

SFC working group
Internet-Draft
Intended status: Standards Track
Expires: January 5, 2015

H. Song
J. You
L. Yong
Y. Jiang
Huawei
July 4, 2014

SFC Header Mapping for Legacy SF
draft-song-sfc-legacy-sf-mapping-02

Abstract

A Service Function Chain (SFC) goes through a list of ordered service functions. One assumption of this document is that legacy service function can participate in the service function chain, but they are not aware of the SFC header, nor interpret it. This document provides a mechanism between an SFC proxy and an SFC-unaware Service Function (i.e. legacy SF), to identify the SFC header associated with a packet that is returned from a legacy SF, without SFC header being explicitly carried in the wired protocol between SFC Proxy and legacy SF.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119](#)].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 5, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
2.	Terminology	4
3.	Mechanisms	4
3.1.	For Transparent Service Functions	4
3.1.1.	Layer 2 MAC Address	4
3.1.2.	VLAN	5
3.1.3.	QinQ	6
3.1.4.	VXLAN	7
3.2.	For Non-transparent Service Functions	9
4.	Operation Consideration	9
5.	Security considerations	11
6.	Acknowledgement	11
7.	Informative References	11
	Authors' Addresses	11

[1.](#) Introduction

A service function chain goes through a list of ordered service functions. One assumption of this document is that some service functions are kept as legacy, and they do not have to be aware of the SFC header, nor interpret it. This document provides a mechanism between an SFC proxy and a legacy SF ([\[I-D.quinn-sfc-arch\]](#)), to identify the SFC header associated with a packet that is returned from a legacy SF, without anything in the SFC header being explicitly carried in the wired protocol between an SFC proxy and a legacy SF.

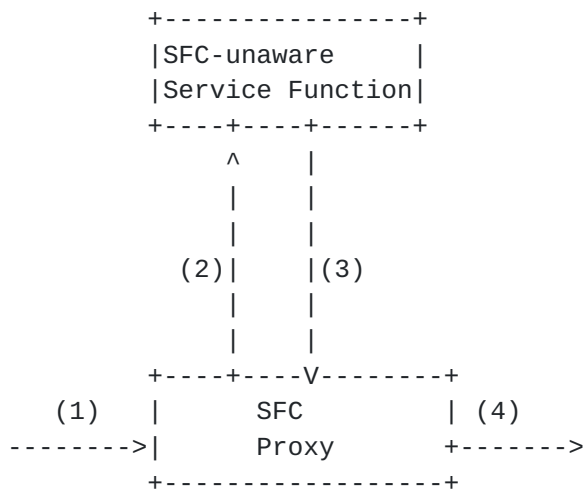


Figure 1: Procedure of a packet processed by a legacy SF

The legacy service function (i.e. SFC-unaware service function in the Figure 1) only handles packets without SFC header, because it does not understand the SFC header. One advantage is that the existing service functions don't need to be upgraded to support SFC. Otherwise it may be a hindrance for the widely adoption of SFC.

Assuming that for some legacy SFs, the packet header is transparent to them, i.e., this kind of SFs will not modify the layer 2 or layer 3 packet headers. If the payload in the SFC encapsulation is layer 3 traffic, it will be kept as it is, and a new layer 2 header will be added before sending to the SF. However if the payload in the SFC encapsulation is layer 2 traffic, the SFC proxy may modify the original source MAC address and use the new source MAC address for mapping to the stored SFC header. This will not impact the SF processing. The SF will send the traffic back after processing. For the current stage, we leave the legacy SFs which modify the original packet headers as an open issue for further study.

As shown in Figure 1, there are four steps. The SFC proxy receives a packet, and removes its SFC header, which may optionally contain metadata, and stores the SFC header locally, and then sends the original packet to the SF. After SF processing the packet, the traffic will be sent back to the SFC proxy. The SFC proxy retrieves the pre- stored SFC header accordingly, and encapsulates the packet with the SFC header, and then sends the packet to next-hop service function. The key problem here is how to map the packet to its original SFC header.

If the SFC header is not changed per flow at a certain point, e.g., a specific SFC proxy (i.e. each flow has a specific SFC header in a SFC proxy, but in another SFC proxy, the SFC header is different), then the SFC proxy needs to find the original SFC header per flow. If the

SFC header is changed per packet for a specific flow at a certain point, then the SFC proxy needs to find the original SFC header per packet. The second case may happen if different packets in a flow carry different metadata (e.g. the metadata can be injected to the packet by a DPI appliance). It's also the reason why five-tuple cannot be used for the mapping to retrieve the original SFC header.

An expiration time can be used for each mapping entry in the SFC proxy. If the SFC header in that entry has not been retrieved after the expiration time, the entry will be deleted from the entry table.

2. Terminology

The terminology used in this document is defined below:

Legacy SF: A conventional service function that does not support SFC header, i.e. SFC-unaware SF.

Transparent SF: A service function that does not change any bit of the original service packet header (Layer 2, layer 3, and layer 4) sent to it, but it may drop packets.

Non-transparent SF: A service function that changes some part of the original service packet header sent to it.

Original Service Packet: The payload in a SFC encapsulation packet or a packet constructed based on the original payload.

3. Mechanisms

The mechanisms used in this document require that each forwarding entity and its connected service functions in a same layer 2 network. The following are considerations mainly for transparent SFs. If the original payload packet is a layer 2 packet, and the mapping method used is layer 2 MAC address, then the assumption is that the SF does not need to look into the layer 2 header. If it does, other mechanisms should be used.

3.1. For Transparent Service Functions

If the service function is transparent to packet headers, the following methods can be used for SFC header mapping.

3.1.1. Layer 2 MAC Address

The layer 2 MAC address is used to associate a SFC header between SFC proxy and SF, i.e. each SFC header will be assigned a source MAC address on the SFC proxy. If SFC header can be changed per packet,

then SFC proxy assigns a new source MAC address for each packet it received, otherwise, it assigns a new MAC address for each flow it received.

When SFC proxy received the returned packet from the SF, it retrieves the packet's original SFC header by using the MAC address as a key. And then it encapsulates the packet with that SFC header and sends to the next hop.

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

```

Outer Ethernet Header:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               SF Destination MAC Address               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| SF Destination MAC Address   | SFC Proxy Source MAC Address |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               SFC Proxy Source MAC Address               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   Ethertype = 0x0800           |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Original IP Payload:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               Original Payload               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

[3.1.2.](#) VLAN

If the network between the SFC proxy and SF is a layer 2 network, and in the case that a SF need to look into the MAC address of the packet, then VLAN can be used for the mapping between them. The SFC proxy removes the SFC header and sends the packet to the SF, with encapsulating a certain VLAN ID. It locally maintains the mapping between VLAN ID and the SFC header. When it gets the returned packet from the SF, it removes the VLAN part from the packet and retrieves the corresponding SFC header according to the VLAN ID, and then encapsulates SFC header into that packet before sending to the next service function.

The VLAN ID can be used for mapping per flow, i.e. each flow will be assigned a new VLAN ID. If SFC header could be changed per packet, the length of VLAN ID is not enough for mapping.


```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

```

Outer Ethernet Header:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               SFI Destination MAC Address                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| SF Destination MAC Address      | SFC Proxy Source MAC Address      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               SFC Proxy Source MAC Address                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|OptnlEthtype = C-Tag 802.1Q      |Outer.VLAN Tag Information      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Ethertype = 0x0800          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Original IP Payload:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Original Payload                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

[3.1.3. QinQ](#)

If the network between the SFC proxy and SF is already a VLAN network, and the SF needs to look into the MAC address, then QinQ is used for the communication between SFC proxy and SF. The SFC proxy remove the SFC header and send the original traffic to SF with a certain outer VLAN ID. It locally maintains the mapping between outer VLAN ID and the SFC header.

If the network between SFC proxy and SF is not a VLAN network, then QinQ can be used for either per flow mapping or per packet mapping, using two layer VLAN fields. If the network between SFC proxy and SF is a VLAN network, then QinQ can only be used for per flow mapping, using one VLAN field.


```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

```

Outer Ethernet Header:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               SF Destination MAC Address                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| SF Destination MAC Address      | SFC Proxy Source MAC Address      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               SFC Proxy Source MAC Address                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|OptnlEthtype = S-Tag 802.1Q      |Outer.VLAN Tag Information          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|Ethertype = C-Tag 802.1Q         |Inner.VLAN Tag Information          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Ethertype = 0x0800          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Original IP Payload:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               Original Payload                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

3.1.4. VXLAN

If the SFC proxy and SF are already deployed in a QinQ network, then VXLAN [[I-D.mahalingam-dutt-dcops-vxlan](#)] can be used for the mapping, i.e. VNI can be used for the mapping between them. This tunneling technology is only used when the original packet type is at layer 2 and the SF has to look into the layer 2 MAC header.

The drawback of this mechanism is that it requires both SFC proxy and SF to support VXLAN.


```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

```

Outer Ethernet Header:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               SF Destination MAC Address               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|SFI Destination MAC Address   | SFC Proxy Source MAC Address |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               SFC Proxy Source MAC Address             |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|OptnlEthtype = C-Tag 802.1Q   |Outer.VLAN Tag Information   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   Ethertype = 0x0800         |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Outer IP Header:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|Version|  IHL  |Type of Service|          Total Length          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Identification      |Flags|      Fragment Offset      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|Time to Live |Protocol=17(UDP) |      Header Checksum      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               Outer Source IPv4 Address                |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               Outer Destination IPv4 Address            |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Outer UDP Header:

```

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Source Port = xxxx      |      Dest Port = VXLAN Port  |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      UDP Length              |      UDP Checksum              |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

VXLAN Header:

```

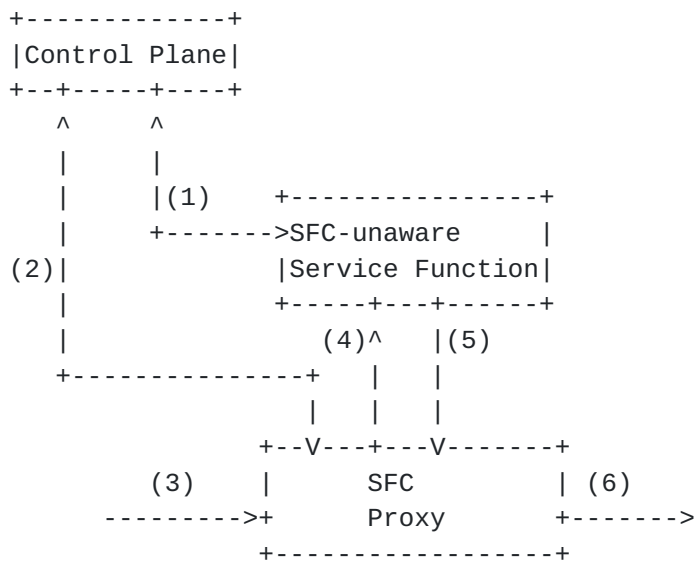
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|R|R|R|R|I|R|R|R|              Reserved              |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|              VXLAN Network Identifier (VNI) |      Reserved      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```


3.2. For Non-transparent Service Functions

Non transparent service functions including NAT (Network Address Translation), WOC (WAN Optimization Controller) and etc, are more complicated, as they may change any part of the original packet sent to them. It is better to analyze case by case, to utilize a specific field that the SF does not change for the mapping and retrieving the SFC header. We would like to leave it for open discussion.

The use case below is just one example that SFC proxy can learn the behavior of the SF changing the packet. In this example, the following method is used for SFC header mapping. The SF needs to report its mapping rules (e.g. 5-tuple mapping rules) to the control plane (step 1), and then the control plane can notify the SFC proxy the mapping information (step 2). According to the mapping information, the SFC proxy can establish a mapping table for the SFC header, the original header, and the processed header of the packet. After receiving the packet from the SF (step 5), the SFC proxy retrieves the SFC header from the mapping table by using the processed header as a key.



4. Operation Consideration

The following table shows all the methods and the conditions to use.

Table 1: Operation Consideration

	Methods	Ingress Flow Mapping	Egress Flow Mapping	Application Condition
For Trans-parent SF	MAC Address	1.5-tuple->Source MAC address 2.Any SFC packet->Source MAC address	Source MAC address->SFC header	L2 header won't be modified by the SF.
	VLAN	5-tuple->VLAN ID	VLAN ID->SFC header	L2 header won't be modified by the SF.
	QinQ	5-tuple->Outer VLAN ID	Outer VLAN ID->SFC header	The SFI is required to support QinQ. L2 header won't be modified by the SF.
	VXLAN	5-tuple->VNI	VNI->SFC header	The SFI is required to support VXLAN. L2 header won't be modified by the SF.
For Non-trans-parent SF	5-tuple Mapping	5-tuple ->5-tuple	5-tuple ->SFC header	The 5-tuple of the original packet won't be modified by the SF.
	Control Plane	e.g. 5-tuple ->5-tuple'	e.g. 5-tuple' ->SFC header	The SFC proxy must be configured or be able to obtain the mapping rules of the SF. The SF only changes the 5-tuple according to the 5-tuple mapping rules.

5. Security considerations

When the layer 2 header of the original packet is modified and sent to the SF, if the SF needs to look into the layer 2 header, it may cause security threats. It also provides diagrams of the main entities that the information model is comprised of.

6. Acknowledgement

The authors would like to thank Ron Parker for his comments.

7. Informative References

[I-D.mahalingam-dutt-dcops-vxlan]

Mahalingam, M., Dutt, D., Duda, K., Agarwal, P., Kreeger, L., Sridhar, T., Bursell, M., and C. Wright, "VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", [draft-mahalingam-dutt-dcops-vxlan-09](#) (work in progress), April 2014.

[I-D.quinn-sfc-arch]

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

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

Authors' Addresses

Haibin Song
Huawei
101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu 210012
China

Email: haibin.song@huawei.com

Jianjie You
Huawei
101 Software Avenue, Yuhuatai District
Nanjing, 210012
China

Email: youjianjie@huawei.com

Lucy Yong
Huawei
5340 Legacy Drive
Plano, TX 75025
U.S.A.

Email: lucy.yong@huawei.com

Yuanlong Jiang
Huawei
Bantian, Longgang district
Shenzhen 518129
China

Email: jiangyuanlong@huawei.com

