Stream: Internet Engineering Task Force (IETF)
RFC: 9015
Category: Standards Track
Published: May 2021
ISSN: 2070-1721
Authors:
A. Farrel, *Old Dog Consulting*
J. Drake, *Juniper Networks*
E. Rosen, *Juniper Networks*
J. Uttaro, *AT&T*
L. Jalil, *Verizon*

RFC 9015
BGP Control Plane for the Network Service Header in Service Function Chaining

Abstract

This document describes the use of BGP as a control plane for networks that support service function chaining. The document introduces a new BGP address family called the "Service Function Chain (SFC) Address Family Identifier / Subsequent Address Family Identifier" (SFC AFI/SAFI) with two Route Types. One Route Type is originated by a node to advertise that it hosts a particular instance of a specified service function. This Route Type also provides "instructions" on how to send a packet to the hosting node in a way that indicates that the service function has to be applied to the packet. The other Route Type is used by a controller to advertise the paths of "chains" of service functions and give a unique designator to each such path so that they can be used in conjunction with the Network Service Header (NSH) defined in RFC 8300.

This document adopts the service function chaining architecture described in RFC 7665.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9015.

Copyright Notice

Table of Contents

1. Introduction

As described in [RFC7498], the delivery of end-to-end services can require a packet to pass through a series of Service Functions (SFs) -- e.g., WAN and application accelerators, Deep Packet Inspection (DPI) engines, firewalls, TCP optimizers, and server load balancers -- in a specified order; this is termed "service function chaining". There are a number of issues associated with deploying and maintaining service function chaining in production networks, which are described below.

Historically, if a packet needed to travel through a particular service chain, the nodes hosting the service functions of that chain were placed in the network topology in such a way that the packet could not reach its ultimate destination without first passing through all the service functions in the proper order. This need to place the service functions at particular topological locations

limited the ability to adapt a service function chain to changes in network topology (e.g., link or node failures), network utilization, or offered service load. These topological restrictions on where the service functions could be placed raised the following issues:

1. The process of configuring or modifying a service function chain is operationally complex and may require changes to the network topology.
2. Alternate or redundant service functions may need to be co-located with the primary service functions.
3. When there is more than one path between source and destination, forwarding may be asymmetric, and it may be difficult to support bidirectional service function chains using simple routing methodologies and protocols without adding mechanisms for traffic steering or traffic engineering.

In order to address these issues, the service function chaining architecture describes service function chains that are built in their own overlay network (the service function overlay network), coexisting with other overlay networks, over a common underlay network [RFC7665]. A service function chain is a sequence of service functions through which packet flows that satisfy specified criteria will pass.

This document describes the use of BGP as a control plane for networks that support service function chaining. The document introduces a new BGP address family called the "Service Function Chain (SFC) Address Family Identifier / Subsequent Address Family Identifier" (SFC AFI/SAFI) with two Route Types. One Route Type is originated by a node to advertise that it hosts a particular instance of a specified service function. This Route Type also provides "instructions" on how to send a packet to the hosting node in a way that indicates that the service function has to be applied to the packet. The other Route Type is used by a controller (a centralized network component responsible for planning and coordinating service function chaining within the network) to advertise the paths of "chains" of service functions and give a unique designator to each such path so that they can be used in conjunction with the Network Service Header (NSH) [RFC8300].

This document adopts the service function chaining architecture described in [RFC7665].

1.1. Requirements Language

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Terminology

This document uses the following terms from [RFC7665]:

- Bidirectional Service Function Chain
- Classifier
- Service Function (SF)

- Service Function Chain (SFC)
- Service Function Forwarder (SFF)
- Service Function Instance (SFI)
- Service Function Path (SFP)
- SFC branching

Additionally, this document uses the following terms from [RFC8300]:

- Network Service Header (NSH)
- Service Index (SI)
- Service Path Identifier (SPI)

This document introduces the following terms:

Service Function Instance Route (SFIR): A new BGP Route Type advertised by the node that hosts an SFI to describe the SFI and to announce the way to forward a packet to the node through the underlay network.

Service Function Overlay Network: The logical network comprised of classifiers, SFFs, and SFIs that are connected by paths or tunnels through underlay transport networks.

Service Function Path Route (SFPR): A new BGP Route Type originated by controllers to advertise the details of each SFP.

Service Function Type (SFT): An indication of the function and features of an SFI.

2. Overview

This section provides an overview of service function chaining in general and the control plane defined in this document. After reading this section, readers may find it helpful to look through Section 8 for some simple worked examples.

2.1. Overview of Service Function Chaining

In [RFC8300], a Service Function Chain (SFC) is an ordered list of Service Functions (SFs). A Service Function Path (SFP) is an indication of which instances of SFs are acceptable to be traversed in an instantiation of an SFC in a service function overlay network. The Service Path Identifier (SPI) is a 24-bit number that identifies a specific SFP, and a Service Index (SI) is an 8-bit number that identifies a specific point in that path. In the context of a particular SFP (identified by an SPI), an SI represents a particular service function and indicates the order of that SF in the SFP.

Within the context of a specific SFP, an SI references a set of one or more SFs. Each of those SFs may be supported by one or more Service Function Instances (SFIs). Thus, an SI may represent a choice of SFIs of one or more service function types. By deploying multiple SFIs for a single SF, one can provide load balancing and redundancy.

A special functional element, called a "classifier", is located at each ingress point to a service function overlay network. It assigns the packets of a given packet flow to a specific SFP. This may be done by comparing specific fields in a packet's header with local policy, which may be customer/network/service specific. The classifier picks an SFP and sets the SPI accordingly; it then sets the SI to the value of the SI for the first hop in the SFP, and then prepends a Network Service Header (NSH) [RFC8300] containing the assigned SPI/SI to that packet. Note that the classifier and the node that hosts the first SF in an SFP need not be located at the same point in the service function overlay network.

Note that the presence of the NSH can make it difficult for nodes in the underlay network to locate the fields in the original packet that would normally be used to constrain equal-cost multipath (ECMP) forwarding. Therefore, it is recommended that the node prepending the NSH also provide some form of entropy indicator that can be used in the underlay network. How this indicator is generated and supplied, and how an SFF generates a new entropy indicator when it forwards a packet to the next SFF, are out of the scope of this document.

The Service Function Forwarder (SFF) receives a packet from the previous node in an SFP, removes the packet's link layer or tunnel encapsulation, and hands the packet and the NSH to the SFI for processing. The SFI has no knowledge of the SFP.

When the SFF receives the packet and the NSH back from the SFI, it must select the next SFI along the path using the SPI and SI in the NSH and potentially choosing between multiple SFIs (possibly of different SFTs), as described in Section 5. In the normal case, the SPI remains unchanged, and the SI will have been decremented to indicate the next SF along the path. But other possibilities exist if the SF makes other changes to the NSH through a process of reclassification:

- The SI in the NSH may indicate:
  - A previous SF in the path; this is known as "looping" (see Section 6).
  - An SF further down the path; this is known as "jumping" (again see Section 6).
- The SPI and the SI may point to an SF on a different SFP; this is known as "branching" (see Section 6).

Such modifications are limited to within the same service function overlay network. That is, an SPI is known within the scope of service function overlay network. Furthermore, the new SI value is interpreted in the context of the SFP identified by the SPI.

As described in [RFC8300], an SPI that is unknown or not valid is treated as an error, and the SFF drops the packet; such errors should be logged, and such logs are subject to rate limits.

Also, as described in [RFC8300], an SFF receiving an SI that is unknown in the context of the SPI can reduce the value to the next meaningful SI value in the SFP indicated by the SPI. If no such value exists, or if the SFF does not support reducing the SI, the SFF drops the packet and should log the event; such logs are also subject to rate limits.

The SFF then selects an SFI that provides the SF denoted by the SPI/SI and forwards the packet to the SFF that supports that SFI.

[RFC8300] makes it clear that the intended scope is for use within a single provider's operational domain.

This document adopts the service function chaining architecture described in [RFC7665] and adds a control plane to support the functions, as described in Section 2.2. An essential component of this solution is the controller. This is a network component responsible for planning SFPs within the network. It gathers information about the availability of SFIs and SFFs, instructs the control plane about the SFPs to be programmed, and instructs the classifiers how to assign traffic flows to individual SFPs.

2.2. Control Plane Overview

To accomplish the function described in Section 2.1, this document introduces the Service Function Type (SFT), which is the category of SF that is supported by an SFF (such as "firewall"). An IANA registry of service function types is introduced in Section 10.5 and is consistent with types used in other work, such as [BGP-LS-SR]. An SFF may support SFs of multiple different SFTs, and it may support multiple SFIs of each SF.

The registry of SFT values (see Section 10.5) is split into three ranges with assignment policies per [RFC8126]:

- The special-purpose SFT values range is assigned through Standards Action. Values in that range are used for special SFC operations and do not apply to the types of SF that may form part of the SFP.
- The First Come First Served range tracks assignments of SFT values made by any party that defines an SF type. Reference through an Internet-Draft is desirable, but not required.
- The Private Use range is not tracked by IANA and is primarily intended for use in private networks where the meaning of the SFT values is locally tracked and under the control of a local administrator.

It is envisaged that the majority of SFT values used will be assigned from the First Come First Served space in the registry. This will ensure interoperability, especially in situations where software and hardware from different vendors are deployed in the same networks, or when networks are merged. However, operators of private networks may choose to develop their own SFs and manage the configuration and operation of their network through their own list of SFT values.

This document also introduces a new BGP AFI/SAFI (values 31 and 9, respectively) for "SFC Routes". Two SFC Route Types are defined by this document: the Service Function Instance Route (SFIR) and the Service Function Path Route (SFPR). As detailed in Section 3, the Route Type is indicated by a subfield in the Network Layer Reachability Information (NLRI).

- The SFIR is advertised by the node that provides access to the service function instance (i.e., the SFF). The SFIR describes a particular instance of a particular SF (i.e., an SFI) and the way to forward a packet to it through the underlay network, i.e., IP address and encapsulation information.

- The SFPRs are originated by controllers. One SFPR is originated for each SFP. The SFPR specifies:
  A. the SPI of the path,
  B. the sequence of SFTs and/or SFIs of which the path consists, and
  C. for each such SFT or SFI, the SI that represents it in the identified path.

This approach assumes that there is an underlay network that provides connectivity between SFFs and controllers and that the SFFs are grouped to form one or more service function overlay networks through which SFPs are built. We assume that the controllers have BGP connectivity to all SFFs and all classifiers within each service function overlay network.

When choosing the next SFI in a path, the SFF uses the SPI and SI as well as the SFT to choose among the SFIs, applying, for example, a load-balancing algorithm or direct knowledge of the underlay network topology, as described in Section 4.

The SFF then encapsulates the packet using the encapsulation specified by the SFIR of the selected SFI and forwards the packet. See Figure 1.

Thus, the SFF can be seen as a portal in the underlay network through which a particular SFI is reached.

Figure 1 shows a reference model for the service function chaining architecture. There are four SFFs (SFF-1 through SFF-4) connected by tunnels across the underlay network. Packets arrive at a classifier and are channeled along SFPs to destinations reachable through SFF-4.

SFF-1 and SFF-4 each have one instance of one SF attached (SFa and SFe). SFF-2 has two types of SF attached: one instance of one (SFc) and three instances of the other (SFb). SFF-3 has just one instance of an SF (SFd), but in this case, the type of SFd is the same type as SFb (SFTx).

This figure demonstrates how load balancing can be achieved by creating several SFPs that satisfy the same SFC. Suppose an SFC needs to include SFa, an SF of type SFTx, and SFc. A number of SFPs can be constructed using any instance of SFb or using SFd. Load balancing may be applied at two places:

- The classifier may distribute different flows onto different SFPs to share the load in the network and across SFIs.
- SFF-2 may distribute different flows (on the same SFP) to different instances of SFb to share the processing load.

Note that, for convenience and clarity, Figure 1 shows only a few tunnels between SFFs. There could be a full mesh of such tunnels, or more likely, a selection of tunnels connecting key SFFs to enable the construction of SFPs and balance load and traffic in the network. Further, the figure does not show any controllers; these would each have BGP connectivity to the classifier and all of the SFFs.

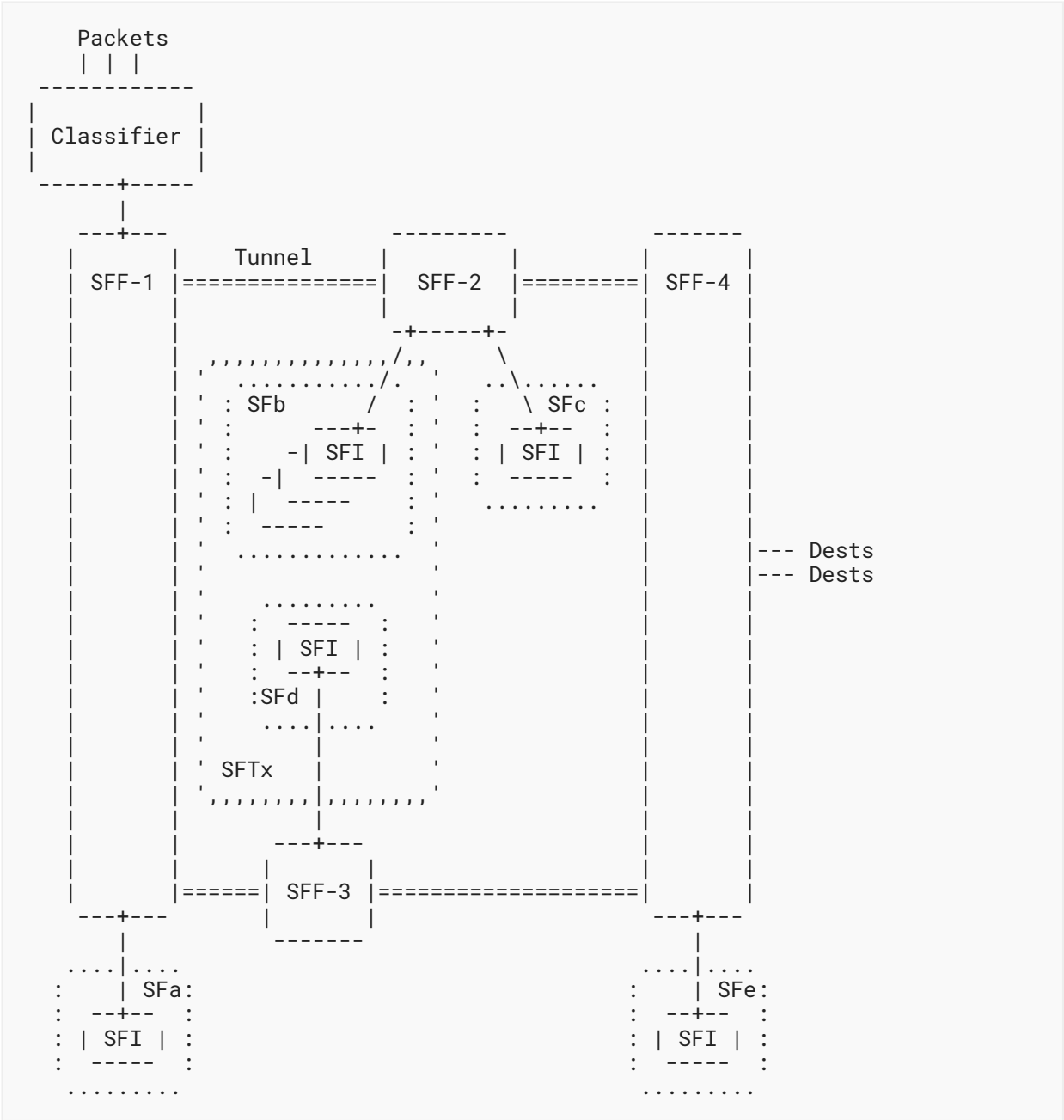

Figure 1: The Service Function Chaining Architecture Reference Model

As previously noted, [RFC8300] makes it clear that the mechanisms it defines are intended for use within a single provider's operational domain. This reduces the requirements on the control plane function.

Section 5.2 of [RFC7665] sets out the functions provided by a control plane for a service function chaining network. The functions are broken down into six items, the first four of which are completely covered by the mechanisms described in this document:

1. Visibility of all SFs and the SFFs through which they are reached.
2. Computation of SFPs and programming into the network.
3. Selection of SFIs explicitly in the SFP or dynamically within the network.
4. Programming of SFFs with forwarding path information.

The fifth and sixth items in the list in RFC 7665 concern the use of metadata. These are more peripheral to the control plane mechanisms defined in this document but are discussed in Section 4.4.

3. BGP SFC Routes

This document defines a new AFI/SAFI for BGP, known as "SFC", with an NLRI that is described in this section.

The format of the SFC NLRI is shown in Figure 2.

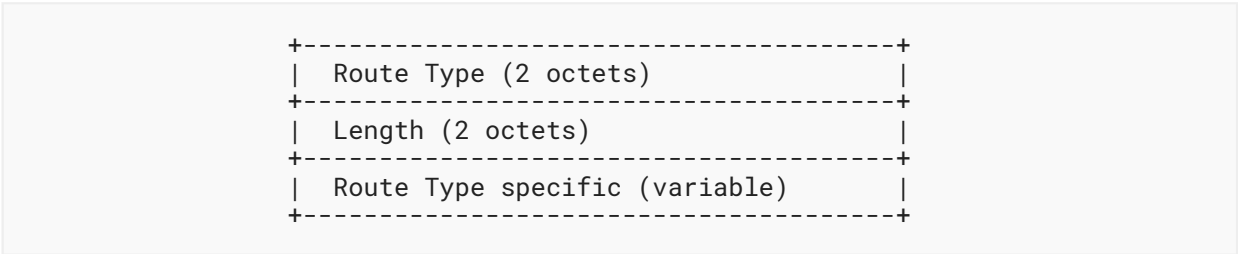

Figure 2: The Format of the SFC NLRI

The "Route Type" field determines the encoding of the rest of the Route Type specific SFC NLRI.

The "Length" field indicates the length, in octets, of the "Route Type specific" field of the SFC NLRI.

This document defines the following Route Types:

1. Service Function Instance Route (SFIR)
2. Service Function Path Route (SFPR)

An SFIR is used to identify an SFI. An SFPR defines a sequence of SFs (each of which has at least one instance advertised in an SFIR) that form an SFP.

The detailed encoding and procedures for these Route Types are described in subsequent sections.

The SFC NLRI is carried in BGP [RFC4271] using BGP Multiprotocol Extensions [RFC4760] with an Address Family Identifier (AFI) of 31 and a Subsequent Address Family Identifier (SAFI) of 9. The "NLRI" field in the MP_REACH_NLRI/MP_UNREACH_NLRI attribute contains the SFC NLRI, encoded as specified above.

In order for two BGP speakers to exchange SFC NLRIs, they **MUST** use BGP capabilities advertisements to ensure that they both are capable of properly processing such NLRIs. This is done as specified in [RFC4760], by using capability code 1 (Multiprotocol BGP) with an AFI of 31 and a SAFI of 9.

The "nexthop" field of the MP_REACH_NLRI attribute of the SFC NLRI **MUST** be set to a loopback address of the advertising SFF.

3.1. Service Function Instance Route (SFIR)

Figure 3 shows the Route Type specific NLRI of the SFIR.

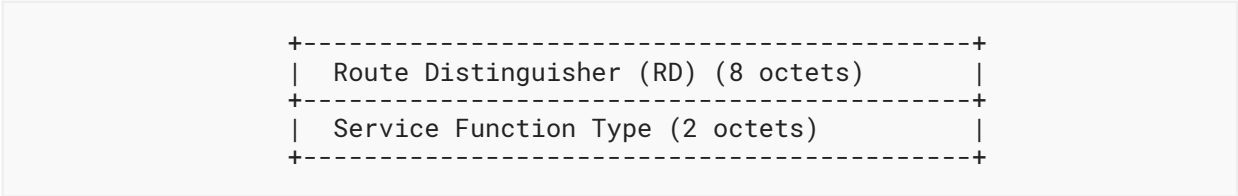

Figure 3: SFIR Route Type Specific NLRI

[RFC4364] defines a Route Distinguisher (RD) as consisting of a two-byte "Type" field and a six-byte "Value" field, and it defines RD types 0, 1, and 2. In this specification, the RD (used for the SFIR) **MUST** be of type 0, 1, or 2.

If two SFIRs are originated from different administrative domains (within the same provider's operational domain), they **MUST** have different RDs. In particular, SFIRs from different VPNs (for different service function overlay networks) **MUST** have different RDs, and those RDs **MUST** be different from any non-VPN SFIRs.

The SFT identifies the functions/features an SF can offer, e.g., classifier, firewall, load balancer. There may be several SFIs that can perform a given service function. Each node hosting an SFI **MUST** originate an SFIR for each type of SF that it hosts (as indicated by the SFT value), and it **MAY** advertise an SFIR for each instance of each type of SF. A minimal advertisement allows construction of valid SFPs and leaves the selection of SFIs to the local SFF; a detailed advertisement may have scaling concerns but allows a controller that constructs an SFP to make an explicit choice of SFI.

Note that a node may advertise all its SFIs of one SFT in one shot using normal BGP UPDATE packing. That is, all of the SFIRs in an Update share a common Tunnel Encapsulation and Route Target (RT) attribute. See also Section 3.2.1.

The SFIR representing a given SFI will contain an NLRI with "RD" field set to an RD as specified above, and with the "SFT" field set to identify that SFI's SFT. The values for the "SFT" field are taken from a registry administered by IANA (see Section 10). A BGP UPDATE containing one or more SFIRs **MUST** also include a tunnel encapsulation attribute [RFC9012]. If a data packet needs to be sent to an SFI identified in one of the SFIRs, it will be encapsulated as specified by the tunnel encapsulation attribute and then transmitted through the underlay network.

Note that the tunnel encapsulation attribute **MUST** contain sufficient information to allow the advertising SFF to identify the overlay or VPN network that a received packet is transiting. This is because the [SPI, SI] in a received packet is specific to a particular overlay or VPN network.

3.1.1. SFIR Pool Identifier Extended Community

This document defines a new transitive Extended Community [RFC4360] of type 0x0b called the "SFC Extended Community". When used with Sub-Type 1, this is called the "SFIR Pool Identifier extended community". It **MAY** be included in SFIR advertisements, and it is used to indicate the identity of a pool of SFIRs to which an SFIR belongs. Since an SFIR may be a member of more than one pool, multiple of these extended communities may be present on a single SFIR advertisement.

SFIR pools allow SFIRs to be grouped for any purpose. Possible uses include control plane scalability and stability. A pool identifier may be included in an SFPR to indicate a set of SFIs that are acceptable at a specific point on an SFP (see Sections 3.2.1.3 and 4.3).

The SFIR Pool Identifier Extended Community is encoded in 8 octets as shown in Figure 4.

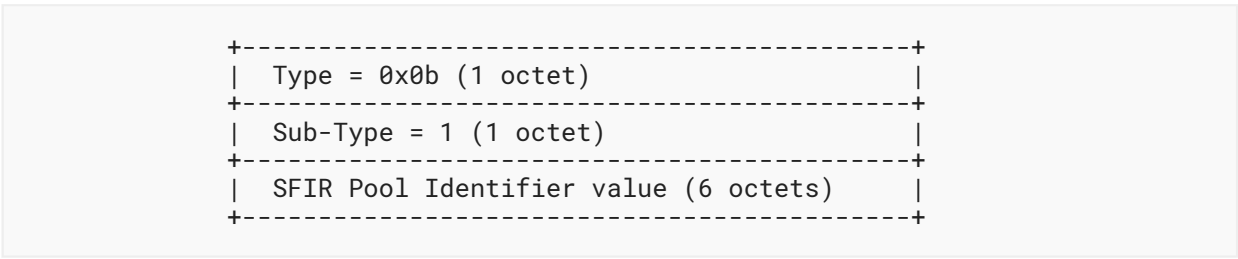

Figure 4: The SFIR Pool Identifier Extended Community

The SFIR Pool Identifier value is encoded in a 6-octet field in network byte order, and the value is unique within the scope of an overlay network. This means that pool identifiers need to be centrally managed, which is consistent with the assignment of SFIs to pools.

3.1.2. MPLS Mixed Swapping/Stacking Extended Community

As noted in Section 3.1.1, this document defines a new transitive Extended Community of type 0x0b called the "SFC Extended Community". When used with Sub-Type 2, this is called the "MPLS Mixed Swapping/Stacking Labels Extended Community". The community is encoded as shown in Figure 5. It contains a pair of MPLS labels: an SFC Context Label and an SF Label, as described in [RFC8595]. Each label is 20 bits encoded in a 3-octet (24-bit) field with 4 trailing bits that **MUST** be set to zero.

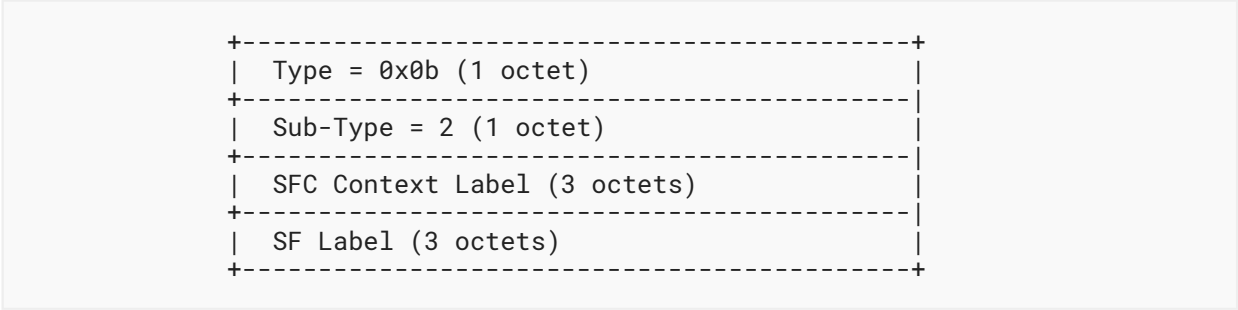

Figure 5: The MPLS Mixed Swapping/Stacking Labels Extended Community

Note that it is assumed that each SFF has one or more globally unique SFC Context Labels and that the context-label space and the SPI-address space are disjoint. In other words, a label value cannot be used to indicate both an SFC context and an SPI, and it can be determined from knowledge of the label spaces whether a label indicates an SFC context or an SPI.

If an SFF supports SFP Traversal with an MPLS Label Stack, it **MUST** include this Extended Community with the SFIRs that it advertises.

See Section 7.6 for a description of how this Extended Community is used.

3.2. Service Function Path Route (SFPR)

Figure 6 shows the Route Type specific NLRI of the SFPR.

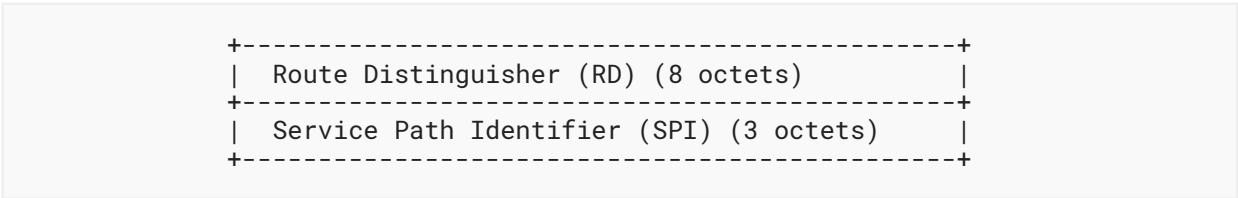

Figure 6: SFPR Route Type Specific NLRI

[RFC4364] defines a Route Distinguisher (RD) as consisting of a two-byte "Type" field and a six-byte "Value" field, and it defines RD types 0, 1, and 2. In this specification, the RD (used for the SFPR) **MUST** be of type 0, 1, or 2.

All SFPs **MUST** be associated with an RD. The association of an SFP with an RD is determined by provisioning. If two SFPRs are originated from different controllers, they **MUST** have different RDs. Additionally, SFPRs from different VPNs (i.e., in different service function overlay networks) **MUST** have different RDs, and those RDs **MUST** be different from any non-VPN SFPRs.

The Service path identifier is defined in [RFC8300] and is the value to be placed in the "Service Path Identifier" field of the NSH of any packet sent on this SFP. It is expected that one or more controllers will originate these routes in order to configure a service function overlay network.

The SFP is described in a new BGP Path attribute, the SFP attribute. Section 3.2.1 shows the format of that attribute.

3.2.1. The SFP Attribute

[RFC4271] defines BGP Path attributes. This document introduces a new Optional Transitive Path attribute called the "SFP attribute", with value 37. The first SFP attribute **MUST** be processed, and subsequent instances **MUST** be ignored.

The common fields of the SFP attribute are set as follows:

- The Optional bit is set to 1 to indicate that this is an optional attribute.
- The Transitive bit is set to 1 to indicate that this is a transitive attribute.
- The Extended Length bit is set if the length of the SFP attribute is encoded in one octet (set to 0) or two octets (set to 1), as described in [RFC4271].
- The Attribute Type Code is set to 37.

The content of the SFP attribute is a series of Type-Length-Value (TLV) constructs. Some TLVs may include Sub-TLVs. All TLVs and Sub-TLVs have a common format:

Type: A single octet indicating the type of the SFP attribute TLV. Values are taken from the registry described in Section 10.3.

Length: A two-octet field indicating the length of the data following the "Length" field, counted in octets.

Value: The contents of the TLV.

The formats of the TLVs defined in this document are shown in the following sections. The presence rules and meanings are as follows.

- The SFP attribute contains a sequence of zero or more Association TLVs. That is, the Association TLV is **OPTIONAL**. Each Association TLV provides an association between this SFPR and another SFPR. Each associated SFPR is indicated using the RD with which it is advertised (we say the SFPR-RD to avoid ambiguity).
- The SFP attribute contains a sequence of one or more Hop TLVs. Each Hop TLV contains all of the information about a single hop in the SFP.
- Each Hop TLV contains an SI value and a sequence of one or more SFT TLVs. Each SFT TLV contains an SFI reference for each instance of an SF that is allowed at this hop of the SFP for the specific SFT. Each SFI is indicated using the RD with which it is advertised (we say the SFIR-RD to avoid ambiguity).

Section 6 of [RFC4271] describes the handling of malformed BGP attributes, or those that are in error in some way. [RFC7606] revises BGP error handling specifically for the UPDATE message, provides guidelines for the authors of documents defining new attributes, and revises the error-handling procedures for a number of existing attributes. This document introduces the SFP attribute and so defines error handling as follows:

- When parsing a message, an unknown Attribute Type Code or a length that suggests that the attribute is longer than the remaining message is treated as a malformed message, and the "treat-as-withdraw" approach is used as per [RFC7606].
- When parsing a message that contains an SFP attribute, the following cases constitute errors:
  1. Optional bit is set to 0 in the SFP attribute.
  2. Transitive bit is set to 0 in the SFP attribute.
  3. Unknown "TLV Type" field found in the SFP attribute.
  4. TLV length that suggests the TLV extends beyond the end of the SFP attribute.
  5. Association TLV contains an unknown SFPR-RD.
  6. No Hop TLV found in the SFP attribute.
  7. No Sub-TLV found in a Hop TLV.
  8. Unknown SFIR-RD found in an SFT TLV.

- The errors listed above are treated as follows:

  1, 2, 4, 6, 7: The attribute **MUST** be treated as malformed and the "treat-as-withdraw" approach used as per [RFC7606].

  3: Unknown TLVs **MUST** be ignored, and message processing **MUST** continue.

  5, 8: The absence of an RD with which to correlate is nothing more than a soft error. The receiver **SHOULD** store the information from the SFP attribute until a corresponding advertisement is received.

3.2.1.1. The Association TLV

The Association TLV is an optional TLV in the SFP attribute. It **MAY** be present multiple times. Each occurrence provides an association with another SFP as advertised in another SFPR. The format of the Association TLV is shown in Figure 7.

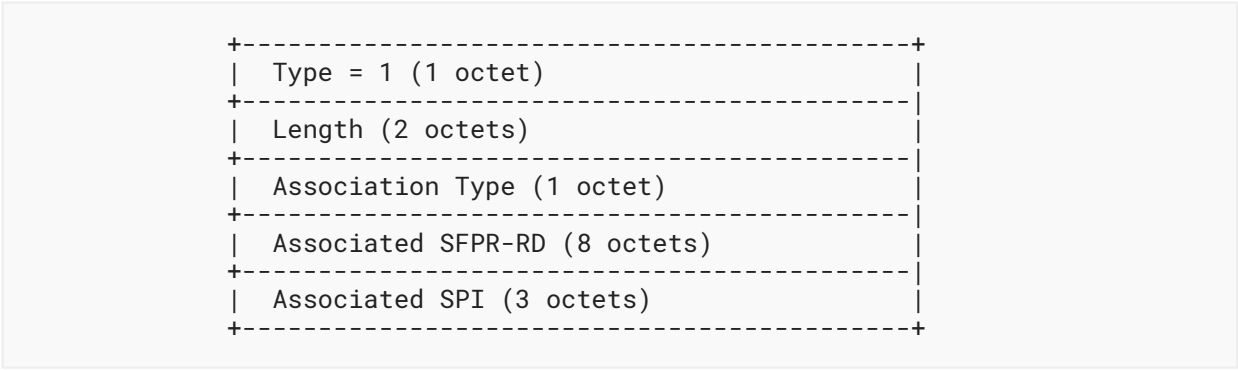

Figure 7: The Format of the Association TLV

The fields are as follows:

- "Type" is set to 1 to indicate an Association TLV.
- "Length" indicates the length in octets of the "Association Type" and "Associated SFPR-RD" fields. The value of the "Length" field is 12.
- The "Association Type" field indicates the type of association. The values are tracked in an IANA registry (see Section 10.4). Only one value is defined in this document: Type 1 indicates association of two unidirectional SFPs to form a bidirectional SFP. An SFP attribute **SHOULD NOT** contain more than one Association TLV with Association Type 1; if more than one is present, the first one **MUST** be processed, and subsequent instances **MUST** be ignored. Note that documents that define new association types must also define the presence rules for Association TLVs of the new type.
- The Associated SFPR-RD contains the RD of the associated SFP as advertised in an SFPR.
- The Associated SPI contains the SPI of the associated SFP as advertised in an SFPR.

Association TLVs with unknown Association Type values **SHOULD** be ignored. Association TLVs that contain an Associated SFPR-RD value equal to the RD of the SFPR in which they are contained **SHOULD** be ignored. If the Associated SPI is not equal to the SPI advertised in the SFPR indicated by the Associated SFPR-RD, then the Association TLV **SHOULD** be ignored. In all three of these cases, an implementation **MAY** reject the SFP attribute as malformed and use the "treat-as-withdraw" approach per [RFC7606]; however, implementors are cautioned that such an approach may make an implementation less flexible in the event of future extensions to this protocol.

Note that when two SFPRs reference each other using the Association TLV, one SFPR advertisement will be received before the other. Therefore, processing of an association **MUST NOT** be rejected simply because the Associated SFPR-RD is unknown.

Further discussion of correlation of SFPRs is provided in Section 7.1.

3.2.1.2. The Hop TLV

There is one Hop TLV in the SFP attribute for each hop in the SFP. The format of the Hop TLV is shown in Figure 8. At least one Hop TLV **MUST** be present in an SFP attribute.

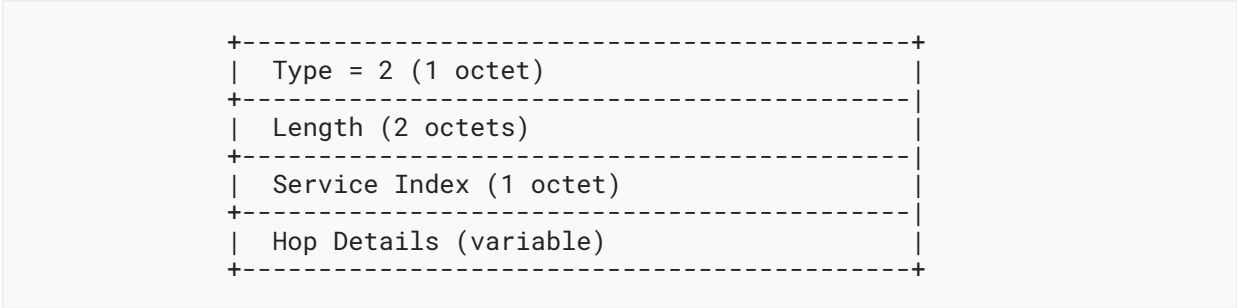

Figure 8: The Format of the Hop TLV

The fields are as follows:

- "Type" is set to 2 to indicate a Hop TLV.
- "Length" indicates the length, in octets, of the "Service Index" and "Hop Details" fields.
- The Service Index is defined in [RFC8300] and is the value found in the "Service Index" field of the NSH that an SFF will use to look up to which next SFI a packet is to be sent.
- The "Hop Details" field consists of a sequence of one or more Sub-TLVs.

Each hop of the SFP may demand that a specific type of SF is executed, and that type is indicated in Sub-TLVs of the Hop TLV. At least one Sub-TLV **MUST** be present. This document defines the SFT Sub-TLV (see Section 3.2.1.3) and the MPLS Swapping/Stacking Sub-TLV (see Section 3.2.1.4); other Sub-TLVs may be defined in future. The SFT Sub-TLV provides a list of which types of SF are acceptable at a specific hop, and for each type it allows a degree of control to be imposed on the choice of SFIs of that particular type. The MPLS Swapping/Stacking Sub-TLV indicates the type of SFC encoding to use in an MPLS label stack.

If no Hop TLV is present in an SFP attribute, it is a malformed attribute.

3.2.1.3. The SFT Sub-TLV

The SFT Sub-TLV **MAY** be included in the list of Sub-TLVs of the Hop TLV. The format of the SFT Sub-TLV is shown in Figure 9. The Hop Sub-TLV contains a list of SFIR-RD values each taken from the advertisement of an SFI. Together they form a list of acceptable SFIs of the indicated type.

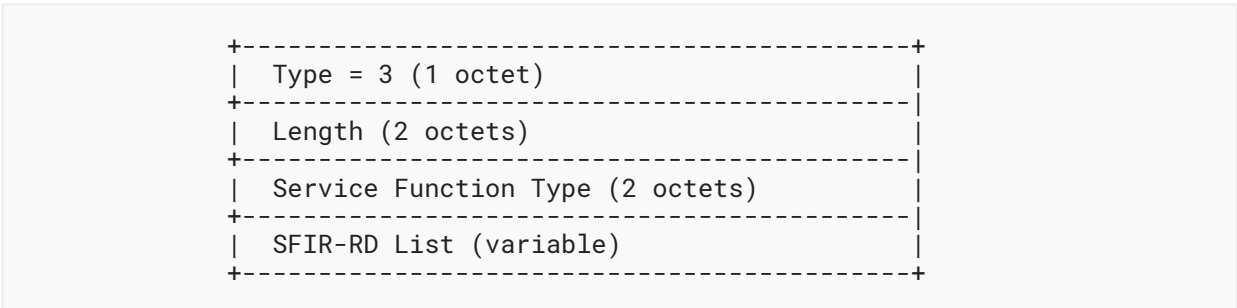

Figure 9: The Format of the SFT Sub-TLV

The fields are as follows:

- "Type" is set to 3 to indicate an SFT Sub-TLV.
- "Length" indicates the length, in octets, of the "Service Function Type" and "SFIR-RD List" fields.
- The SFT value indicates the category (type) of SF that is to be executed at this hop. The types are as advertised for the SFs supported by the SFFs. SFT values in the range 1-31 are special-purpose SFT values and have meanings defined by the documents that describe them -- the value "Change Sequence" is defined in Section 6.1 of this document.
- The hop description is further qualified beyond the specification of the SFTs by listing, for each SFT in each hop, the SFIs that may be used at the hop. The SFIs are identified using the SFIR-RDs from the advertisements of the SFIs in the SFIRs. Note that if the list contains one or more SFIR Pool Identifiers, then for each, the SFIR-RD list is effectively expanded to include the SFIR-RD of each SFIR advertised with that SFIR Pool Identifier. An SFIR-RD of value zero has special meaning, as described in Section 5. Each entry in the list is eight octets long, and the number of entries in the list can be deduced from the value of the "Length" field.
- Note that an SFIR-RD is of type 0, 1, or 2 (as described in Section 3.1). Thus, the high-order octet of an RD found in an SFIR-RD List always has a value of 0x00. However, the high-order octet of an SFIR Pool Identifier (an Extended Community with "Type" field 0x0b) will always have a nonzero value. This allows the node processing the SFIR-RD list to distinguish between the two types of list entry.

3.2.1.4. MPLS Swapping/Stacking Sub-TLV

The MPLS Swapping/Stacking Sub-TLV (Type value 4) is a zero-length Sub-TLV that is **OPTIONAL** in the Hop TLV and is used when the data representation is MPLS (see Section 7.5). When present, it indicates to the classifier imposing an MPLS label stack that the current hop is to use an {SFC Context Label, SF label} rather than an {SPI, SF} label pair. See Section 7.6 for more details.

3.2.1.5. SFP Traversal With MPLS Label Stack TLV

The SFP Traversal With MPLS Label Stack TLV (Type value 5) is a zero-length TLV that can be carried in the SFP attribute and indicates to the classifier and the SFFs on the SFP that an MPLS label stack with label swapping/stacking is to be used for packets traversing the SFP. All of the SFFs specified at each of the SFP's hops **MUST** have advertised an MPLS Mixed Swapping/Stacking Extended Community (see Section 3.1.2) for the SFP to be considered usable.

3.2.2. General Rules for the SFP Attribute

It is possible for the same SFI, as described by an SFIR, to be used in multiple SFPRs.

When two SFPRs have the same SPI but different SFPR-RDs, there can be three cases:

1. Two or more controllers are originating SFPRs for the same SFP. In this case, the content of the SFPRs is identical, and the duplication is to ensure receipt and provide controller redundancy.

2. There is a transition in content of the advertised SFP, and the advertisements may originate from one or more controllers. In this case, the content of the SFPRs will be different.
3. The reuse of an SPI may result from a configuration error.

There is no way in any of these cases for the receiving SFF to know which SFPR to process, and the SFPRs could be received in any order. At any point in time, when multiple SFPRs have the same SPI but different SFPR-RDs, the SFF **MUST** use the SFPR with the numerically lowest SFPR-RD when interpreting the RDs as 8-octet integers in network byte order. The SFF **SHOULD** log this occurrence to assist with debugging.

Furthermore, a controller that wants to change the content of an SFP is **RECOMMENDED** to use a new SPI and so create a new SFP onto which the classifiers can transition packet flows before the SFPR for the old SFP is withdrawn. This avoids any race conditions with SFPR advertisements.

Additionally, a controller **SHOULD NOT** reuse an SPI after it has withdrawn the SFPR that used it until at least a configurable amount of time has passed. This timer **SHOULD** have a default of one hour.

4. Mode of Operation

This document describes the use of BGP as a control plane to create and manage a service function overlay network.

4.1. Route Targets

The main feature introduced by this document is the ability to create multiple service function overlay networks through the use of Route Targets (RTs) [RFC4364].

Every BGP UPDATE containing an SFIR or SFPR carries one or more RTs. The RT carried by a particular SFIR or SFPR is determined by the provisioning of the route's originator.

Every node in a service function overlay network is configured with one or more import RTs. Thus, each SFF will import only the SFPRs with matching RTs, allowing the construction of multiple service function overlay networks or the instantiation of SFCs within a Layer 3 Virtual Private Network (L3VPN) or Ethernet VPN (EVPN) instance (see Section 7.3). An SFF that has a presence in multiple service function overlay networks (i.e., one that imports more than one RT) will usually maintain separate forwarding state for each overlay network.

4.2. Service Function Instance Routes

The SFIR (see Section 3.1) is used to advertise the existence and location of a specific SFI; it consists of:

- The RT as just described.
- A Service Function Type (SFT) that is the type of service function that is provided (such as "firewall").
- A Route Distinguisher (RD) that is unique to a specific overlay.

4.3. Service Function Path Routes

The SFPR (see Section 3.2) describes a specific path of an SFC. The SFPR contains the Service Path Identifier (SPI) used to identify the SFP in the NSH in the data plane. It also contains a sequence of Service Indexes (SIs). Each SI identifies a hop in the SFP, and each hop is a choice between one or more SFIs.

As described in this document, each SFP route is identified in the service function overlay network by an RD and an SPI. The SPI is unique within a single VPN instance supported by the underlay network.

The SFPR advertisement comprises:

- An RT as described in Section 4.1.
- A tuple that identifies the SFPR.
  - An RD that identifies an advertisement of an SFPR.
  - The SPI that uniquely identifies this path within the VPN instance distinguished by the RD. This SPI also appears in the NSH.
- A series of SIs. Each SI is used in the context of a particular SPI and identifies one or more SFs (distinguished by their SFTs). For each SF, it identifies a set of SFIs that instantiate the SF. The values of the SI indicate the order in which the SFs are to be executed in the SFP that is represented by the SPI.
- The SI is used in the NSH to identify the entries in the SFP. Note that the SI values have meaning only relative to a specific path. They have no semantic other than to indicate the order of SFs within the path and are assumed to be monotonically decreasing from the start to the end of the path [RFC8300].
- Each SI is associated with a set of one or more SFIs that can be used to provide the indexed SF within the path. Each member of the set comprises:
  - The RD used in an SFIR advertisement of the SFI.
  - The SFT that indicates the type of function as used in the same SFIR advertisement of the SFI.

This may be summarized as follows, where the notations "SFPR-RD" and "SFIR-RD" are used to distinguish the two different RDs, and where "*" indicates a multiplier:

```
RT, {SFPR-RD, SPI}, m * {SI, {n * {SFT, p * SFIR-RD} } }
```

Where:

RT: Route Target

SFPR-RD: The Route Descriptor of the SFPR advertisement

SPI: Service Path Identifier used in the NSH

m: The number of hops in the SFP

n: The number of choices of SFT for a specific hop

p: The number of choices of SFI for a given SFT in a specific hop

SI: Service Index used in the NSH to indicate a specific hop

SFT: The Service Function Type used in the same advertisement of the SFIR

SFIR-RD: The Route Descriptor used in an advertisement of the SFIR

That is, there can be multiple SFTs at a given hop, as described in Section 5.

Note that the values of SI are from the set {255, ..., 1} and are monotonically decreasing within the SFP. SIs **MUST** appear in order within the SFPR (i.e., monotonically decreasing) and **MUST NOT** appear more than once. Gaps **MAY** appear in the sequence, as described in Section 4.5.1. Malformed SFPRs **MUST** be discarded and **MUST** cause any previous instance of the SFPR (same SFPR-RD and SPI) to be discarded.

Note that if the SFIR-RD list in an SFT TLV contains one or more SFIR Pool Identifiers, then in the above expression, "p" is the sum of the number of individual SFIR-RD values and the sum for each SFIR Pool Identifier of the number of SFIRs advertised with that SFIR Pool Identifier. In other words, the list of SFIR-RD values is effectively expanded to include the SFIR-RD of each SFIR advertised with each SFIR Pool Identifier in the SFIR-RD list.

The choice of SFI is explained further in Section 5. Note that an SFIR-RD value of zero has special meaning, as described in that section.

4.4. Classifier Operation

As shown in Figure 1, the classifier is a component that is used to assign packets to an SFP.

The classifier is responsible for determining to which packet flow a packet belongs. The mechanism it uses to achieve that classification is out of the scope of this document but might include inspection of the packet header. The classifier has been instructed (by the controller or through some other configuration mechanism -- see Section 7.4) which flows are to be assigned to which SFPs, and so it can impose an NSH on each packet and initialize the NSH with the SPI of the selected SFP and the SI of its first hop.

Note that instructions delivered to the classifier may include information about the metadata to encode (and the format for that encoding) on packets that are classified by the classifier to a particular SFP. As mentioned in Section 2.2, this corresponds to the fifth element of control plane functionality described in [RFC7665]. Such instructions fall outside the scope of this specification (but see Section 7.4), as do instructions to other service function chaining elements on how to interpret metadata (as described in the sixth element of control plane functionality described in [RFC7665]).

4.5. Service Function Forwarder Operation

Each packet sent to an SFF is transmitted encapsulated in an NSH. The NSH includes an SPI and SI: the SPI indicates the SFPR advertisement that announced the SFP; the tuple SPI/SI indicates a specific hop in a specific path and maps to the RD/SFT of a particular SFIR advertisement.

When an SFF gets an SFPR advertisement, it will first determine whether to import the route by examining the RT. If the SFPR is imported, the SFF then determines whether it is on the SFP by looking for its own SFIR-RDs or any SFIR-RD with value zero in the SFPR. For each occurrence in the SFP, the SFF creates forwarding state for incoming packets and forwarding state for outgoing packets that have been processed by the specified SFI.

The SFF creates local forwarding state for packets that it receives from other SFFs. This state makes the association between the SPI/SI in the NSH of the received packet and one or more specific local SFIs, as identified by the SFIR-RD/SFT. If there are multiple local SFIs that match, this is because a single advertisement was made for a set of equivalent SFIs, and the SFF may use local policy (such as load balancing) to determine to which SFI to forward a received packet.

The SFF also creates next-hop forwarding state for packets received back from the local SFI that need to be forwarded to the next hop in the SFP. There may be a choice of next hops, as described in Section 4.3. The SFF could install forwarding state for all potential next hops or it could choose to only install forwarding state for a subset of the potential next hops. If a choice is made, then it will be as described in Section 5.

The installed forwarding state may change over time, reacting to changes in the underlay network and the availability of particular SFIs. Note that the forwarding state describes how one SFF sends packets to another SFF, but not how those packets are routed through the underlay network. SFFs may be connected by tunnels across the underlay, or packets may be sent addressed to the next SFF and routed through the underlay. In any case, transmission across the underlay requires encapsulation of packets with a header for transport in the underlay network.

Note that SFFs only create and store forwarding state for the SFPs on which they are included. They do not retain state for all SFPs advertised.

An SFF may also install forwarding state to support looping, jumping, and branching. The protocol mechanism for explicit control of looping, jumping, and branching uses a specific reserved SFT value at a given hop of an SFPR and is described in Section 6.1.

4.5.1. Processing with "Gaps" in the SI Sequence

The behavior of an SF, as described in [RFC8300], is to decrement the value of the "SI" field in the NSH by one before returning a packet to the local SFF for further processing. This means that there is a good reason to assume that the SFP is composed of a series of SFs, each indicated by an SI value one less than the previous.

However, there is an advantage to having nonsuccessive SIs in an SPI. Consider the case where an SPI needs to be modified by the insertion or removal of an SF. In the latter case, this would lead to a "gap" in the sequence of SIs, and in the former case, this could only be achieved if a gap already existed into which the new SF with its new SI value could be inserted. Otherwise, all "downstream" SFs would need to be renumbered.

Now, of course, such renumbering could be performed, but it would lead to a significant disruption to the SFC as all the SFFs along the SFP were "reprogrammed". Thus, to achieve dynamic modification of an SFP (and even in-service modification), it is desirable to be able to make these modifications without changing the SIs of the elements that were present before the modification. This will produce much more consistent/predictable behavior during the convergence period, where otherwise the change would need to be fully propagated.

Another approach says that any change to an SFP simply creates a new SFP that can be assigned a new SPI. All that would be needed would be to give a new instruction to the classifier, and traffic would be switched to the new SFP that contains the new set of SFs. This approach is practical but neglects to consider that the SFP may be referenced by other SFPs (through "branch" instructions) and used by many classifiers. In those cases, the corresponding configuration resulting from a change in SPI may have wide ripples and create scope for errors that are hard to trace.

Therefore, while this document requires that the SI values in an SFP are monotonically decreasing, it makes no assumption that the SI values are sequential. Configuration tools may apply that rule, but they are not required to. To support this, an SFF **SHOULD** process as follows when it receives a packet:

- If the SI indicates a known entry in the SFP, the SFF **MUST** process the packet as normal, looking up the SI and determining to which local SFI to deliver the packet.
- If the SI does not match an entry in the SFP, the SFF **MUST** reduce the SI value to the next (smaller) value present in the SFP and process the packet using that SI.
- If there is no smaller SI (i.e., if the end of the SFP has been reached), the SFF **MUST** treat the SI value as not valid, as described in [RFC8300].

This makes the behavior described in this document a superset of the function in [RFC8300]. That is, an implementation that strictly follows RFC 8300 in performing SI decrements in units of one is perfectly in line with the mechanisms defined in this document.

SFF implementations **MAY** choose to only support contiguous SI values in an SFP. Such an implementation will not support receiving an SI value that is not present in the SFP and will discard the packets as described in [RFC8300].

5. Selection within Service Function Paths

As described in Section 2, the SPI/SI in the NSH passed back from an SFI to the SFF may leave the SFF with a choice of next-hop SFTs and a choice of SFIs for each SFT. That is, the SPI indicates an SFPR, and the SI indicates an entry in that SFPR. Each entry in an SFPR is a set of one or more SFT/SFIR-RD pairs. The SFF **MUST** choose one of these, identify the SFF that supports the chosen SFI, and send the packet to that next-hop SFF.

The choice be may offered for load balancing across multiple SFIs, or for discrimination between different actions necessary at a specific hop in the SFP. Different SFT values may exist at a given hop in an SFP to support several cases:

- There may be multiple instances of similar service functions that are distinguished by different SFT values. For example, firewalls made by vendor A and vendor B may need to be identified by different SFT values because, while they have similar functionality, their behavior is not identical. Then, some SFPs may limit the choice of SF at a given hop by specifying the SFT for vendor A, but other SFPs might not need to control which vendor's SF is used and so can indicate that either SFT can be used.
- There may be an obvious branch needed in an SFP, such as the processing after a firewall where admitted packets continue along the SFP, but suspect packets are diverted to a "penalty box". In this case, the next hop in the SFP will be indicated with two different SFT values.

In the typical case, the SFF chooses a next-hop SFF by looking at the set of all SFFs that support the SFs identified by the SI (that set having been advertised in individual SFIR advertisements), finding the one or more that are "nearest" in the underlay network, and choosing between next-hop SFFs using its own load-balancing algorithm.

An SFI may influence this choice process by passing additional information back, along with the packet and NSH. This information may influence local policy at the SFF to either cause it to favor a next-hop SFF (perhaps selecting one that is not nearest in the underlay) or influence the load-balancing algorithm.

This selection applies to the normal case but also applies in the case of looping, jumping, and branching (see Section 6).

Suppose an SFF in a particular service function overlay network (identified by a particular import RT, RT-z) needs to forward an NSH-encapsulated packet whose SPI is SPI-x and whose SI is SI-y. It does the following:

1. It looks for an installed SFPR that carries RT-z and has SPI-x in its NLRI. If there is none, then such packets cannot be forwarded.
2. From the SFP attribute of that SFPR, it finds the Hop TLV with SI value set to SI-y. If there is no such Hop TLV, then such packets cannot be forwarded.

3. It then finds the "relevant" set of SFIRs by going through the list of SFT TLVs contained in the Hop TLV as follows:
   A. An SFIR is relevant if it carries RT-z, the SFT in its NLRI matches the SFT value in one of the SFT TLVs, and the RD value in its NLRI matches an entry in the list of SFIR-RDs in that SFT TLV.
   B. If an entry in the SFIR-RD list of an SFT TLV contains the value zero, then an SFIR is relevant if it carries RT-z and the SFT in its NLRI matches the SFT value in that SFT TLV. That is, any SFIR in the service function overlay network defined by RT-z and with the correct SFT is relevant.
   C. If a pool identifier is in use, then an SFIR is relevant if it is a member of the pool.

Each of the relevant SFIRs identifies a single SFI and contains a tunnel encapsulation attribute that specifies how to send a packet to that SFI. For a particular packet, the SFF chooses a particular SFI from the set of relevant SFIRs. This choice is made according to local policy.

A typical policy might be to figure out the set of SFIs that are closest and load balance among them. But this is not the only possible policy.

Thus, at any point in time when an SFF selects its next hop, it chooses from the intersection of the set of next-hop RDs contained in the SFPR and the RDs contained in the SFF's local set of SFIRs (i.e., according to the determination of "relevance", above). If the intersection is null, the SFPR is unusable. Similarly, when this condition applies on the controller that originated the SFPR, it **SHOULD** either withdraw the SFPR or re-advertise it with a new set of RDs for the affected hop.

6. Looping, Jumping, and Branching

As described in Section 2, an SFI or an SFF may cause a packet to "loop back" to a previous SF on a path in order that a sequence of functions may be re-executed. This is simply achieved by replacing the SI in the NSH with a higher value, instead of decreasing it as would normally be the case, to determine the next hop in the path.

Section 2 also describes how an SFI or SFF may cause a packet to "jump forward" to an SF on a path that is not the immediate next SF in the SFP. This is simply achieved by replacing the SI in the NSH with a lower value than would be achieved by decreasing it by the normal amount.

A more complex option to move packets from one SFP to another is described in [RFC8300] and Section 2, where it is termed "branching". This mechanism allows an SFI or SFF to make a choice of downstream treatments for packets based on local policy and the output of the local SF. Branching is achieved by changing the SPI in the NSH to indicate the new path and setting the SI to indicate the point in the path at which the packets enter.

Note that the NSH does not include a marker to indicate whether a specific packet has been around a loop before. Therefore, the use of NSH metadata [RFC8300] may be required in order to prevent infinite loops.

6.1. Protocol Control of Looping, Jumping, and Branching

If the SFT value in an SFT TLV in an SFPR has the special-purpose SFT value "Change Sequence" (see Section 10), then this is an indication that the SFF may make a loop, jump, or branch according to local policy and information returned by the local SFI.

In this case, the SPI and SI of the next hop are encoded in the eight bytes of an entry in the SFIR-RD list as follows:

3 bytes SPI

1 byte SI

4 bytes Reserved (**SHOULD** be set to zero and ignored)

If the SI in this encoding is not part of the SFPR indicated by the SPI in this encoding, then this is an explicit error that **SHOULD** be detected by the SFF when it parses the SFPR. The SFPR **SHOULD NOT** cause any forwarding state to be installed in the SFF, and packets received with the SPI that indicates this SFPR **SHOULD** be silently discarded.

If the SPI in this encoding is unknown, the SFF **SHOULD NOT** install any forwarding state for this SFPR but **MAY** hold the SFPR pending receipt of another SFPR that does use the encoded SPI.

If the SPI matches the current SPI for the path, this is a loop or jump. In this case, if the SI is greater than or equal to the current SI, it is a loop. If the SPI matches and the SI is less than the next SI, it is a jump.

If the SPI indicates another path, this is a branch, and the SI indicates the point at which to enter that path.

The Change Sequence SFT is just another SFT that may appear in a set of SFI/SFT tuples within an SI and is selected as described in Section 5.

Note that special-purpose SFTs **MUST NOT** be advertised in SFIRs. If such an SFIR is received, it **SHOULD** be ignored.

6.2. Implications for Forwarding State

Support for looping and jumping requires that the SFF has forwarding state established to an SFF that provides access to an instance of the appropriate SF. This means that the SFF must have seen the relevant SFIR advertisements and mush have known that it needed to create the forwarding state. This is a matter of local configuration and implementation; for example, an implementation could be configured to install forwarding state for specific looping/jumping.

Support for branching requires that the SFF has forwarding state established to an SFF that provides access to an instance of the appropriate entry SF on the other SFP. This means that the SFF must have seen the relevant SFIR and SFPR advertisements and known that it needed to

create the forwarding state. This is a matter of local configuration and implementation; for example, an implementation could be configured to install forwarding state for specific branching (identified by SPI and SI).

7. Advanced Topics

This section highlights several advanced topics introduced elsewhere in this document.

7.1. Correlating Service Function Path Instances

It is often useful to create bidirectional SFPs to enable packet flows to traverse the same set of SFs, but in the reverse order. However, packets on SFPs in the data plane (per [RFC8300]) do not contain a direction indicator, so each direction must use a different SPI.

As described in Section 3.2.1.1, an SFPR can contain one or more correlators encoded in Association TLVs. If the Association Type indicates "Bidirectional SFP", then the SFP advertised in the SFPR is one direction of a bidirectional pair of SFPs, where the other in the pair is advertised in the SFPR with RD as carried in the "Associated SFPR-RD" field of the Association TLV. The SPI carried in the "Associated SPI" field of the Association TLV provides a cross-check against the SPI advertised in the SFPR with RD as carried in the "Associated SFPR-RD" field of the Association TLV.

As noted in Section 3.2.1.1, when SFPRs reference each other, one SFPR advertisement will be received before the other. Therefore, processing of an association will require that the first SFPR not be rejected simply because the Associated SFPR-RD it carries is unknown. However, the SFP defined by the first SFPR is valid and **SHOULD** be available for use as a unidirectional SFP, even in the absence of an advertisement of its partner.

Furthermore, in error cases where SFPR-a associates with SFPR-b, but SFPR-b associates with SFPR-c such that a bidirectional pair of SFPs cannot be formed, the individual SFPs are still valid and **SHOULD** be available for use as unidirectional SFPs. An implementation **SHOULD** log this situation, because it represents a controller error.

Usage of a bidirectional SFP may be programmed into the classifiers by the controller. Alternatively, a classifier may look at incoming packets on a bidirectional packet flow, extract the SPI from the received NSH, and look up the SFPR to find the reverse-direction SFP to use when it sends packets.

See Section 8 for an example of how this works.

7.2. Considerations for Stateful Service Functions

Some service functions are stateful. That means that they build and maintain state derived from configuration or the packet flows that they handle. In such cases, it can be important or necessary that all packets from a flow continue to traverse the same instance of a service function so that the state can be leveraged and does not need to be regenerated.

In the case of bidirectional SFPs, it may be necessary to traverse the same instances of a stateful service function in both directions. A firewall is a good example of such a service function.

This issue becomes a concern where there are multiple parallel instances of a service function and a determination of which one to use could normally be left to the SFF as a load-balancing or local-policy choice.

For the forward-direction SFP, the concern is that the same choice of SF is made for all packets of a flow under normal network conditions. It may be possible to guarantee that the load-balancing functions applied in the SFFs are stable and repeatable, but a controller that constructs SFPs might not want to trust to this. The controller can, in these cases, build a number of more specific SFPs, each traversing a specific instance of the stateful SFs. In this case, the load-balancing choice can be left up to the classifier. Thus, the classifier selects which instance of a stateful SF is used by a particular flow by selecting the SFP that the flow uses.

For bidirectional SFPs where the same instance of a stateful SF must be traversed in both directions, it is not enough to leave the choice of SFI as a local choice, even if the load balancing is stable, because coordination would be required between the decision points in the forward and reverse directions, and this may be hard to achieve in all cases except where it is the same SFF that makes the choice in both directions.

Note that this approach necessarily increases the amount of SFP state in the network (i.e., there are more SFPs). It is possible to mitigate this effect by careful construction of SFPs built from a concatenation of other SFPs.

Section 8.9 includes some simple examples of SFPs for stateful SFs.

7.3. VPN Considerations and Private Service Functions

Likely deployments include reserving specific instances of SFs for specific customers or allowing customers to deploy their own SFs within the network. Building SFs in such environments requires that suitable identifiers be used to ensure that SFFs distinguish which SFIs can be used and which cannot.

This problem is similar to a problem in the way that VPNs are supported and is solved in a similar way. The "RT" field is used to indicate a set of SFs from which all choices must be made.

7.4. Flow Specification for SFC Classifiers

[RFC8955] defines a set of BGP routes that can be used to identify the packets in a given flow using fields in the header of each packet, and a set of actions -- encoded as Extended Communities -- that can be used to disposition those packets. This document enables the use of these mechanisms by SFC classifiers by defining a new action Extended Community called "Flow Specification for SFC Classifiers", identified by the value 0x0d. Note that implementation of this section of this specification will be controllers or classifiers communicating with each other directly for the purpose of instructing the classifier how to place packets onto an SFP. So that the implementation of classifiers can be kept simple, and to avoid the confusion between the purposes of different Extended Communities, a controller **MUST NOT** include other action

Extended Communities at the same time as a "Flow Specification for SFC Classifiers" Extended Community. A "Flow Specification for SFC Classifiers" Traffic Filtering Action Extended Community advertised with any other Traffic Filtering Action Extended Community **MUST** be treated as malformed in line with [RFC8955] and result in the flow-specification UPDATE message being handled as "treat-as-withdraw", according to [RFC7606], Section 2.

To put the flow specification into context, when multiple service function chaining overlays are present in one network, each FlowSpec update **MUST** be tagged with the route target of the overlay or VPN network for which it is intended.

This Extended Community is encoded as an 8-octet value, as shown in Figure 10.

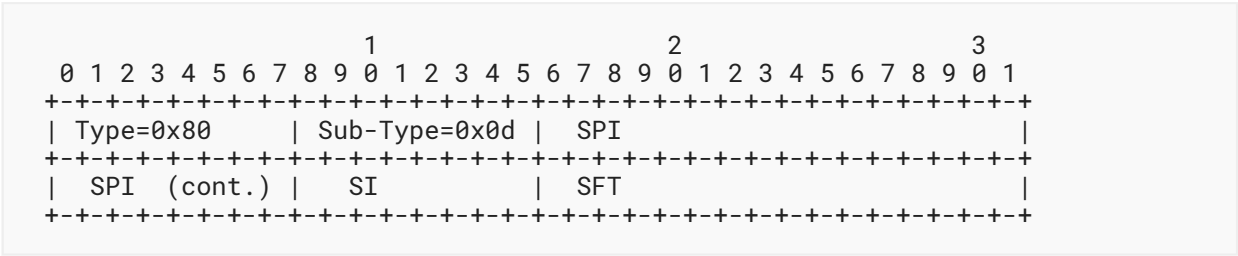

Figure 10: The Format of the Flow Specification for SFC Classifiers Extended Community

The Extended Community contains the Service Path Identifier (SPI), Service Index (SI), and Service Function Type (SFT), as defined elsewhere in this document. Thus, each action extended community defines the entry point (not necessarily the first hop) into a specific SFP. This allows, for example, different flows to enter the same SFP at different points.

Note that, according to [RFC8955], a given flow-specification update may include multiple of these action Extended Communities. If a given action extended community does not contain an installed SFPR with the specified {SPI, SI, SFT}, it **MUST NOT** be used for dispositioning the packets of the specified flow.

The normal case of packet classification for service function chaining will see a packet enter the SFP at its first hop. In this case, the SI in the Extended Community is superfluous, and the SFT may also be unnecessary. To allow these cases to be handled, a special meaning is assigned to an SI of zero (not a valid value) and an SFT of zero (a reserved value in the registry -- see Section 10.5).

- If an SFC Classifiers Extended Community is received with SI = 0, then it means that the first hop of the SFP indicated by the SPI **MUST** be used.
- If an SFC Classifiers Extended Community is received with SFT = 0, then there are two subcases:
  - If there is a choice of SFT in the hop indicated by the value of the SI (including SI = 0), then SFT = 0 means there is a free choice of which SFT to use, according to local policy).
  - If there is no choice of SFT in the hop indicated by the value of SI, then SFT = 0 means that the value of the SFT at that hop, as indicated in the SFPR for the indicated SPI, **MUST** be used.

One of the filters that the flow specification may describe is the VPN to which the traffic belongs. Additionally, as noted above, to put the indicated SPI into context when multiple SFC overlays are present in one network, each FlowSpec update **MUST** be tagged with the route target of the overlay or VPN network for which it is intended.

Note that future extensions might be made to the Flow Specification for SFC Classifiers Extended Community to provide instruction to the classifier about what metadata to add to packets that it classifies for forwarding on a specific SFP; however, that is outside the scope of this document.

7.5. Choice of Data Plane SPI/SI Representation

This document ties together the control and data planes of a service function chaining overlay network through the use of the SPI/SI that is nominally carried in the NSH of a given packet. However, in order to handle situations in which the NSH is not ubiquitously deployed, it is also possible to use alternative data plane representations of the SPI/SI by carrying the identical semantics in other protocol fields, such as MPLS labels [RFC8595].

This document defines a new Sub-TLV for the tunnel encapsulation attribute [RFC9012], the SPI/SI Representation Sub-TLV of type 16. This Sub-TLV **MAY** be present in each Tunnel TLV contained in a tunnel encapsulation attribute when the attribute is carried by an SFIR. The "Value" field of this Sub-TLV is a two-octet field of flags numbered counting from the most significant bit, each of which describes how the originating SFF expects to see the SPI/SI represented in the data plane for packets carried in the tunnels described by the Tunnel TLV.

The following bits are defined by this document and are tracked in an IANA registry described in Section 10.10:

Bit 0: If this bit is set, the NSH is to be used to carry the SPI/SI in the data plane.

Bit 1: If this bit is set, two labels in an MPLS label stack are to be used as described in Section 7.5.1.

If a given Tunnel TLV does not contain an SPI/SI Representation Sub-TLV, then it **MUST** be processed as if such a Sub-TLV is present with Bit 0 set and no other bits set. That is, the absence of the Sub-TLV **SHALL** be interpreted to mean that the NSH is to be used.

If a given Tunnel TLV contains an SPI/SI Representation Sub-TLV with a "Value" field that has no flag set, then the tunnel indicated by the Tunnel TLV **MUST NOT** be used for forwarding SFC packets. If a given Tunnel TLV contains an SPI/SI Representation Sub-TLV with both bit 0 and bit 1 set, then the tunnel indicated by the Tunnel TLV **MUST NOT** be used for forwarding SFC packets. The meaning and rules for the presence of other bits is to be defined in future documents, but implementations of this specification **MUST** set other bits to zero and ignore them on receipt.

If a given Tunnel TLV contains more than one SPI/SI Representation Sub-TLV, then the first one **MUST** be considered and subsequent instances **MUST** be ignored.

Note that the MPLS representation of the logical NSH may be used even if the tunnel is not an MPLS tunnel. Conversely, MPLS tunnels may be used to carry other encodings of the logical NSH (specifically, the NSH itself). It is a requirement that both ends of a tunnel over the underlay network know that the tunnel is used for service function chaining and know what form of NSH representation is used. The signaling mechanism described here allows coordination of this information.

7.5.1. MPLS Representation of the SPI/SI

If bit 1 is set in the SPI/SI Representation Sub-TLV, then labels in the MPLS label stack are used to indicate SFC forwarding and processing instructions to achieve the semantics of a logical NSH. The label stack is encoded as shown in [RFC8595].

7.6. MPLS Label Swapping/Stacking Operation

When a classifier constructs an MPLS label stack for an SFP, it starts with that SFP's last hop. If the last hop requires an {SPI, SI} label pair for label swapping, it pushes the SI (set to the SI value of the last hop) and the SFP's SPI onto the MPLS label stack. If the last hop requires a {context label, SFI label} label pair for label stacking, it selects a specific SFIR and pushes that SFIR's SFI label and context label onto the MPLS label stack.

The classifier then moves sequentially back through the SFP one hop at a time. For each hop, if the hop requires an {SPI, SI} and there is an {SPI, SI} at the top of the MPLS label stack, the SI is set to the SI value of the current hop. If there is not an {SPI, SI} at the top of the MPLS label stack, it pushes the SI (set to the SI value of the current hop) and the SFP's SPI onto the MPLS label stack.

If the hop requires a {context label, SFI label}, it selects a specific SFIR and pushes that SFIR's SFI label and context label onto the MPLS label stack.

7.7. Support for MPLS-Encapsulated NSH Packets

[RFC8596] describes how to transport SFC packets using the NSH over an MPLS transport network. Signaling that this approach is in use is supported by this document as follows:

- A "BGP Tunnel Encapsulation Attribute" Sub-TLV is included with the codepoint 10 (representing "MPLS Label Stack") from the "BGP Tunnel Encapsulation Attribute Sub-TLVs" registry defined in [RFC9012].
- An "SFP Traversal With MPLS Label Stack" TLV is included containing an "SPI/SI Representation" Sub-TLV with bit 0 set and bit 1 cleared.

In this case, the MPLS label stack constructed by the SFF to forward a packet to the next SFF on the SFP will consist of the labels needed to reach that SFF, and if label stacking is used, it will also include the labels advertised in the MPLS Label Stack Sub-TLV and the labels remaining in the stack needed to traverse the remainder of the SFP.

8. Examples

Most of the examples in this section use IPv4 addressing. But there is nothing special about IPv4 in the mechanisms described in this document, and they are equally applicable to IPv6. A few examples using IPv6 addressing are provided in Section 8.10.

Assume we have a service function overlay network with four SFFs (SFF1, SFF2, SFF3, and SFF4). The SFFs have addresses in the underlay network as follows:

```
SFF1 192.0.2.1
SFF2 192.0.2.2
SFF3 192.0.2.3
SFF4 192.0.2.4
```

Each SFF provides access to some SFIs from the four SFTs, SFT=41, SFT=42, SFT=43, and SFT=44, as follows:

```
SFF1 SFT=41 and SFT=42
SFF2 SFT=41 and SFT=43
SFF3 SFT=42 and SFT=44
SFF4 SFT=43 and SFT=44
```

The service function network also contains a controller with address 198.51.100.1.

This example service function overlay network is shown in Figure 11.

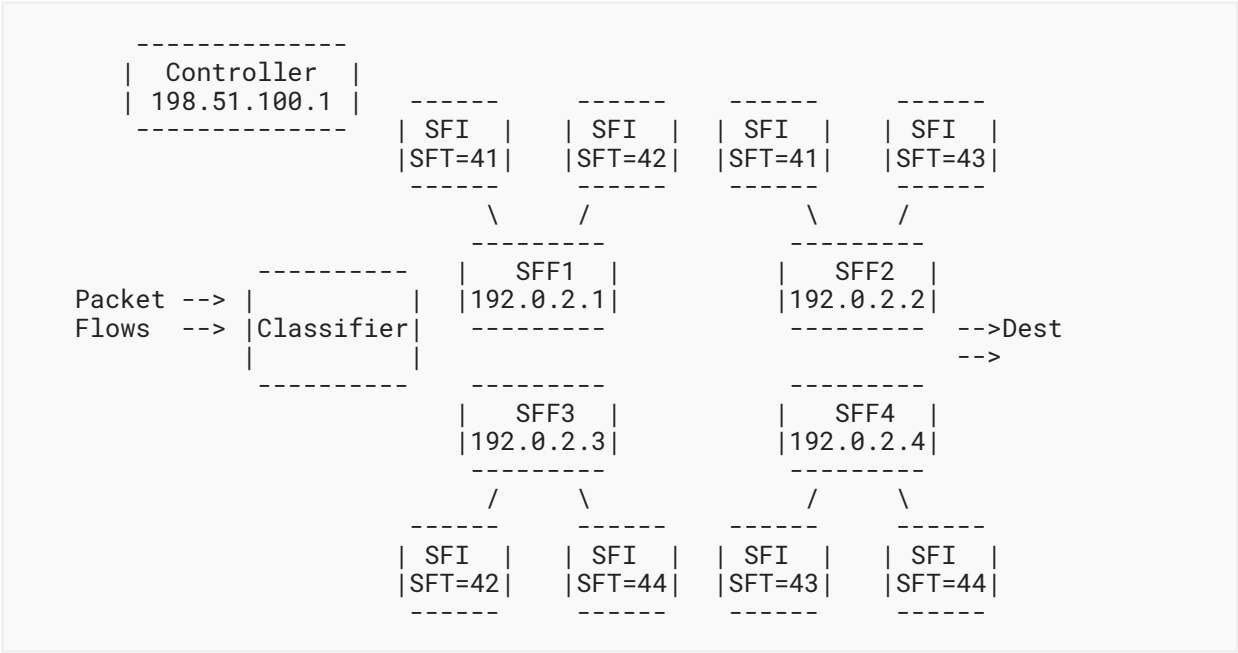

Figure 11: Example Service Function Overlay Network

The SFFs advertise routes to the SFIs they support. These advertisements contain RDs that are set according to the network operator's configuration model. In all of these IPv4 examples, we use RDs of Type 1 such that the available six octets are partitioned as four octets for the IPv4 address of the advertising SFF, and two octets that are a local index of the SFI. This scheme is chosen purely for convenience of documentation, and an operator is totally free to use any other scheme so long as it conforms to the definitions of SFIR and SFPR in Sections 3.1 and 3.2.

Thus, we see the following SFIRs advertised:

```
RD = 192.0.2.1/1, SFT = 41
RD = 192.0.2.1/2, SFT = 42
RD = 192.0.2.2/1, SFT = 41
RD = 192.0.2.2/2, SFT = 43
RD = 192.0.2.3/7, SFT = 42
RD = 192.0.2.3/8, SFT = 44
RD = 192.0.2.4/5, SFT = 43
RD = 192.0.2.4/6, SFT = 44
```

Note that the addressing used for communicating between SFFs is taken from the tunnel encapsulation attribute of the SFIR and not from the SFIR-RD.

8.1. Example Explicit SFP with No Choices

Consider the following SFPR.

```
SFP1:  RD = 198.51.100.1/101, SPI = 15,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, SFT = 43, RD = 192.0.2.2/2]
```

The SFP consists of an SF of Type 41 located at SFF1, followed by an SF of Type 43 located at SFF2. This path is fully explicit, and each SFF is offered no choice in forwarding packets along the path.

SFF1 will receive packets on the path from the classifier and will identify the path from the SPI (15). The initial SI will be 255, and so SFF1 will deliver the packets to the SFI for SFT 41.

When the packets are returned to SFF1 by the SFI, the SI will be decreased to 250 for the next hop. SFF1 has no flexibility in the choice of SFF to support the next-hop SFI and will forward the packet to SFF2, which will send the packets to the SFI that supports SFT 43 before forwarding the packets to their destinations.

8.2. Example SFP with Choice of SFIs

```
SFP2:  RD = 198.51.100.1/102, SPI = 16,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, SFT = 43, {RD = 192.0.2.2/2,
                             RD = 192.0.2.4/5 } ]
```

In this example, the path also consists of an SF of Type 41 located at SFF1, and this is followed by an SF of Type 43. However, in this case, the SI = 250 contains a choice between the SFI located at SFF2 and the SFI located at SFF4.

SFF1 will receive packets on the path from the classifier and will identify the path from the SPI (16). The initial SI will be 255, and so SFF1 will deliver the packets to the SFI for SFT 41.

When the packets are returned to SFF1 by the SFI, the SI will be decreased to 250 for the next hop. SFF1 now has a choice of next-hop SFFs to execute the next hop in the path. It can either forward packets to SFF2 or SFF4 to execute a function of Type 43. It uses its local load-balancing algorithm to make this choice. The chosen SFF will send the packets to the SFI that supports SFT 43 before forwarding the packets to their destinations.

8.3. Example SFP with Open Choice of SFIs

```
SFP3:  RD = 198.51.100.1/103, SPI = 17,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, SFT = 44, RD = 0]
```

In this example, the path also consists of an SF of Type 41 located at SFF1, and this is followed by an SI with an RD of zero and SF of Type 44. This means that a choice can be made between any SFF that supports an SFI of Type 44.

SFF1 will receive packets on the path from the classifier and will identify the path from the SPI (17). The initial SI will be 255, and so SFF1 will deliver the packets to the SFI for SFT 41.

When the packets are returned to SFF1 by the SFI, the SI will be decreased to 250 for the next hop. SFF1 now has a free choice of next-hop SFFs to execute the next hop in the path, selecting between all SFFs that support SFs of Type 44. Looking at the SFIRs it has received, SFF1 knows that SF Type 44 is supported by SFF3 and SFF4. SFF1 uses its local load-balancing algorithm to make this choice. The chosen SFF will send the packets to the SFI that supports SFT 44 before forwarding the packets to their destinations.

8.4. Example SFP with Choice of SFTs

```
SFP4:  RD = 198.51.100.1/104, SPI = 18,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, {SFT = 43, RD = 192.0.2.2/2,
                   SFT = 44, RD = 192.0.2.3/8 } ]
```

This example provides a choice of SF type in the second hop in the path. The SI of 250 indicates a choice between SF Type 43 located at SF2 and SF Type 44 located at SF3.

SFF1 will receive packets on the path from the classifier and will identify the path from the SPI (18). The initial SI will be 255, and so SFF1 will deliver the packets to the SFI for SFT 41.

When the packets are returned to SFF1 by the SFI, the SI will be decreased to 250 for the next hop. SFF1 now has a free choice of next-hop SFFs to execute the next hop in the path, selecting between all SFFs that support an SF of Type 43 and SFF3, which supports an SF of Type 44. These may be completely different functions that are to be executed dependent on specific conditions, or they may be similar functions identified with different type identifiers (such as firewalls from different vendors). SFF1 uses its local policy and load-balancing algorithm to make this choice and may use additional information passed back from the local SFI to help inform its selection. The chosen SFF will send the packets to the SFI that supports the chosen SFT before forwarding the packets to their destinations.

8.5. Example Correlated Bidirectional SFPs

```
SFP5:  RD = 198.51.100.1/105, SPI = 19,
       Assoc-Type = 1, Assoc-RD = 198.51.100.1/106, Assoc-SPI = 20,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, SFT = 43, RD = 192.0.2.2/2]

SFP6:  RD = 198.51.100.1/106, SPI = 20,
       Assoc-Type = 1, Assoc-RD = 198.51.100.1/105, Assoc-SPI = 19,
       [SI = 254, SFT = 43, RD = 192.0.2.2/2],
       [SI = 249, SFT = 41, RD = 192.0.2.1/1]
```

This example demonstrates correlation of two SFPs to form a bidirectional SFP, as described in Section 7.1.

Two SFPRs are advertised by the controller. They have different SPIs (19 and 20), so they are known to be separate SFPs, but they both have Association TLVs with Association Type set to 1, indicating bidirectional SFPs. Each has an "Associated SFPR-RD" field containing the value of the other SFPR-RD to correlate the two SFPs as a bidirectional pair.

As can be seen from the SFPRs in this example, the paths are symmetric: the hops in SFP5 appear in the reverse order in SFP6.

8.6. Example Correlated Asymmetrical Bidirectional SFPs

```
SFP7:  RD = 198.51.100.1/107, SPI = 21,
       Assoc-Type = 1, Assoc-RD = 198.51.100.1/108, Assoc-SPI = 22,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, SFT = 43, RD = 192.0.2.2/2]

SFP8:  RD = 198.51.100.1/108, SPI = 22,
       Assoc-Type = 1, Assoc-RD = 198.51.100.1/107, Assoc-SPI = 21,
       [SI = 254, SFT = 44, RD = 192.0.2.4/6],
       [SI = 249, SFT = 41, RD = 192.0.2.1/1]
```

Asymmetric bidirectional SFPs can also be created. This example shows a pair of SFPs with distinct SPIs (21 and 22) that are correlated in the same way as in the example in Section 8.5.

However, unlike in that example, the SFPs are different in each direction. Both paths include a hop of SF Type 41, but SFP7 includes a hop of SF Type 43 supported at SFF2, while SFP8 includes a hop of SF Type 44 supported at SFF4.

8.7. Example Looping in an SFP

```
SFP9:  RD = 198.51.100.1/109, SPI = 23,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, SFT = 44, RD = 192.0.2.4/5],
       [SI = 245, {SFT = 1, RD = {SPI=23, SI=255, Rsv=0},
                   SFT = 42, RD = 192.0.2.3/7 } ]
```

Looping and jumping are described in Section 6. This example shows an SFP that contains an explicit loop-back instruction that is presented as a choice within an SFP hop.

The first two hops in the path (SI = 255 and SI = 250) are normal. That is, the packets will be delivered to SFF1 and SFF4 in turn for execution of SFs of Type 41 and 44, respectively.

The third hop (SI = 245) presents SFF4 with a choice of next hop. It can either forward the packets to SFF3 for an SF of Type 42 (the second choice) or it can loop back.

The loop-back entry in the SFPR for SI = 245 is indicated by the special-purpose SFT value 1 ("Change Sequence"). Within this hop, the RD is interpreted as encoding the SPI and SI of the next hop (see Section 6.1). In this case, the SPI is 23, which indicates that this is a loop or branch, i.e., the next hop is on the same SFP. The SI is set to 255; this is a higher number than the current SI (245), indicating a loop.

SFF4 must make a choice between these two next hops. The packet will be either forwarded to SFF3 with the NSH SI decreased to 245 or looped back to SFF1 with the NSH SI reset to 255. This choice will be made according to local policy, information passed back by the local SFI, and details in the packets' metadata that are used to prevent infinite looping.

8.8. Example Branching in an SFP

```
SFP10:  RD = 198.51.100.1/110, SPI = 24,
       [SI = 254, SFT = 42, RD = 192.0.2.3/7],
       [SI = 249, SFT = 43, RD = 192.0.2.2/2]

SFP11:  RD = 198.51.100.1/111, SPI = 25,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, SFT = 1, RD = {SPI=24, SI=254, Rsv=0}]
```

Branching follows a similar procedure to that for looping (and jumping), as shown in Section 8.7. However, there are two SFPs involved.

SFP10 shows a normal path with packets forwarded to SFF3 and SFF2 for execution of service functions of Type 42 and 43, respectively.

SFP11 starts as normal (SFF1 for an SF of Type 41), but then SFF1 processes the next hop in the path and finds a "Change Sequence" special-purpose SFT. The "SFIR-RD" field includes an SPI of 24, which indicates SFP10, not the current SFP. The SI in the SFIR-RD is 254, so SFF1 knows that it must set the SPI/SI in the NSH to 24/254 and send the packets to the appropriate SFF, as advertised in the SFPR for SFP10 (that is, SFF3).

8.9. Examples of SFPs with Stateful Service Functions

This section provides some examples to demonstrate establishing SFPs when there is a choice of service functions at a particular hop, and where consistency of choice is required in both directions. The scenarios that give rise to this requirement are discussed in Section 7.2.

8.9.1. Forward and Reverse Choice Made at the SFF

Consider the topology shown in Figure 12. There are three SFFs arranged neatly in a line, and the middle one (SFF2) supports three SFIs all of SFT 42. These three instances can be used by SFF2 to load balance so that no one instance is swamped.

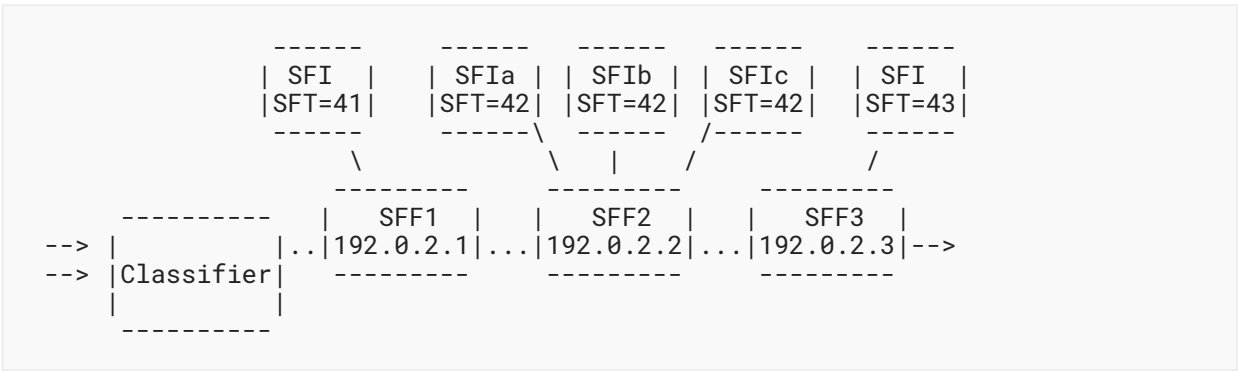

Figure 12: Example Where Choice Is Made at the SFF

This leads to the following SFIRs being advertised.

```
RD = 192.0.2.1/11, SFT = 41
RD = 192.0.2.2/11, SFT = 42  (for SFIa)
RD = 192.0.2.2/12, SFT = 42  (for SFIb)
RD = 192.0.2.2/13, SFT = 42  (for SFIc)
RD = 192.0.2.3/11, SFT = 43
```

The controller can create a single forward SFP (SFP12), giving SFF2 the choice of which SFI to use to provide a function of SFT 42, as follows. The load-balancing choice between the three available SFIs is assumed to be within the capabilities of the SFF, and if the SFs are stateful, it is assumed that the SFF knows this and arranges load balancing in a stable, flow-dependent way.

```
SFP12:  RD = 198.51.100.1/112, SPI = 26,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/113, Assoc-SPI = 27,
       [SI = 255, SFT = 41, RD = 192.0.2.1/11],
       [SI = 254, SFT = 42, {RD = 192.0.2.2/11,
                             192.0.2.2/12,
                             192.0.2.2/13 }],
       [SI = 253, SFT = 43, RD = 192.0.2.3/11]
```

The reverse SFP (SFP13) in this case may also be created as shown below, using association with the forward SFP and giving the load-balancing choice to SFF2. This is safe, even in the case that the SFs of Type 42 are stateful, because SFF2 is doing the load balancing in both directions and can apply the same algorithm to ensure that packets associated with the same flow use the same SFI regardless of the direction of travel.

```
SFP13:  RD = 198.51.100.1/113, SPI = 27,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/112, Assoc-SPI = 26,
       [SI = 255, SFT = 43, RD = 192.0.2.3/11],
       [SI = 254, SFT = 42, {RD = 192.0.2.2/11,
                             192.0.2.2/12,
                             192.0.2.2/13 }],
       [SI = 253, SFT = 41, RD = 192.0.2.1/11]
```

How an SFF knows that an attached SFI is stateful is out of the scope of this document. It is assumed that this will form part of the process by which SFIs are registered as local to SFFs. Section 7.2 provides additional observations about the coordination of the use of stateful SFIs in the case of bidirectional SFPs.

In general, the problems of load balancing and the selection of the same SFIs in both directions of a bidirectional SFP can be addressed by using sufficiently precisely specified SFPs (specifying the exact SFIs to use) and suitable programming of the classifiers at each end of the SFPs to make sure that the matching pair of SFPs are used.

8.9.2. Parallel End-to-End SFPs with Shared SFF

The mechanism described in Section 8.9.1 might not be desirable because of the functional assumptions it places on SFF2 to be able to load balance with suitable flow identification, stability, and equality in both directions. Instead, it may be desirable to place the responsibility for flow classification in the classifier and let it determine load balancing with the implied choice of SFIs.

Consider the network graph as shown in Figure 12 and with the same set of SFIRs as listed in Section 8.9.1. In this case, the controller could specify three forward SFPs with their corresponding associated reverse SFPs. Each bidirectional pair of SFPs uses a different SFI for the

SF of Type 42. The controller can instruct the classifier how to place traffic on the three bidirectional SFPs, or it can treat them as a group, leaving the classifier responsible for balancing the load.

```
SFP14:  RD = 198.51.100.1/114, SPI = 28,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/117, Assoc-SPI = 31,
       [SI = 255, SFT = 41, RD = 192.0.2.1/11],
       [SI = 254, SFT = 42, RD = 192.0.2.2/11],
       [SI = 253, SFT = 43, RD = 192.0.2.3/11]

SFP15:  RD = 198.51.100.1/115, SPI = 29,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/118, Assoc-SPI = 32,
       [SI = 255, SFT = 41, RD = 192.0.2.1/11],
       [SI = 254, SFT = 42, RD = 192.0.2.2/12],
       [SI = 253, SFT = 43, RD = 192.0.2.3/11]

SFP16:  RD = 198.51.100.1/116, SPI = 30,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/119, Assoc-SPI = 33,
       [SI = 255, SFT = 41, RD = 192.0.2.1/11],
       [SI = 254, SFT = 42, RD = 192.0.2.2/13],
       [SI = 253, SFT = 43, RD = 192.0.2.3/11]

SFP17:  RD = 198.51.100.1/117, SPI = 31,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/114, Assoc-SPI = 28,
       [SI = 255, SFT = 43, RD = 192.0.2.3/11],
       [SI = 254, SFT = 42, RD = 192.0.2.2/11],
       [SI = 253, SFT = 41, RD = 192.0.2.1/11]

SFP18:  RD = 198.51.100.1/118, SPI = 32,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/115, Assoc-SPI = 29,
       [SI = 255, SFT = 43, RD = 192.0.2.3/11],
       [SI = 254, SFT = 42, RD = 192.0.2.2/12],
       [SI = 253, SFT = 41, RD = 192.0.2.1/11]

SFP19:  RD = 198.51.100.1/119, SPI = 33,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/116, Assoc-SPI = 30,
       [SI = 255, SFT = 43, RD = 192.0.2.3/11],
       [SI = 254, SFT = 42, RD = 192.0.2.2/13],
       [SI = 253, SFT = 41, RD = 192.0.2.1/11]
```

8.9.3. Parallel End-to-End SFPs with Separate SFFs

While the examples in Sections 8.9.1 and 8.9.2 place the choice of SFI as subtended from the same SFF, it is also possible that the SFIs are each subtended from a different SFF, as shown in Figure 13. In this case, it is harder to coordinate the choices for forward and reverse paths without some form of coordination between SFF1 and SFF3. Therefore, it would be normal to consider end-to-end parallel SFPs, as described in Section 8.9.2.

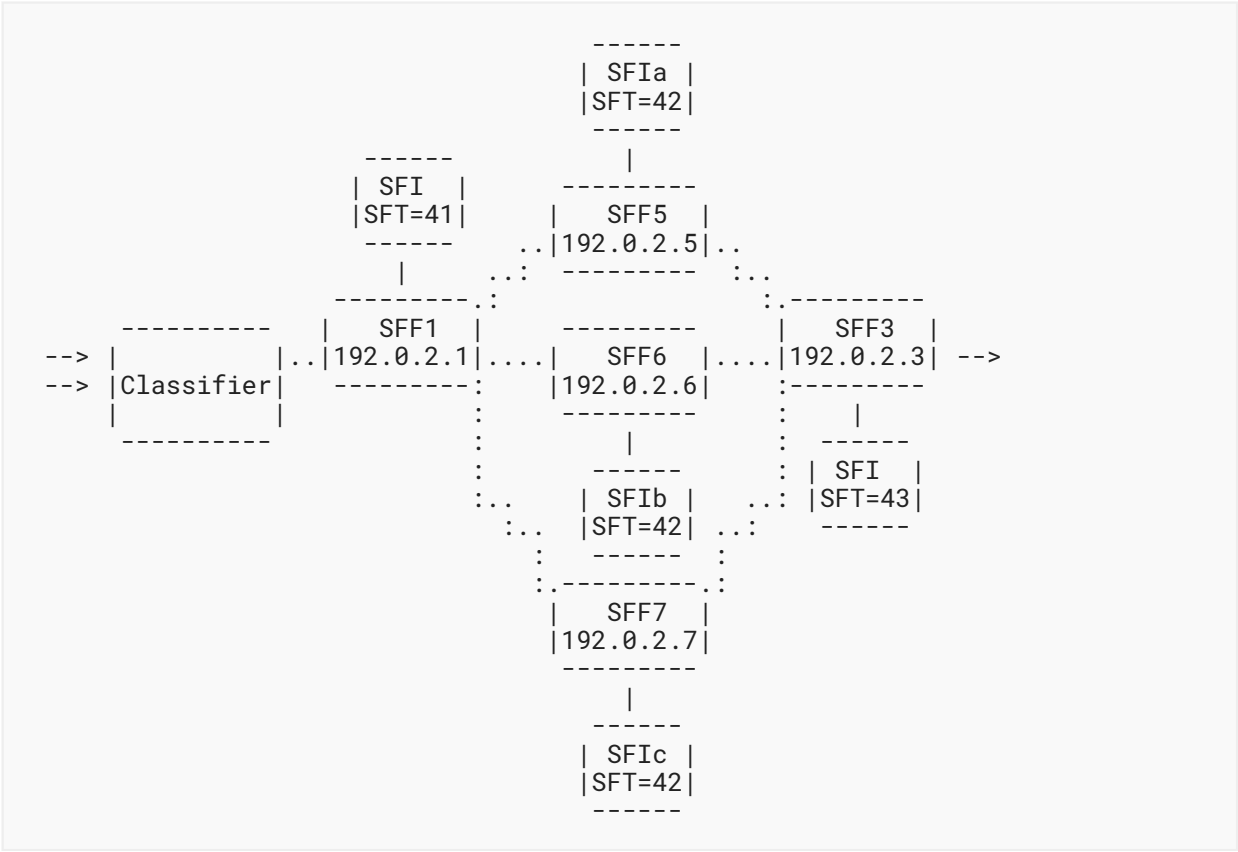

Figure 13: Second Example with Parallel End-to-End SFPs

In this case, five SFIRs are advertised as follows:

```
RD = 192.0.2.1/11, SFT = 41
RD = 192.0.2.5/11, SFT = 42  (for SFIa)
RD = 192.0.2.6/11, SFT = 42  (for SFIb)
RD = 192.0.2.7/11, SFT = 42  (for SFIc)
RD = 192.0.2.3/11, SFT = 43
```

In this case, the controller could specify three forward SFPs with their corresponding associated reverse SFPs. Each bidirectional pair of SFPs uses a different SFF and SFI for the middle hop (for an SF of Type 42). The controller can instruct the classifier how to place traffic on the three bidirectional SFPs, or it can treat them as a group, leaving the classifier responsible for balancing the load.

```
SFP20:  RD = 198.51.100.1/120, SPI = 34,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/123, Assoc-SPI = 37,
       [SI = 255, SFT = 41, RD = 192.0.2.1/11],
       [SI = 254, SFT = 42, RD = 192.0.2.5/11],
       [SI = 253, SFT = 43, RD = 192.0.2.3/11]

SFP21:  RD = 198.51.100.1/121, SPI = 35,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/124, Assoc-SPI = 38,
       [SI = 255, SFT = 41, RD = 192.0.2.1/11],
       [SI = 254, SFT = 42, RD = 192.0.2.6/11],
       [SI = 253, SFT = 43, RD = 192.0.2.3/11]

SFP22:  RD = 198.51.100.1/122, SPI = 36,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/125, Assoc-SPI = 39,
       [SI = 255, SFT = 41, RD = 192.0.2.1/11],
       [SI = 254, SFT = 42, RD = 192.0.2.7/11],
       [SI = 253, SFT = 43, RD = 192.0.2.3/11]

SFP23:  RD = 198.51.100.1/123, SPI = 37,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/120, Assoc-SPI = 34,
       [SI = 255, SFT = 43, RD = 192.0.2.3/11],
       [SI = 254, SFT = 42, RD = 192.0.2.5/11],
       [SI = 253, SFT = 41, RD = 192.0.2.1/11]

SFP24:  RD = 198.51.100.1/124, SPI = 38,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/121, Assoc-SPI = 35,
       [SI = 255, SFT = 43, RD = 192.0.2.3/11],
       [SI = 254, SFT = 42, RD = 192.0.2.6/11],
       [SI = 253, SFT = 41, RD = 192.0.2.1/11]

SFP25:  RD = 198.51.100.1/125, SPI = 39,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/122, Assoc-SPI = 36,
       [SI = 255, SFT = 43, RD = 192.0.2.3/11],
       [SI = 254, SFT = 42, RD = 192.0.2.7/11],
       [SI = 253, SFT = 41, RD = 192.0.2.1/11]
```

8.9.4. Parallel SFPs Downstream of the Choice

The mechanism of parallel SFPs demonstrated in Section 8.9.3 is perfectly functional and may be practical in many environments. However, there may be scaling concerns because of the large amount of state (knowledge of SFPs -- i.e., SFPR advertisements retained) if there is a very large number of possible SFIs (for example, tens of instances of the same stateful SF) or if there are multiple choices of stateful SF along a path. This situation may be mitigated using SFP fragments that are combined to form the end-to-end SFPs.

The example presented here is necessarily simplistic but should convey the basic principle. The example presented in Figure 14 is similar to that in Section 8.9.3 but with an additional first hop.

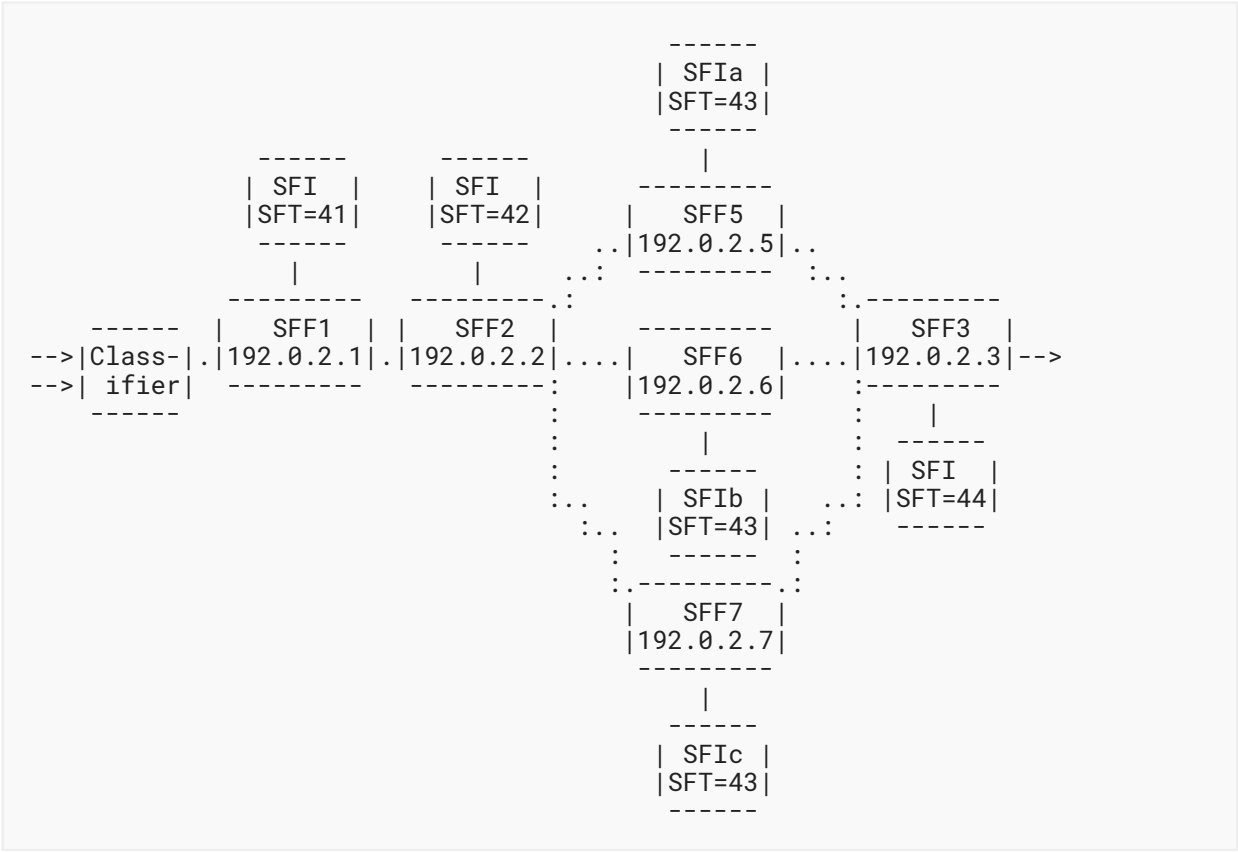

Figure 14: Example with Parallel SFPs Downstream of Choice

The six SFIs are advertised as follows:

```
RD = 192.0.2.1/11, SFT = 41
RD = 192.0.2.2/11, SFT = 42
RD = 192.0.2.5/11, SFT = 43  (for SFIa)
RD = 192.0.2.6/11, SFT = 43  (for SFIb)
RD = 192.0.2.7/11, SFT = 43  (for SFIc)
RD = 192.0.2.3/11, SFT = 44
```

SFF2 is the point at which a load-balancing choice must be made. So "tail-end" SFPs are constructed as follows. Each takes in a different SFF that provides access to an SF of Type 43.

```
SFP26:  RD = 198.51.100.1/126, SPI = 40,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/130, Assoc-SPI = 44,
       [SI = 255, SFT = 43, RD = 192.0.2.5/11],
       [SI = 254, SFT = 44, RD = 192.0.2.3/11]

SFP27:  RD = 198.51.100.1/127, SPI = 41,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/131, Assoc-SPI = 45,
       [SI = 255, SFT = 43, RD = 192.0.2.6/11],
       [SI = 254, SFT = 44, RD = 192.0.2.3/11]

SFP28:  RD = 198.51.100.1/128, SPI = 42,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/132, Assoc-SPI = 46,
       [SI = 255, SFT = 43, RD = 192.0.2.7/11],
       [SI = 254, SFT = 44, RD = 192.0.2.3/11]
```

Now an end-to-end SFP with load-balancing choice can be constructed as follows. The choice made by SFF2 is expressed in terms of entering one of the three "tail-end" SFPs.

```
SFP29:  RD = 198.51.100.1/129, SPI = 43,
       [SI = 255, SFT = 41, RD = 192.0.2.1/11],
       [SI = 254, SFT = 42, RD = 192.0.2.2/11],
       [SI = 253, {SFT = 1, RD = {SPI=40, SI=255, Rsv=0},
                            RD = {SPI=41, SI=255, Rsv=0},
                            RD = {SPI=42, SI=255, Rsv=0} } ]
```

Now, despite the load-balancing choice being made elsewhere than at the initial classifier, it is possible for the reverse SFPs to be well constructed without any ambiguity. The three reverse paths appear as follows.

```
SFP30:  RD = 198.51.100.1/130, SPI = 44,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/126, Assoc-SPI = 40,
       [SI = 255, SFT = 44, RD = 192.0.2.4/11],
       [SI = 254, SFT = 43, RD = 192.0.2.5/11],
       [SI = 253, SFT = 42, RD = 192.0.2.2/11],
       [SI = 252, SFT = 41, RD = 192.0.2.1/11]

SFP31:  RD = 198.51.100.1/131, SPI = 45,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/127, Assoc-SPI = 41,
       [SI = 255, SFT = 44, RD = 192.0.2.4/11],
       [SI = 254, SFT = 43, RD = 192.0.2.6/11],
       [SI = 253, SFT = 42, RD = 192.0.2.2/11],
       [SI = 252, SFT = 41, RD = 192.0.2.1/11]

SFP32:  RD = 198.51.100.1/132, SPI = 46,
      Assoc-Type = 1, Assoc-RD = 198.51.100.1/128, Assoc-SPI = 42,
       [SI = 255, SFT = 44, RD = 192.0.2.4/11],
       [SI = 254, SFT = 43, RD = 192.0.2.7/11],
       [SI = 253, SFT = 42, RD = 192.0.2.2/11],
       [SI = 252, SFT = 41, RD = 192.0.2.1/11]
```

8.10. Examples Using IPv6 Addressing

This section provides several examples using IPv6 addressing. As will be seen from the examples, there is nothing special or clever about using IPv6 addressing rather than IPv4 addressing.

The reference network for these IPv6 examples is based on that described at the top of Section 8 and shown in Figure 11.

Assume we have a service function overlay network with four SFFs (SFF1, SFF3, SFF3, and SFF4). The SFFs have addresses in the underlay network as follows:

```
   SFF1 2001:db8::192:0:2:1
   SFF2 2001:db8::192:0:2:2
   SFF3 2001:db8::192:0:2:3
   SFF4 2001:db8::192:0:2:4
```

Each SFF provides access to some SFIs from the four service function types SFT=41, SFT=42, SFT=43, and SFT=44, just as before:

```
   SFF1 SFT=41 and SFT=42
   SFF2 SFT=41 and SFT=43
   SFF3 SFT=42 and SFT=44
   SFF4 SFT=43 and SFT=44
```

The service function network also contains a controller with address 2001:db8::198:51:100:1.

This example service function overlay network is shown in Figure 15.

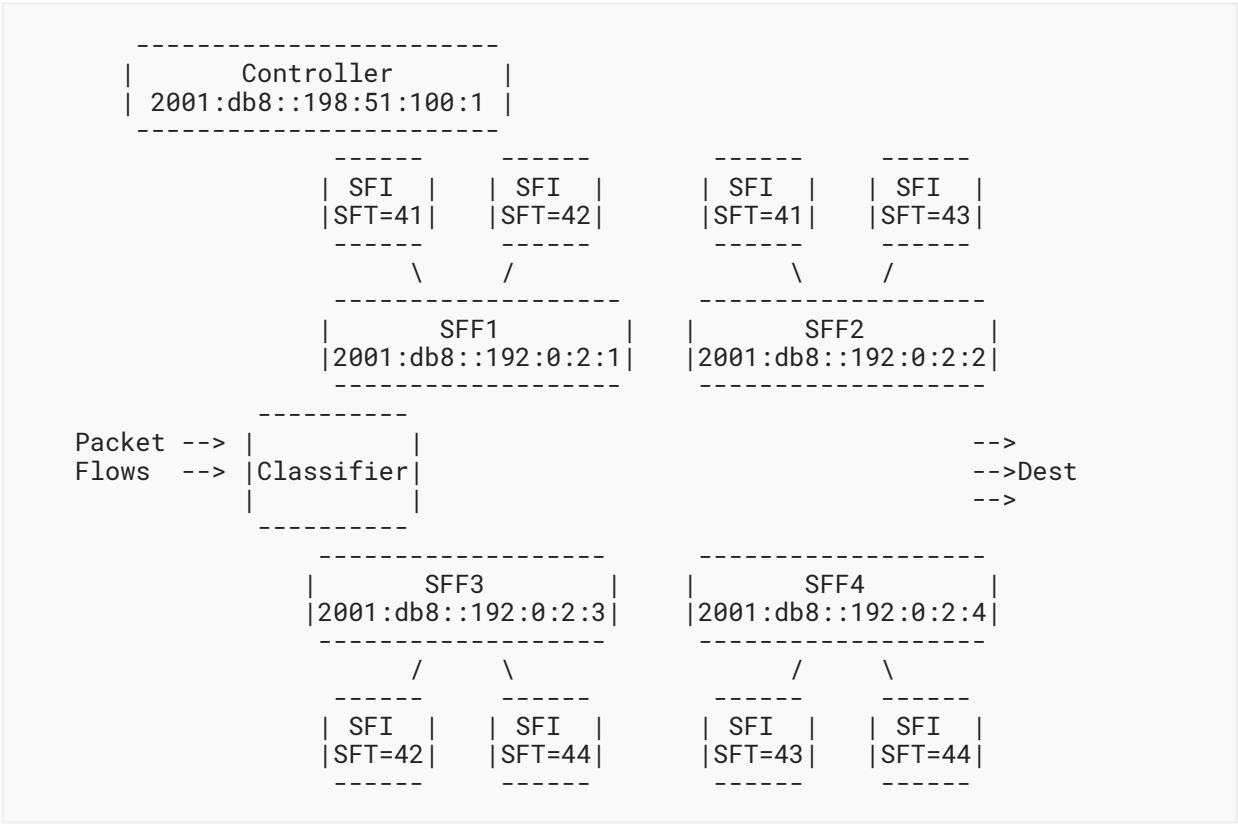

Figure 15: Example Service Function Overlay Network

The SFFs advertise routes to the SFIs they support. These advertisements contain RDs that are set according to the network operator's configuration model. Note that in an IPv6 network, the RD is not large enough to contain the full IPv6 address, as only six octets are available. So, in all of these IPv6 examples, we use RDs of Type 1 such that the available six octets are partitioned as four octets for an IPv4 address of the advertising SFF, and two octets that are a local index of the SFI. Furthermore, we have chosen an IPv6 addressing scheme so that the low-order four octets of the IPv6 address match an IPv4 address of the advertising node. This scheme is chosen purely for convenience of documentation, and an operator is totally free to use any other scheme so long as it conforms to the definitions of SFIR and SFPR in Sections 3.1 and 3.2.

Observant readers will notice that this makes the BGP advertisements shown in these examples exactly the same as in the previous examples. All that is different is that the advertising SFFs and controller have IPv6 addresses.

Thus, we see the following SFIRs advertised.

The SFFs advertise routes to the SFIs they support. So we see the following SFIRs:

```
RD = 192.0.2.1/1, SFT = 41
RD = 192.0.2.1/2, SFT = 42
RD = 192.0.2.2/1, SFT = 41
RD = 192.0.2.2/2, SFT = 43
RD = 192.0.2.3/7, SFT = 42
RD = 192.0.2.3/8, SFT = 44
RD = 192.0.2.4/5, SFT = 43
RD = 192.0.2.4/6, SFT = 44
```

Note that the addressing used for communicating between SFFs is taken from the tunnel encapsulation attribute of the SFIR and not from the SFIR-RD.

8.10.1. Example Explicit SFP with No Choices

Consider the following SFPR similar to that in Section 8.1.

```
SFP1:  RD = 198.51.100.1/101, SPI = 15,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, SFT = 43, RD = 192.0.2.2/2]
```

The SFP consists of an SF of Type 41 located at SFF1, followed by an SF of Type 43 located at SFF2. This path is fully explicit, and each SFF is offered no choice in forwarding a packet along the path.

SFF1 will receive packets on the path from the classifier and will identify the path from the SPI (15). The initial SI will be 255, and so SFF1 will deliver the packets to the SFI for SFT 41.

When the packets are returned to SFF1 by the SFI, the SI will be decreased to 250 for the next hop. SFF1 has no flexibility in the choice of SFF to support the next-hop SFI and will forward the packet to SFF2, which will send the packets to the SFI that supports SFT 43 before forwarding the packets to their destinations.

8.10.2. Example SFP with Choice of SFIs

```
SFP2:  RD = 198.51.100.1/102, SPI = 16,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, SFT = 43, {RD = 192.0.2.2/2,
                             RD = 192.0.2.4/5 } ]
```

In this example, like that in Section 8.2, the path also consists of an SF of Type 41 located at SFF1, and this is followed by an SF of Type 43; but in this case, the SI = 250 contains a choice between the SFI located at SFF2 and the SFI located at SFF4.

SFF1 will receive packets on the path from the classifier and will identify the path from the SPI (16). The initial SI will be 255, and so SFF1 will deliver the packets to the SFI for SFT 41.

When the packets are returned to SFF1 by the SFI, the SI will be decreased to 250 for the next hop. SFF1 now has a choice of next-hop SFFs to execute the next hop in the path. It can either forward packets to SFF2 or SFF4 to execute a function of Type 43. It uses its local load-balancing algorithm to make this choice. The chosen SFF will send the packets to the SFI that supports SFT 43 before forwarding the packets to their destinations.

8.10.3. Example SFP with Open Choice of SFIs

```
SFP3:  RD = 198.51.100.1/103, SPI = 17,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, SFT = 44, RD = 0]
```

In this example, like that in Section 8.3, the path also consists of an SF of Type 41 located at SFF1, and this is followed by an SI with an RD of zero and SF of Type 44. This means that a choice can be made between any SFF that supports an SFI of Type 44.

SFF1 will receive packets on the path from the classifier and will identify the path from the SPI (17). The initial SI will be 255, and so SFF1 will deliver the packets to the SFI for SFT 41.

When the packets are returned to SFF1 by the SFI, the SI will be decreased to 250 for the next hop. SFF1 now has a free choice of next-hop SFFs to execute the next hop in the path, selecting between all SFFs that support SFs of Type 44. Looking at the SFIRs it has received, SFF1 knows that SF Type 44 is supported by SFF3 and SFF4. SFF1 uses its local load-balancing algorithm to make this choice. The chosen SFF will send the packets to the SFI that supports SFT 44 before forwarding the packets to their destinations.

8.10.4. Example SFP with Choice of SFTs

```
SFP4:  RD = 198.51.100.1/104, SPI = 18,
       [SI = 255, SFT = 41, RD = 192.0.2.1/1],
       [SI = 250, {SFT = 43, RD = 192.0.2.2/2,
                   SFT = 44, RD = 192.0.2.3/8 } ]
```

This example, similar to that in Section 8.4, provides a choice of SF type in the second hop in the path. The SI of 250 indicates a choice between SF Type 43 located through SF2 and SF Type 44 located at SF3.

SFF1 will receive packets on the path from the classifier and will identify the path from the SPI (18). The initial SI will be 255, and so SFF1 will deliver the packets to the SFI for SFT 41.

When the packets are returned to SFF1 by the SFI, the SI will be decreased to 250 for the next hop. SFF1 now has a free choice of next-hop SFFs to execute the next hop in the path, selecting between all SFFs that support an SF of Type 43 and SFF3, which supports an SF of Type 44. These may be completely different functions that are to be executed dependent on specific conditions, or they may be similar functions identified with different type identifiers (such as firewalls from different vendors). SFF1 uses its local policy and load-balancing algorithm to make this choice,

and it may use additional information passed back from the local SFI to help inform its selection. The chosen SFF will send the packets to the SFI that supports the chosen SFT before forwarding the packets to their destinations.

9. Security Considerations

The mechanisms in this document use BGP for the control plane. Hence, techniques such as those discussed in [RFC5925] can be used to help authenticate BGP sessions and, thus, the messages between BGP peers, making it harder to spoof updates (which could be used to install bogus SFPs or advertise false SIs) or withdrawals.

Further discussion of security considerations for BGP may be found in the BGP specification itself [RFC4271] and the security analysis for BGP [RFC4272]. [RFC5925] contains a discussion of the inappropriateness of the TCP MD5 signature option for protecting BGP sessions. [RFC6952] includes an analysis of BGP keying and authentication issues.

Additionally, this document depends on other documents that specify BGP Multiprotocol Extensions and the documents that define the attributes that are carried by BGP UPDATEs of the SFC AFI/SAFI. [RFC4760] observes that the use of AFI/SAFI does not change the underlying security issues inherent in the existing BGP. Relevant additional security measures are considered in [RFC9012].

This document does not fundamentally change the security behavior of BGP deployments, which depend considerably on the network operator's perception of risk in their network. It may be observed that the application of the mechanisms described in this document is scoped to a single domain, as implied by [RFC8300] and noted in Section 2.1 of this document. Applicability of BGP within a single domain may enable a network operator to make easier and more consistent decisions about what security measures to apply, and the domain boundary, which BGP enforces by definition, provides a safeguard that prevents leakage of SFC programming in either direction at the boundary.

Service function chaining provides a significant attack opportunity; packets can be diverted from their normal paths through the network, packets can be made to execute unexpected functions, and the functions that are instantiated in software can be subverted. However, this specification does not change the existence of service function chaining, and security issues specific to service function chaining are covered in [RFC7665] and [RFC8300].

This document defines a control plane for service function chaining. Clearly, this provides an attack vector for a service function chaining system, as an attack on this control plane could be used to make the system misbehave. Thus, the security of the BGP system is critically important to the security of the whole service function chaining system. The control plane mechanisms are very similar to those used for BGP/MPLS IP VPNs as described in [RFC4364], and so the security considerations in that document (Section 13) provide good guidance for securing service function chaining systems reliant on this specification. Of particular relevance is the need to securely distinguish between messages intended for the control of different SFC overlays, which is similar to the need to distinguish between different VPNs. Section 19 of [RFC7432] also provides useful guidance on the use of BGP in a similar environment.

Note that a component of a service function chaining system that uses the procedures described in this document also requires communications between a controller and the service function chaining network elements (specifically the SFFs and classifiers). This communication covers instructing the classifiers using BGP mechanisms (see Section 7.4); therefore, the use of BGP security is strongly recommended. But it also covers other mechanisms for programming the classifier and instructing the SFFs and SFs (for example, to bind SFs to an SFF, and to cause the establishment of tunnels between SFFs). This document does not cover these latter mechanisms, and so their security is out of scope, but it should be noted that these communications provide an attack vector on the service function chaining system, and so attention must be paid to ensuring that they are secure.

There is an intrinsic assumption in service function chaining systems that nodes that announce support for specific SFs actually offer those functions and that SFs are not, themselves, attacked or subverted. This is particularly important when the SFs are implemented as software that can be updated. Protection against this sort of concern forms part of the security of any service function chaining system and so is outside the scope of the control plane mechanisms described in this document.

Similarly, there is a vulnerability if a rogue or subverted controller announces SFPs, especially if that controller "takes over" an existing SFP and changes its contents. This corresponds to a rogue BGP speaker entering a routing system, or even a Route Reflector becoming subverted. Protection mechanisms, as above, include securing BGP sessions and protecting software loads on the controllers.

In an environment where there is concern that rogue controllers might be introduced to the network and inject false SFPRs or take over and change existing SFPRs, it is **RECOMMENDED** that each SFF and classifier be configured with the identities of authorized controllers. Thus, the announcement of an SFPR by any other BGP peer would be rejected.

Lastly, note that Section 3.2.2 makes two operational suggestions that have implications for the stability and security of the mechanisms described in this document:

- That modifications to active SFPs not be made.
- That SPIs not be immediately reused.

10. IANA Considerations

10.1. New BGP AF/SAFI

IANA maintains the "Address Family Numbers" registry. IANA has assigned a new Address Family Number from the "Standards Action" range called "BGP SFC" (31), with this document as a reference.

IANA maintains the "Subsequent Address Family Identifiers (SAFI) Parameters" registry. IANA has assigned a new SAFI value from the "Standards Action" range called "BGP SFC" (9), with this document as a reference.

10.2. "SFP attribute" BGP Path Attribute

IANA maintains a registry of "Border Gateway Protocol (BGP) Parameters" with a subregistry of "BGP Path Attributes". IANA has assigned a new Path attribute called "SFP attribute" with a value of 37 and with this document as a reference.

10.3. "SFP Attribute TLVs" Registry

IANA maintains a registry of "Border Gateway Protocol (BGP) Parameters". IANA has created a new subregistry called the "SFP Attribute TLVs" registry.

Valid values are in the range 0 to 65535.

- Values 0 and 65535 are marked "Reserved".
- Values 1 through 65534 are to be assigned according to the "First Come First Served" policy [RFC8126].

This document is a reference for this registry.

The registry tracks:

- Type
- Name
- Reference
- Registration Date

The registry is initially populated as follows:

Type	Name	Reference	Registration Date
1	Association TLV	RFC 9015	2020-09-02
2	Hop TLV	RFC 9015	2020-09-02
3	SFT TLV	RFC 9015	2020-09-02
4	MPLS Swapping/Stacking	RFC 9015	2020-09-02
5	SFP Traversal With MPLS	RFC 9015	2020-09-02

Table 1: SFP Attribute TLVs Subregistry Initial Contents

10.4. "SFP Association Type" Registry

IANA maintains a registry of "Border Gateway Protocol (BGP) Parameters". IANA has created a new subregistry called the "SFP Association Type" registry.

Valid values are in the range 0 to 65535.

- Values 0 and 65535 are marked "Reserved".
- Values 1 through 65534 are assigned according to the "First Come First Served" policy [RFC8126].

This document is given as a reference for this registry.

The new registry tracks:

- Association Type
- Name
- Reference
- Registration Date

The registry should initially be populated as follows:

Association Type	Name	Reference	Date
1	Bidirectional SFP	RFC 9015	2020-09-02

Table 2: SFP Association Type Subregistry Initial Contents

10.5. "Service Function Chaining Service Function Types" Registry

IANA has created a new top-level registry called "Service Function Chaining Service Function Types".

Valid values are in the range 0 to 65535.

- Values 0 and 65535 are marked "Reserved".
- Values 1 through 31 are to be assigned by "Standards Action" [RFC8126] and are referred to as the "special-purpose SFT values".
- Values 32 through 64495 are to be assigned according to the "First Come First Served" policy [RFC8126].
- Values 64496 through 65534 are for Private Use and are not to be recorded by IANA.

This document is given as a reference for this registry.

The registry tracks:

- Value
- Name
- Reference
- Registration Date

The registry is initially populated as follows.

Value	Name	Reference	Date
0	Reserved	RFC 9015	2020-09-02
1	Change Sequence	RFC 9015	2020-09-02
2-31	Unassigned		
32	Classifier	RFC 9015, [BGP-LS-SR]	2020-09-02
33	Firewall	RFC 9015, [BGP-LS-SR]	2020-09-02
34	Load balancer	RFC 9015, [BGP-LS-SR]	2020-09-02
35	Deep packet inspection engine	RFC 9015, [BGP-LS-SR]	2020-09-02
36	Penalty box	RFC 9015, [RFC8300]	2020-09-02
37	WAN accelerator	RFC 9015, [RFC7665], [RFC8300]	2020-09-02
38	Application accelerator	RFC 9015, [RFC7665]	2020-09-02
39	TCP optimizer	RFC 9015, [RFC7665]	2020-09-02
40	Network Address Translator	RFC 9015, [RFC7665]	2020-09-02
41	NAT44	RFC 9015, [RFC7665], [RFC3022]	2020-09-02
42	NAT64	RFC 9015, [RFC7665], [RFC6146]	2020-09-02
43	NPTv6	RFC 9015, [RFC7665], [RFC6296]	2020-09-02
44	Lawful intercept	RFC 9015, [RFC7665]	2020-09-02
45	HOST_ID injection	RFC 9015, [RFC7665]	2020-09-02
46	HTTP header enrichment	RFC 9015, [RFC7665]	2020-09-02
47	Caching engine	RFC 9015, [RFC7665]	2020-09-02
48-64495	Unassigned		
64496-65534	Reserved for Private Use		

Value	Name	Reference	Date
65535	Reserved, not to be allocated	RFC 9015	2020-09-02

Table 3: Service Function Chaining Service Function Types Registry Initial Contents

10.6. Flow Specification for SFC Classifiers

IANA maintains a registry of "Border Gateway Protocol (BGP) Extended Communities" with a subregistry of "Generic Transitive Experimental Use Extended Community Sub-Types". IANA has assigned a new subtype as follows:

> "Flow Specification for SFC Classifiers" with a value of 0x0d and with this document as the reference.

10.7. New BGP Transitive Extended Community Type

IANA maintains a registry of "Border Gateway Protocol (BGP) Extended Communities" with a subregistry of "BGP Transitive Extended Community Types". IANA has assigned a new type as follows:

> SFC (Sub-Types are defined in the "SFC Extended Community Sub-Types" registry) with a value of 0x0b and with this document as the reference.

10.8. "SFC Extended Community Sub-Types" Registry

IANA maintains a registry of "Border Gateway Protocol (BGP) Parameters". IANA has created a new subregistry called the "SFC Extended Community Sub-Types" registry.

IANA has included the following note:

> This registry contains values of the second octet (the "Sub-Type" field) of an extended community when the value of the first octet (the "Type" field) is set to 0x0b.

The allocation policy for this registry is First Come First Served.

Valid values are 0 to 255. The value 0 is reserved and should not be allocated.

IANA has populated this registry with the following entries:

Sub-Type Value	Name	Reference	Date
0	Reserved	RFC 9015	
1	SFIR pool identifier	RFC 9015	2020-09-02

Sub-Type Value	Name	Reference	Date
2	MPLS Label Stack Mixed Swapping/Stacking Labels	RFC 9015	2020-09-02
3-255	Unassigned		

Table 4: SFC Extended Community Sub-Types Subregistry Initial Contents

10.9. New SPI/SI Representation Sub-TLV

IANA has assigned a codepoint from the "BGP Tunnel Encapsulation Attribute Sub-TLVs" registry for the "SPI/SI Representation Sub-TLV" with a value of 16 and with this document as the reference.

10.10. "SFC SPI/SI Representation Flags" Registry

IANA maintains the "BGP Tunnel Encapsulation Attribute Sub-TLVs" registry and has created an associated registry called the "SFC SPI/SI Representation Flags" registry.

Bits are to be assigned by Standards Action. The field is 16 bits long, and bits are counted from the most significant bit as bit zero.

IANA has populated the registry as follows:

Value	Name	Reference
0	NSH data plane	RFC 9015
1	MPLS data plane	RFC 9015

Table 5: SFC SPI/SI Representation Flags Registry Initial Contents

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/info/rfc4271>.

[RFC4360] Sangli, S., Tappan, D., and Y. Rekhter, "BGP Extended Communities Attribute", RFC 4360, DOI 10.17487/RFC4360, February 2006, <https://www.rfc-editor.org/info/rfc4360>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <https://www.rfc-editor.org/info/rfc4364>.

[RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <https://www.rfc-editor.org/info/rfc4760>.

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/info/rfc7432>.

[RFC7606] Chen, E., Ed., Scudder, J., Ed., Mohapatra, P., and K. Patel, "Revised Error Handling for BGP UPDATE Messages", RFC 7606, DOI 10.17487/RFC7606, August 2015, <https://www.rfc-editor.org/info/rfc7606>.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/info/rfc7665>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8300] Quinn, P., Ed., Elzur, U., Ed., and C. Pignataro, Ed., "Network Service Header (NSH)", RFC 8300, DOI 10.17487/RFC8300, January 2018, <https://www.rfc-editor.org/info/rfc8300>.

[RFC8595] Farrel, A., Bryant, S., and J. Drake, "An MPLS-Based Forwarding Plane for Service Function Chaining", RFC 8595, DOI 10.17487/RFC8595, June 2019, <https://www.rfc-editor.org/info/rfc8595>.

[RFC8596] Malis, A., Bryant, S., Halpern, J., and W. Henderickx, "MPLS Transport Encapsulation for the Service Function Chaining (SFC) Network Service Header (NSH)", RFC 8596, DOI 10.17487/RFC8596, June 2019, <https://www.rfc-editor.org/info/rfc8596>.

[RFC8955] Loibl, C., Hares, S., Raszuk, R., McPherson, D., and M. Bacher, "Dissemination of Flow Specification Rules", RFC 8955, DOI 10.17487/RFC8955, December 2020, <https://www.rfc-editor.org/info/rfc8955>.

[RFC9012] Patel, K., Van de Velde, G., Sangli, S., and J. Scudder, "The BGP Tunnel Encapsulation Attribute", RFC 9012, DOI 10.17487/RFC9012, April 2021, <https://www.rfc-editor.org/info/rfc9012>.

11.2. Informative References

[BGP-LS-SR] Dawra, G., Filsfils, C., Talaulikar, K., Clad, F., Bernier, D., Uttaro, J., Decraene, B., Elmalky, H., Xu, X., Guichard, J., and C. Li, "BGP-LS Advertisement of Segment Routing Service Segments", Work in Progress, Internet-Draft, draft-dawra-idr-bgp-ls-sr-service-segments-05, 15 February 2021, <https://tools.ietf.org/html/draft-dawra-idr-bgp-ls-sr-service-segments-05>.

[RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, DOI 10.17487/RFC3022, January 2001, <https://www.rfc-editor.org/info/rfc3022>.

[RFC4272] Murphy, S., "BGP Security Vulnerabilities Analysis", RFC 4272, DOI 10.17487/RFC4272, January 2006, <https://www.rfc-editor.org/info/rfc4272>.

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, DOI 10.17487/RFC5925, June 2010, <https://www.rfc-editor.org/info/rfc5925>.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <https://www.rfc-editor.org/info/rfc6146>.

[RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, DOI 10.17487/RFC6296, June 2011, <https://www.rfc-editor.org/info/rfc6296>.

[RFC6952] Jethanandani, M., Patel, K., and L. Zheng, "Analysis of BGP, LDP, PCEP, and MSDP Issues According to the Keying and Authentication for Routing Protocols (KARP) Design Guide", RFC 6952, DOI 10.17487/RFC6952, May 2013, <https://www.rfc-editor.org/info/rfc6952>.

[RFC7498] Quinn, P., Ed. and T. Nadeau, Ed., "Problem Statement for Service Function Chaining", RFC 7498, DOI 10.17487/RFC7498, April 2015, <https://www.rfc-editor.org/info/rfc7498>.

Acknowledgements

Thanks to Tony Przygienda, Jeff Haas, and Andy Malis for helpful comments, and to Joel Halpern for discussions that improved this document. Yuanlong Jiang provided a useful review and caught some important issues. Stephane Litkowski did an exceptionally good and detailed Document Shepherd review.

Andy Malis contributed text that formed the basis of Section 7.7.

Brian Carpenter and Martin Vigoureux provided useful reviews during IETF Last Call. Thanks also to Sheng Jiang, Med Boucadair, Ravi Singh, Benjamin Kaduk, Roman Danyliw, Adam Roach, Alvaro Retana, Barry Leiba, and Murray Kucherawy for review comments. Ketan Talaulikar provided helpful discussion of the SFT codepoint registry. Ron Bonica kept us honest on the difference between an RD and an RT; Benjamin Kaduk kept us on message about the difference between an RD and an Extended Community.

Contributors

Stuart Mackie
Juniper Networks
Email: wsmackie@juinper.net

Keyur Patel
Arrcus, Inc.
Email: keyur@arrcus.com

Avinash Lingala
AT&T
Email: ar977m@att.com

Authors' Addresses

Adrian Farrel
Old Dog Consulting
Email: adrian@olddog.co.uk

John Drake
Juniper Networks
Email: jdrake@juniper.net

Eric Rosen
Juniper Networks
Email: erosen52@gmail.com

Jim Uttaro
AT&T
Email: ju1738@att.com

Luay Jalil
Verizon
Email: luay.jalil@verizon.com