, P., Sheth, N., Raszuk, R., et al., "Dissemination of Flow Specification Rules", RFC 5575, August 2009.

[draft-ietf-bess-evpn-overlay-01] A. Sajassi, et al, "A Network Virtualization Overlay Solution using EVPN", draft-ietf-bess-evpn-overlay, February 2015.

[draft-ietf-sfc-nsh] Quinn, P., et al, "Network Service Header", draft-ietf-sfc-nsh-00, March 2015.

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

[draft-rijsman-sfc-metadata-considerations] B. Rijsman, et al. "Metadata Considerations", draft-rijsman-sfc-metadata-considerations-00, February 12, 2014

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC4023] Worster, T., Rekhter, Y., and E. Rosen, "Encapsulating MPLS in IP or Generic Routing Encapsulation (GRE)", RFC 4023, March 2005.

[RFC7510] Xu, X., Sheth, N. et al, "Encapsulating MPLS in UDP", RFC 7510, April 2015.

[draft-ietf-i2rs-architecture] Atlas, A., Halpern, J., Hares, S., Ward, D., and T Nadeau, "An Architecture for the Interface to the Routing System", draft-ietf-i2rs-architecture, work in progress, March 2015.

[consistent-hash] Karger, D.; Lehman, E.; Leighton, T.; Panigrahy, R.; Levine, M.; Lewin, D. (1997). "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web". Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing. ACM Press New York, NY, USA. pp. 654-663.

[draft-ietf-idr-link-bandwidth] P. Mohapatra, R. Fernando, "BGP Link Bandwidth Extended Community", draft-ietf-idr-link-

bandwidth, work in progress.

[I-D.fang-l3vpn-virtual-pe]
L. Fang, et al., "BGP/MPLS IP VPN Virtual PE",
draft-fang-l3vpn-virtual-pe, work in progress.

[I-D.ietf-i2rs-problem-statement]
Atlas, A., Nadeau, T., and D. Ward, "Interface to the Routing System Problem Statement",
draft-ietf-i2rs-problem-statement, work in progress.

Authors' Addresses

Rex Fernando
Cisco
170 W Tasman Dr
San Jose, CA
Email: rex@cisco.com

Stuart Mackie
Juniper Networks
1133 Innovation Way
Sunnyvale, CA 94089
USA
Email: wsmackie@juniper.net

Dhananjaya Rao
Cisco
170 W Tasman Dr
San Jose, CA
Email: dhrao@cisco.com

Bruno Rijsman
Juniper Networks
1133 Innovation Way
Sunnyvale, CA 94089
USA
Email: brijsman@juniper.net

Maria Napierala
AT&T
200 Laurel Avenue
Middletown, NJ 07748
Email: mnapierala@att.com