Service Function Chaining   D. Dolson
Internet-Draft   Sandvine
Intended status: Informational   S. Homma
Expires: January 27, 2017   NTT
                                                         D. Lopez
                                                   Telefonica I+D
                                                     M. Boucadair
                                                           Orange
                                                           D. Liu
                                                    Alibaba Group
                                                            T. Ao
                                                  ZTE Corporation
                                                            V. Vu
                                                              SSU
                                                    July 26, 2016

          Hierarchical Service Function Chaining (hSFC)
                  draft-ietf-sfc-hierarchical-00

Abstract

   Hierarchical Service Function Chaining (hSFC) is a network
   architecture allowing an organization to compartmentalize a large-
   scale network into multiple domains of administration.

   The goals of hSFC are to make a large-scale network easier to reason
   about, simpler to control and to able support independent functional
   groups within large operators.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

   Service Function Chaining (SFC) is a technique for prescribing
   differentiated traffic forwarding policies within an SFC-enabled
   domain. SFC is described in detail in the SFC architecture document
   [RFC7665], and is not repeated here.

   In this document we consider the difficult problem of implementing
   SFC across a large, geographically dispersed network comprised of
   millions of hosts and thousands of network forwarding elements,
   involving multiple operational teams (with varying functional
   responsibilities). We expect asymmetrical routing is inherent in the
   network, while recognizing that some Service Functions (SFs) require
   bidirectional traffic for transport-layer sessions (e.g., NATs,
   firewalls). We assume that some Service Function Paths (SFPs) need
   to be selected on the basis of application-specific data visible to
   the network, with transport-layer coordinate (typically, 5-tuple)
   stickiness to specific SF instances.

   Note: in this document, the notion of the "path" of a packet is the
   series of SF instances traversed by a packet. The means of
   delivering packets between SFs (the forwarding mechanisms enforced in
   the underlying network) is not relevant to the discussion.

   Difficult problems are often made easier by decomposing them in a
   hierarchical (nested) manner. So instead of considering an
   omniscient SFC Control Plane that can manage (create, withdraw,
   supervise, etc.) complete SFPs from one end of the network to the
   other, we decompose the network into smaller sub-domains. Each sub-
   domain may support a subset of the network applications or a subset
   of the users. The criteria for determining decomposition into SFC-
   enabled sub-domains are beyond the scope of this document.

   Note that decomposing a network into multiple SFC-enabled domains
   should permit end-to-end visibility of SFs and SFPs. Decomposition
   should also be implemented with special care to ease monitoring and
   troubleshooting of the network and services as a whole.

   An example of simplifying a network by using multiple SF domains is
   further discussed in [I-D.ietf-sfc-dc-use-cases].

   We assume the SFC-aware nodes use NSH [I-D.ietf-sfc-nsh] or a similar
   labeling mechanism.

   The "domains" discussed in this document are assumed to be under
   control of a single organization, such that there is a strong trust
   relationship between the domains. The intention of creating multiple
   domains is to improve the ability to operate a network. It is

outside of the scope of the document to consider domains operated by
different organizations.

[2](#). **Hierarchical Service Function Chaining (hSFC)**

A hierarchy has multiple levels. The top-most level encompasses the
entire network domain to be managed, and lower levels encompass
portions of the network.

[2.1](#). **Top Level**

Considering the example depicted in Figure 1, a top-level network
domain includes SFC data plane components distributed over a wide
area, including:

o Classifiers (CFs),

o Service Function Forwarders (SFFs) and

o Sub-domains.

For the sake of clarity, components of the underlay network are not
shown; an underlay network is assumed to provide connectivity between
SFC data plane components.

Top-level SFPs carry packets from classifiers through a series of
SFFs and sub-domains, with the operations within sub-domains being
opaque to the higher levels.

We expect the system to include a top-level control-plane having
responsibility for configuring forwarding and classification (see
[[I-D.ietf-sfc-control-plane](#)]). The top-level Service Chaining
control-plane manages end-to-end service chains and associated
service function paths from network edge points to sub-domains and
configuring top-level classifiers at a coarse level (e.g., based on
source or destination host) to forward traffic along paths that will
transit appropriate sub-domains. Figure 1 shows one possible service
chain passing from edge, through two sub-domains, to network egress.
The top-level control plane does not configure classification or
forwarding within the sub-domains.

At this network-wide level, the number of SFPs required is a linear
function of the number of ways in which a packet is required to
traverse different sub-domains and egress the network. Note that the
various paths which may be taken within a sub-domain are not
represented by distinct network-wide SFPs; specific policies at the
ingress nodes of each sub-domain bind flows to sub-domain paths.

Packets are classified at the edge of the network to select the paths
by which sub-domains are to be traversed. At the ingress of each
sub-domain, paths are reclassified to select the paths by which SFs
in the sub-domain are to be traversed. At the egress of each sub-
domain, packets are returned to the top-level paths. Contrast this
with an approach requiring the top-level classifier to select paths
to specify all of the SFs in each sub-domain.

It should be assumed that some SFs require bidirectional symmetry of
paths (see more in Section 4). Therefore the classifiers at the top
level must be configured with policies ensuring outgoing packets take
the reverse path of incoming packets through sub-domains.

```
                    +------------+
                    |Sub-domain#1|
                    |  in DC1    |
                    +----+-------+
                         |
                    .---- SFF1 ------.    +--+
         +--+    /       /  |          \--|CF|
     --->|CF|--/---->'      |            \ +--+
         +--+ /   SC#1      |             \
              |             |             |
              |             V      .------>|--->
              |            /   /          |
               \          |   /          /
         +--+  \          |  /          /   +--+
         |CF|---\         | /          /---|CF|
         +--+    '---- SFF2 ------'      +--+
                         |
                    +----+-------+
                    |Sub-domain#2|
                    |   in DC2   |
                    +------------+
```
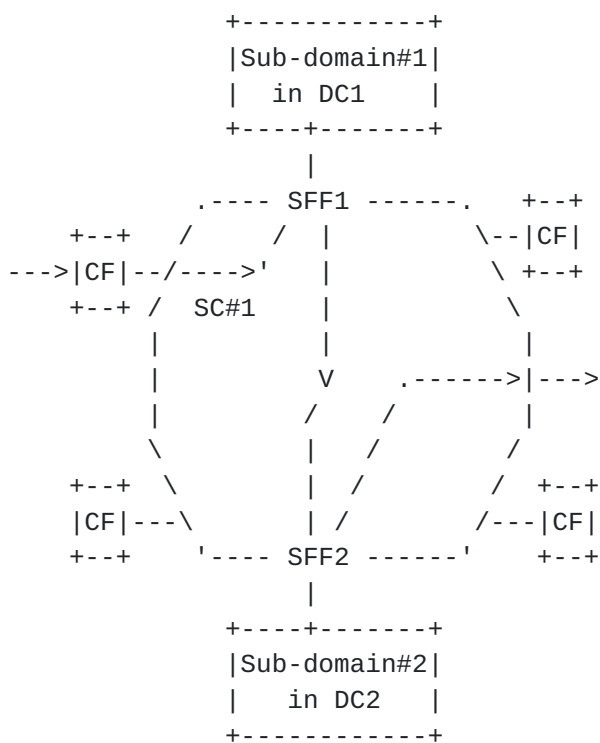
One path is shown from edge classifier to SFF1 to Sub-domain#1
(residing in data-center1) to SFF1 to SFF2 (residing in data-center
2) to Sub-domain#2 to SFF2 to network egress.

           Figure 1: Network-wide view of top level of hierarchy
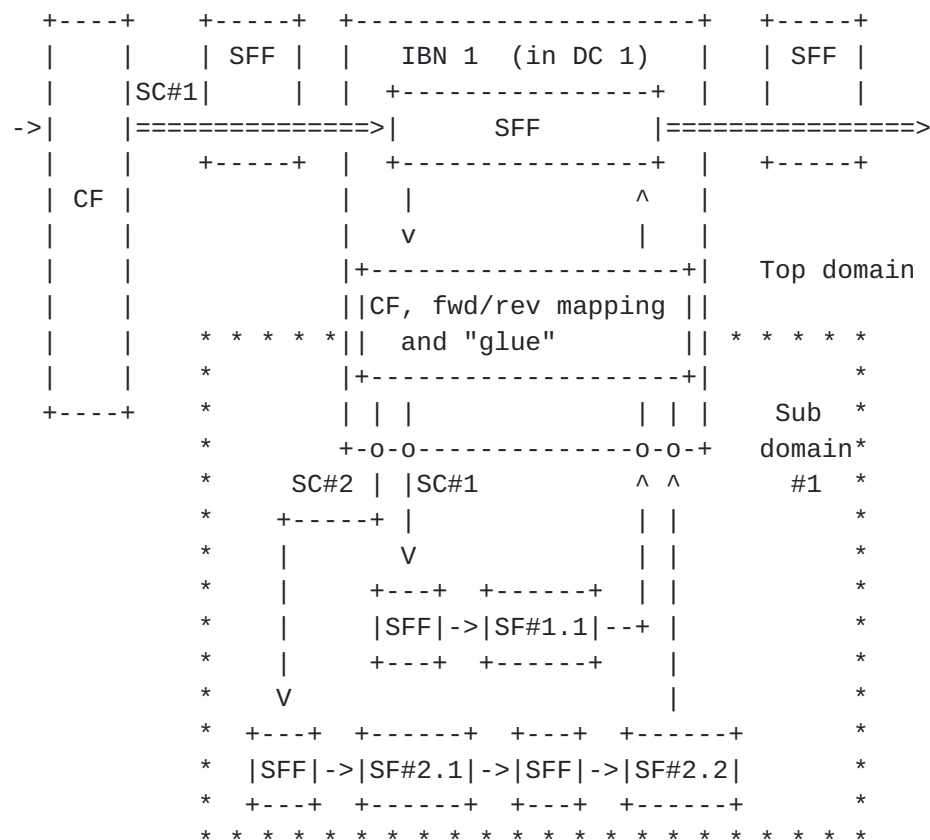
2.2. Lower Levels

Each of the sub-domains in Figure 1 is an SFC-enabled domain.

Unlike the top level, data packets entering the sub-domain are
already SFC-encapsulated. Figure 2 shows a sub-domain interfaced
with a higher-level domain by means of an Internal Boundary Node

(IBN). It is the purpose of the IBN to apply classification rules
and direct the packets to the selected local SFPs terminating at an
egress IBN. The egress IBN finally restores packets to the original
SFC shim and hands them off to SFFs.

Each sub-domain intersects a subset of the total paths that are
possible in the higher-level domain. An IBN is concerned with
higher-level paths, but only those traversing its sub-domain. A top-
level control element may configure the IBN as an SF (i.e., the IBN
plays the SF role in the top-level domain).

Each sub-domain is likely to have a control-plane that can operate
independently of the top-level control-plane. The sub-domain
control-plane configures the classification and forwarding rules in
the sub-domain. The classification rules reside in the IBN, where
SFC encapsulation of the top-level domain is converted to/from SFC
encapsulation of the lower-level domain.

```
   +----+    +-----+ +--------------------+   +-----+
   |    |    | SFF | |   IBN 1  (in DC 1)  |   | SFF |
   |    |SC#1|     | | +----------------+  |   |     |
->|    |===============>|      SFF        |===============>
   |    |    +-----+ |  +----------------+ |   +-----+
   | CF |            |  |                ^  |
   |    |            |  v                |  |
   |    |            |+--------------------+|  Top domain
   |    |            ||CF, fwd/rev mapping ||
   |    |    * * * * *||   and "glue"       || * * * * *
   |    |    *       |+--------------------+|        *
   +----+    *       | | |             | | |   Sub  *
             *       +-o-o-------------o-o-+  domain*
             *       SC#2 | |SC#1       ^ ^       #1 *
             *      +-----+ |           | |          *
             *      |       V           | |          *
             *      |     +---+  +------+| |          *
             *      |     |SFF|->|SF#1.1|--+ |        *
             *      |     +---+  +------+    |        *
             *      V                        |        *
             *   +---+  +------+  +---+  +------+      *
             *   |SFF|->|SF#2.1|->|SFF|->|SF#2.2|      *
             *   +---+  +------+  +---+  +------+      *
             * * * * * * * * * * * * * * * * * * * *
```

*** Sub-domain boundary; === top-level chain; --- low-level chain.

                Figure 2: Sub-domain within a higher-level domain

If desired, the pattern can be applied recursively. For example, SF#1.1 in Figure 2 could be a sub-domain of the sub-domain.

3. Internal Boundary Node (IBN)

A network element termed "Internal Boundary Node" (IBN) bridges packets between domains. It behaves as an SF to the higher level, and looks like a classifier and end-of-chain to the lower level.

To achieve the benefits of hierarchy, the IBN should be applying more granular traffic classification rules at the lower level than the traffic passed to it. This means that the number of SFPs within the lower level is greater than the number of SFPs arriving to the IBN.

The IBN is also the termination of lower-level SFPs. This is because the packets exiting lower-level SF paths must be returned to the higher-level SF paths and forwarded to the next hop in the higher-level domain.

3.1. IBN Path Configuration

An operator of a lower-level domain may be aware of which high-level paths transit their domain, or they may wish to accept any paths.

When packets enter the sub-domain, the Service Path Identifier (SPI) and Service Index (SI) are re-marked according to the path selected by the classifier.

After exiting a path in the sub-domain, packets can be restored to an original upper-level SFP by these methods:

1. Saving SPI and SI in transport-layer flow state,

2. Pushing SPI and SI into metadata,

3. Using unique lower-level paths per upper-level path coordinates,

4. Nesting NSH headers, encapsulating the higher-level NSH headers within the lower-level NSH headers,

5. Saving upper-level by a flow ID and placing an hSFC flow ID into metadata,

3.1.1. Flow-Stateful IBN

An IBN can be flow-aware, returning packets to the correct higher-level SFP on the basis of the transport-layer coordinates (typically, a 5-tuple) of packets exiting the lower-level SFPs.

When packets are received by the IBN on a higher-level path, the
encapsulated packets are parsed for IP and transport-layer (TCP, UDP,
etc.) coordinates. State is created, indexed by these coordinates
({source-IP, destination-IP, source-port, destination-port and
transport protocol} typically). The state contains at least critical
fields of the encapsulating SFC header (or perhaps the entire
header).

The simplest approach has the packets return to the same IBN at the
end of the chain that classified the packet at the start of the
chain. This is because the required transport-coordinates state is
rapidly changing and most efficiently kept locally. If the packet is
returned to a different IBN for egress, transport-coordinates state
must be synchronized between the IBNs.

When a packet returns to the IBN at the end of a chain, the SFC
header is removed, the packet is parsed for IP and transport-layer
coordinates, and state is retrieved from them. The state contains
the information required to forward the packet within the higher-
level service chain.

State cannot be created by packets arriving from the lower-level
chain; when state cannot be found for such packets, they must be
dropped.

This stateful approach is limited to use with SFs that retain the
transport coordinates of the packet. This approach cannot be used
with SFs that modify those coordinates (e.g., NATs) or otherwise
create packets for new coordinates other than those received (e.g.,
as an HTTP cache might do to retrieve content on behalf of the
original flow). In both cases, the fundamental problem is the
inability to forward packets when state cannot be found for the
packet transport-layer coordinates.

In the stateful approach, there are issues caused by having state,
such as how long the state should be maintained (it must time out
eventually), as well as whether the state needs to be replicated to
other devices to create a highly available network.

It is valid to consider the state to be disposable after failure,
since it can be re-created by each new packet arriving from the
higher-level domain. For example, if an IBN loses all flow state,
the state is re-created by an end-point retransmitting a TCP packet.

If an SFC domain handles multiple network regions (e.g., multiple
private networks), the coordinates may be augmented with additional
parameters, perhaps using some metadata to identify the network
region.

In this stateful approach, it is not necessary for the sub-domain's
control-plane to modify paths when higher-level paths are changed.
The complexity of the higher-level domain does not cause complexity
in the lower-level domain.

Since it doesn't depend on NSH in the lower domain, this flow-
stateful approach can be applied to translation methods of converting
NSH to other forwarding techniques. (Refer to Section 6.)

3.1.2. Encoding Upper-Level Paths in Metadata

An IBN can push the upper-level Service Path Identifier (SPI) and
Service Index (SI) (or encoding thereof) into a metadata field of the
lower-level encapsulation (e.g., placing upper-level path information
into a metadata field of NSH). When packets exit the lower-level
path, the upper-level SPI and SI can be restored from the metadata
retrieved from the packet.

This approach requires the SFs in the path to be capable of
forwarding the metadata and appropriately attaching metadata to any
packets injected for a flow.

Using new metadata may inflate packet size when variable-length
metadata (type 2 from NSH [I-D.ietf-sfc-nsh]) is used.

It is conceivable that the MD-type 1 Mandatory Context Header fields
of NSH [I-D.ietf-sfc-nsh] are not all relevant to the lower-level
domain. In this case, one of the metadata slots of the Mandatory
Context Header could be repurposed within the lower-level domain, and
restored when leaving.

In this metadata approach, it is not necessary for the sub-domain's
control element to modify paths when higher-level paths are changed.
The complexity of the higher-level domain does not cause complexity
in the lower-level domain.

3.1.3. Using Unique Paths per Upper-Level Path

In this approach, paths within the sub-domain are constrained so that
a SPI (of the sub-domain) unambiguously indicates the egress SPI and
SI (of the upper domain). This allows the original path information
to be restored at sub-domain egress from a look-up table using the
sub-domain SPI.

Whenever the upper-level domain provisions a path via the lower-level
domain, the lower-level domain controller must provision
corresponding paths to traverse the lower-level domain.

A down-side of this approach is that the number of paths in the
lower-level domain is multiplied by the number of paths in the
higher-level domain that traverse the lower-level domain. I.e., a
sub-path must be created for each combination of upper SPI/SI and
lower chain.

3.1.4. Nesting Upper-Level NSH within Lower-Level NSH

In this approach, when packets arrive at the IBN in the top-level
domain, the classifier in the IBN determines the path for the lower-
level domain and pushes the new NSH header in front of the original
NSH header.

As shown in Figure 3 the Lower-NSH Header used to forward packets in
the lower-level domain precedes the Upper-NSH Header from the top-
level domain.

```
                    +------------------+
                    | Overlay Header   |
                    +------------------+
                    | Lower-NSH Header |
                    +------------------+
                    | Upper-NSH Header |
                    +------------------+
                    | Original Packet  |
                    +------------------+
```

Figure 3: Encapsulation of NSH within NSH

The traffic with the above stack of two-layer-NSH header is to be
forwarded according to the Lower-NSH header in the lower-level SFC
domain. The Upper-NSH header is preserved in the packets but not
used for forwarding. At the last SFF of the chain of the lower-level
domain (which resides in the IBN), the Lower-NSH header is removed
from the packet, and then the packet is forwarded by the IBN to an
SFF of the upper-level domain, which will be forwarded according to
the Upper-NSH header.

With such encapsulation, Upper-NSH information is carried along the
extent of the lower-level chain without modification.

A benefit of this approach is that it does not require state in the
IBN or configuration to encode fields in meta-data.

However, the down-side is it does require SFs in the lower-level
domain to be able to parse multiple layers of NSH. If the SF injects

packets, it must also be able to deal with adding appropriate
multiple layers of headers to injected packets.

3.1.5. Stateful / Metadata Hybrid

The basic idea of this approach is for the IBN to save upper domain
encapsulation information such that it can be retrieved by a unique
identifier, termed an "hSFC Flow ID". An example ID is shown in
Table 1.

```
+-----------+-----+-----+----------+----------+----------+----------+
| hSFC Flow | SPI | SI  | Context1 | Context2 | Context3 | Context4 |
| ID        |     |     |          |          |          |          |
+-----------+-----+-----+----------+----------+----------+----------+
| 1         | 45  | 254 | 100      | 2112     | 12345    | 7        |
+-----------+-----+-----+----------+----------+----------+----------+
```

   Table 1: Example Mapping of an hSFC Flow ID to Upper-Level Header

The ID is placed in the metadata in NSH headers of the packet in the
lower domain, as shown in Figure 4. When packets exit the lower
domain, the IBN uses the ID to retrieve the appropriate NSH
encapsulation for returning the packet to the upper domain.

```
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |Ver|O|C|R|R|R|R|R|R|   Length  | MD-type=0x1  | Next Protocol |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Service Path Identifer           | Service Index |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      hSFC Flow ID                           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                 Mandatory Context Header                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                 Mandatory Context Header                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                 Mandatory Context Header                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

      Figure 4: Storing hSFC Flow ID in lower-level metadata

Advantages of this approach include:

o Does not require state based on 5-tuple, so it works with
   functions that change the IP addresses or ports of a packet such
   as NATs,

   o Does not require all domains to have the same metadata scheme,

   o Can be used to restore any upper-domain information, not just
      service path,

   o The lower domain only requires a single item of metadata
      regardless of the number of items of metadata used in the upper
      domain. (For MD-Type 1, this leaves 3 slots for use in the lower
      domain.)

   o No special functionality is required of the SF, other than the
      usual ability to preserve metadata and to apply metadata to
      injected packets.

   Disadvantages include those of other stateful approaches, including
   state timeout and replication mentioned in Section 3.1.1.

   There may be a large number of unique NSH encapsulations to be
   stored, given that the hSFC Flow ID must represent all of the bits in
   the upper-level encapsulation. This might consume a lot of memory or
   create out-of-memory situations in which IDs cannot be created or old
   IDs are discarded while still in use.

3.2. Gluing Levels Together

   The SPI or metadata on a packet received by the IBN may be used as
   input to reclassification and path selection within the lower-level
   domain.

   In some cases the meanings of the various path IDs and metadata must
   be coordinated between domains.

   One approach is to use well-known identifier values in metadata,
   communicated by some organizational registry.

   Another approach is to use well-known labels for chain identifiers or
   metadata, as an indirection to the actual identifiers. The actual
   identifiers can be assigned by control-plane systems. For example, a
   sub-domain classifier could have a policy, "if pathID=classA then
   chain packet to path 1234"; the higher-level controller would be
   expected to configure the concrete higher-level pathID for classA.

3.3. Decrementing Service Index

   Because the IBN acts as a Service Function to the higher-level
   domain, it must decrement the Service Index in the NSH headers of the
   higher-level path.

A good strategy seems to be to do this when the packet is first
received by the IBN, before applying any of the strategies of
Section 3.1, immediately prior to classification.

4. Sub-domain Classifier

Within the sub-domain (referring to Figure 2), after the IBN removes
higher-level encapsulation from incoming packets, it sends the
packets to the classifier, which selects the encapsulation for the
packet within the sub-domain.

One of the goals of the hierarchical approach is to make it easy to
have transport-flow-aware service chaining with bidirectional paths.
For example, it is desired that for each TCP flow, the client-to-
server packets traverse the same SFs as the server-to-client packets,
but in the opposite sequence. We call this bidirectional symmetry.
If bidirectional symmetry is required, it is the responsibility of
the control-plane to be aware of symmetric paths and configure the
classifier to chain the traffic in a symmetric manner.

Another goal of the hierarchical approach is to simplify the
mechanisms of scaling in and scaling out service functions. All of
the complexities of load-balancing among multiple SFs can be handled
within a sub-domain, under control of the classifier, allowing the
higher-level domain to be oblivious to the existence of multiple SF
instances.

Considering the requirements of bidirectional symmetry and load-
balancing, it is useful to have all packets entering a sub-domain to
be received by the same classifier or a coordinated cluster of
classifiers. There are both stateful and stateless approaches to
ensuring bidirectional symmetry.

5. Control Plane Elements

Although control protocols have not yet been standardized, from the
point of view of hierarchical service function chaining we have these
expectations:

o Each control-plane instance manages a single level of hierarchy of
   a single domain.

o Each control-plane is agnostic about other levels of hierarchy.
   This aspect allows humans to reason about the system within a
   single domain and allows control-plane algorithms to use only
   domain-local inputs. Top-level control does not need visibility
   to sub-domain policies, nor does sub-domain control need
   visibility to higher-level policies.

   o Sub-domain control-planes are agnostic about control-planes of
      other sub-domains. This allows both humans and machines to
      manipulate sub-domain policy without considering policies of other
      domains.

   Recall that the IBN acts as an SF in the higher-level domain
   (receiving SF instructions from the higher-level control-plane) and
   as a classifier in the lower-level domain (receiving classification
   rules from the sub-domain control-plane). In this view, it is the
   IBN that glues the layers together.

   The above expectations are not intended to prohibit network-wide
   control. A control hierarchy can be envisaged to distribute
   information and instructions to multiple domains and sub-domains.
   Control hierarchy is outside the scope of this document.

[6](). Extension for Adopting to NSH-Unaware Service Functions

   The hierarchical approach can be used for dividing networks into NSH-
   aware and NSH-unaware domains by converting NSH encapsulation to
   other forwarding techniques (e.g., 5-tuple-based routing with
   OpenFlow), as shown in Figure 5.

```
                    * * * * * * * * * * * * * * * * * *
              *    NSH-aware domain                    *
              *         +-------+        +-------+      *
              *         | SF#1  |        | SF#5  |      *
              *         +-o---o-+        +-o---o-+      *
              *          ^    |          ^    |        *
              *         +-|---|-+        +-|---|-+      *
              *         | |SFF| |        | |SFF| |      *
              *         +-|---|-+        +-|---|-+      *
              *          .    |          |    .         *
              * +--+    /     |          |     \        *
              -->|CF|--'      |          |      '------->
              * +--+          v          |             *
              *         +---o-----------o---+          *
               .*.*.*.*.|  / |   IBN   | \ |*.*.*.
               .        +-o--o---------o--o-+         .
               .          |  |          ^  ^          .
               .          |  +-+        +-+  |         .
               .        +---+    v      |    +---+     .
               .        |      +-o-----o-+   |    |    .
               .        |      | SF#2   |    |    |    .
               .        |      +---------+   |    |    .
               .        +--+                 +--+      .
               .         |   +---------+      |        .
               .         v   |         v      |        .
               .        +-o---o-+    +-o---o-+          .
               .        | SF#3  |    | SF#4  |          .
               .        +-------+    +-------+          .
               .    NSH-unaware domain                 .
               . . . . . . . . . . . . . . . . . .
```
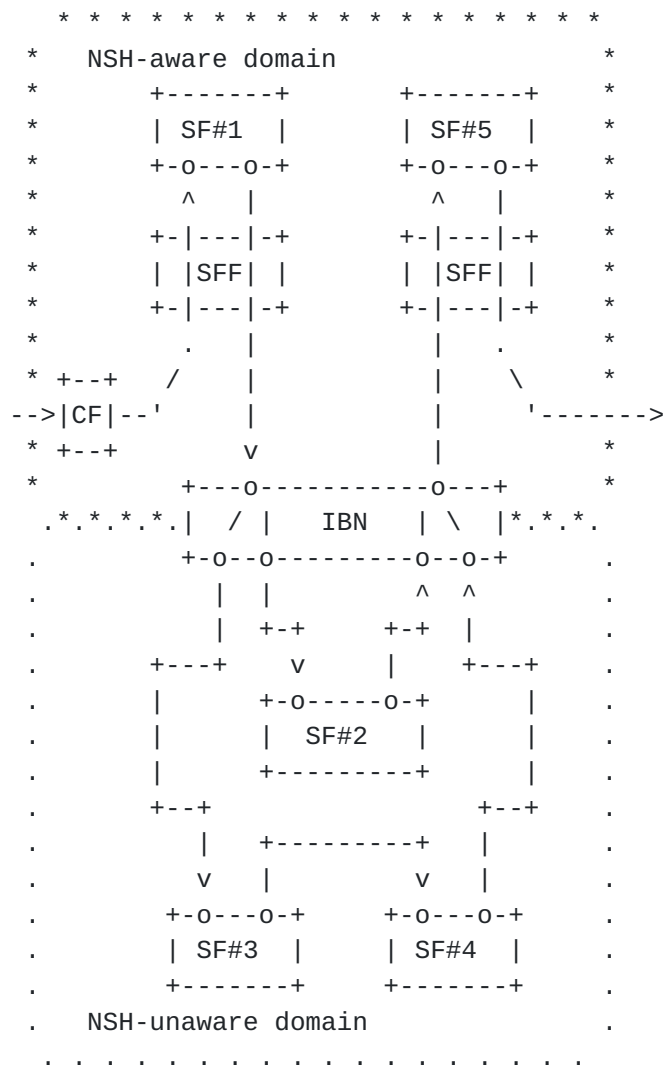
   SF#1 and SF#5 are NSH-aware and SF#2, SF#3 and SF#4 are NSH-unaware.
   In the NSH-unaware domain, packets are conveyed in a format supported
   by SFs which are deployed there.

           Figure 5: Dividing NSH-aware and NSH-unaware domains

6.1. Purpose

   This approach is expected to facilitate service chaining in networks
   in which NSH-aware and NSH-unaware SFs coexist. Some examples of
   such situations are:

   o In a period of transition from legacy SFs to NSH-aware SFs and

   o Supporting multi-tenancy.

6.2. Requirements for IBN

In this usage, an IBN classifier is required to have an NSH
conversion table for applying packets to appropriate lower-level
paths and returning packets to the correct higher-level paths. For
example, the following methods would be used for saving/restoring
upper-level path information:

o Saving SPI and SI in transport-layer flow state (refer to
   Section 3.1.1) and

o Using unique lower-level paths per upper-level NSH coordinates
   (refer to Section 3.1.3).

Especially, the use of unique paths approach would be good for
translating NSH to a different forwarding technique in the lower
level. A single path in the upper level may be branched to multiple
paths in the lower level such that any lower-level path is only used
by one upper-level path. This allows unambiguous restoration to the
upper-level path.

In addition, an IBN might be required to convert metadata contained
in NSH to the format appropriate to the packet in the lower-level
path. For example, some legacy SFs identify subscriber based on
information of network topology, such as VID, and IBN would be
required to create VLAN to packets from metadata if subscriber
identifier is conveyed as metadata in higher-level domains.

Other fundamental functions required as IBN (e.g., maintaining
metadata of upper level or decrementing Service Index) are same as
normal usage.

7. Acknowledgements

The concept of Hierarchical Service Path Domains was introduced in
[I-D.homma-sfc-forwarding-methods-analysis] as a means to improve
scalability of service chaining in large networks.

The concept of nested NSH headers was introduced in
[I-D.ao-sfc-for-dc-interconnect] as a means of creating hierarchical
SFC in a data center.

The authors would like to thank the following individuals for
providing valuable feedback:

Ron Parker

Christian Jacquenet

Jie Cao

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

Hierarchical service function chaining makes use of service chaining architecture, and hence inherits the security considerations described in the architecture document.

Furthermore, hierarchical service function chaining inherits security considerations of the data-plane protocols (e.g., NSH) and control-plane protocols used to realize the solution.

The systems described in this document bear responsibility for forwarding internet traffic. In some cases the systems are responsible for maintaining separation of traffic in private networks.

This document describes systems within different domains of administration that must have consistent configurations in order to properly forward traffic and to maintain private network separation. Any protocol designed to distribute the configurations must be secure from tampering.

All of the systems and protocols must be secure from modification by untrusted agents.

Security considerations related to the control plane are discussed in [I-D.ietf-sfc-control-plane].

10. References

10.1. Normative References

[I-D.ietf-sfc-control-plane]
          Boucadair, M., "Service Function Chaining (SFC) Control
          Plane Components & Requirements", draft-ietf-sfc-control-
          plane-06 (work in progress), May 2016.

[I-D.ietf-sfc-nsh]
          Quinn, P. and U. Elzur, "Network Service Header", draft-
          ietf-sfc-nsh-05 (work in progress), May 2016.

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

10.2. Informative References

   [I-D.ao-sfc-for-dc-interconnect]
              Ao, T. and W. Bo, "Hierarchical SFC for DC
              Interconnection", draft-ao-sfc-for-dc-interconnect-01
              (work in progress), October 2015.

   [I-D.homma-sfc-forwarding-methods-analysis]
              Homma, S., Naito, K., Lopez, D., Stiemerling, M., Dolson,
              D., Gorbunov, A., Leymann, N., Bottorff, P., and d.
              don.fedyk@hpe.com, "Analysis on Forwarding Methods for
              Service Chaining", draft-homma-sfc-forwarding-methods-
              analysis-05 (work in progress), January 2016.

   [I-D.ietf-sfc-dc-use-cases]
              Surendra, S., Tufail, M., Majee, S., Captari, C., and S.
              Homma, "Service Function Chaining Use Cases In Data
              Centers", draft-ietf-sfc-dc-use-cases-02 (work in
              progress), January 2015.

Appendix A. Examples of Hierarchical Service Function Chaining

   The advantage of hierarchical service function chaining compared with
   normal or flat service function chaining is that it can reduce the
   management complexity significantly. This section discusses examples
   that show those advantages.

A.1. Reducing the Number of Service Function Paths

   In this case, hierarchical service function chaining is used to
   simplify service function chaining management by reducing the number
   of Service Function Paths.

   As shown in Figure 6, there are two domains, each with different
   concerns: a Security Domain that selects Service Functions based on
   network conditions and an Optimization Domain that selects Service
   Functions based on traffic protocol.

   In this example there are five security functions deployed in the
   Security Domain. The Security Domain operator wants to enforce the
   five different security policies, and the Optimization Domain
   operator wants to apply different optimizations (either cache or
   video optimization) to each of these two types of traffic. If we use

flat SFC (normal branching), 10 SFPs are needed in each domain. In
contrast, if we use hierarchical SFC, only 5 SFPs in Security Domain
and 2 SFPs in Optimization Domain will be required, as shown in
Figure 7.

In the flat model, the number of SFPs is the product of the number of
functions in all of the domains. In the hSFC model, the number of
SFPs is the sum of the number of functions. For example, adding a
"bypass" path in the Optimization Domain would cause the flat model
to require 15 paths (5 more), but cause the hSFC model to require one
more path in the Optimization Domain.

```
          . . . . . . . . . . . .   . . . . . . . . . . . .
          . Security Domain     .   .  Optimization Domain  .
          .                     .   .                       .
          .    +-1---[      ]------------------>[Cache  ]------->
          .    |      [ WAF ]    .   .                       .
          .    +-2-->[      ]---------------->[Video Opt.]---->
          .    |                 .   .                       .
          .    +-3---[Anti ]----------------->[Cache  ]------->
          .    |      [Virus]    .   .                       .
          .    +-4-->[      ]--------------->[Video Opt.]---->
          .    |                 .   .                       .
          .    +-5-->[      ]--------------->[Cache  ]------->
 [DPI]--->[CF]---|      [ IPS ]    .   .                       .
          .    +-6-->[      ]--------------->[Video Opt.]---->
          .    |                 .   .                       .
          .    +-7-->[      ]----------------->[Cache  ]------->
          .    |      [ IDS ]    .   .                       .
          .    +-8-->[      ]--------------->[Video Opt.]---->
          .    |                 .   .                       .
          .    +-9-->[Traffic]---------------->[Cache  ]------->
          .    |      [Monitor]  .   .                       .
          .    +-10->[       ]-------------->[Video Opt.]---->
          . . . . . . . . . . . .   . . . . . . . . . . . .
```
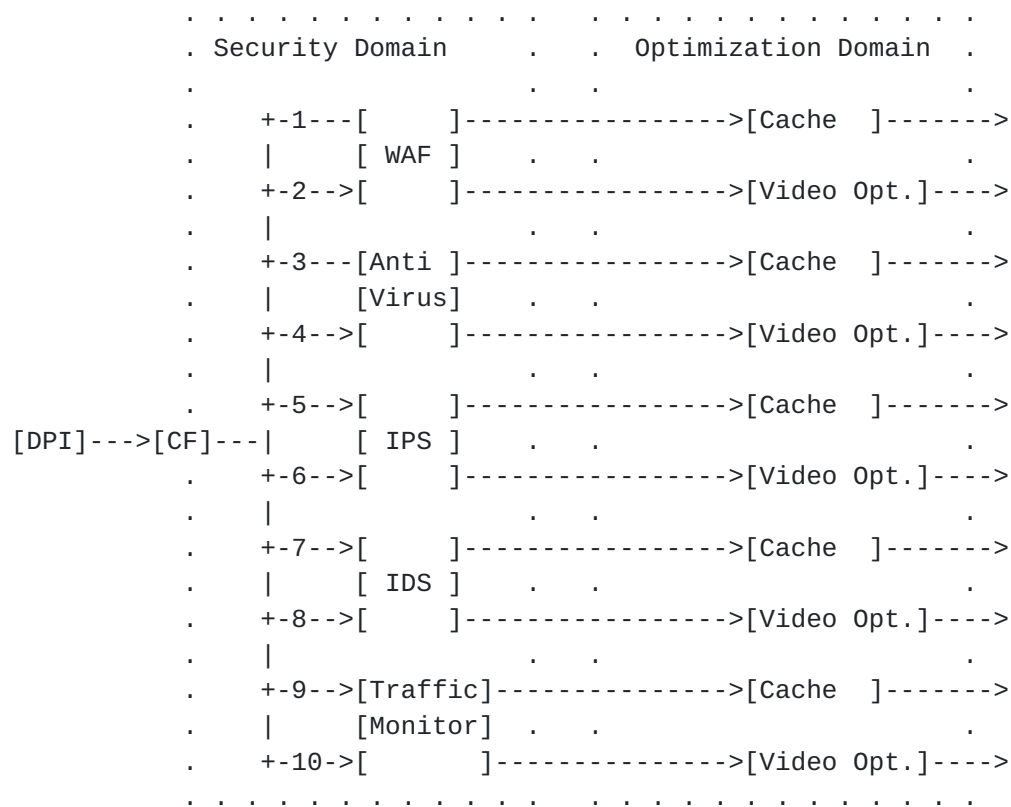
The classifier must select paths that determine the combination of
Security and Optimization concerns. 1:WAF+Cache, 2:WAF+VideoOpt,
3:AntiVirus+Cache, 4:AntiVirus+VideoOpt, 5: IPS+Cache,
6:IPS+VideoOpt, 7:IDS+Cache, 8:IDS+VideoOpt, 9:TrafficMonitor+Cache,
10:TrafficMonitor+VideoOpt

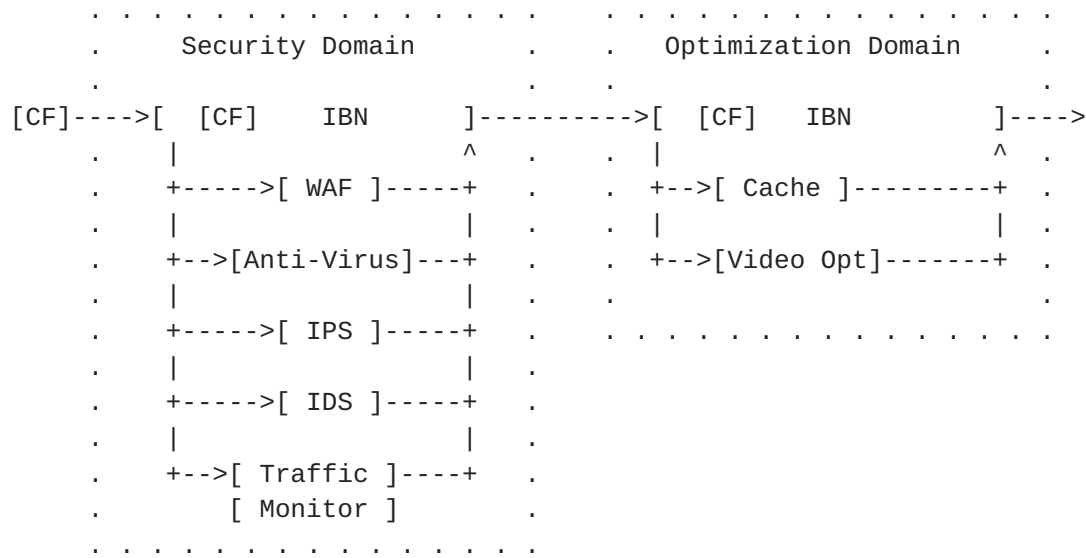Figure 6: Flat SFC (normal branching)

```
     . . . . . . . . . . . . . .     . . . . . . . . . . . . . . .
     .        Security Domain     .   .    Optimization Domain     .
     .                            .   .                            .
   [CF]---->[  [CF]    IBN     ]---------->[  [CF]    IBN        ]---->
     .      |                  ^  .   .  |                    ^  .
     .      +----->[ WAF ]-----+  .   .  +-->[ Cache ]---------+  .
     .      |                  |  .   .  |                     |  .
     .      +-->[Anti-Virus]---+  .   .  +-->[Video Opt]-------+  .
     .      |                  |  .   .                           .
     .      +----->[ IPS ]-----+  .   . . . . . . . . . . . . . .
     .      |                  |  .
     .      +----->[ IDS ]-----+  .
     .      |                  |  .
     .      +-->[ Traffic ]----+  .
     .          [ Monitor ]       .
     . . . . . . . . . . . . . .
```

           Figure 7: Simplified path management with Hierarchical SFC

A.2. Managing a Distributed Data-Center Network

   Hierarchical service function chaining can be used to simplify inter-
   data-center SFC management. In the example of Figure 8, shown below,
   there is a central data center (Central DC) and multiple local data
   centers (Local DC#1, #2, #3) that are deployed in a geographically
   distributed manner. All of the data centers are under a single
   administrative domain.

   The central DC may have some service functions that the local DC
   needs, such that the local DC needs to chain traffic via the central
   DC. This could be because:

   o Some service functions are deployed as dedicated hardware
      appliances, and there is a desire to lower the cost (both CAPEX
      and OPEX) of deploying such service functions in all data centers.

   o Some service functions are being trialed, introduced or otherwise
      handle a relatively small amount of traffic. It may be cheaper to
      manage these service functions in a single central data center and
      steer packets to the central data center than to manage these
      service functions in all data centers.

```
                   +-----------+
                   |Central DC |
                   +-----------+
                      ^   ^    ^
                      |   |    |
                    .---|--|---|----.
                   /   /   |   |     \
                  /   /    |    \     \
   +-----+       /   /     |     \     \    +-----+
   |Local|      |   /      |      \     |   |Local|
   |DC#1 |--|--.         |         .----|----|DC#3 |
   +-----+  |           |            |   +-----+
             \          |           /
              \         |          /
               \        |         /
             '----------------'
                      |
                 +-----+
                 |Local|
                 |DC#2 |
                 +-----+
```
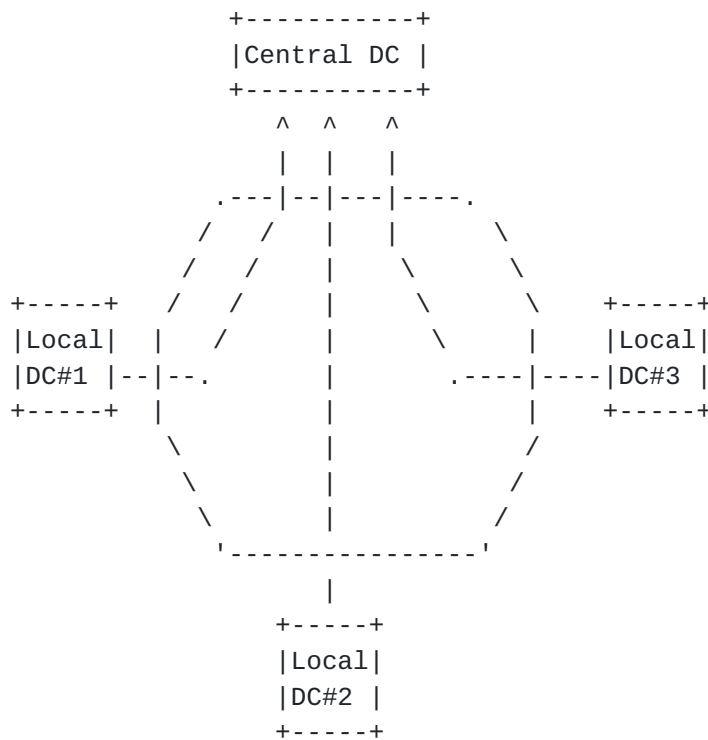
                Figure 8: Simplify inter-DC SFC management

   For large data center operators, one local DC may have tens of
   thousands of servers and hundred of thousands of virtual machines.
   SFC can be used to manage user traffic. For example, SFC can be used
   to classify user traffic based on service type, DDoS state etc.

   In such large scale data center, using flat SFC is very complex,
   requiring a super-controller to configure all data centers. For
   example, any changes to Service Functions or Service Function Paths
   in the central DC (e.g., deploying a new SF) would require updates to
   all of the Service Function Paths in the local DCs accordingly.
   Furthermore, requirements for symmetric paths add additional
   complexity when flat SFC is used in this scenario.

   Conversely, if using hierarchical SFC, each data center can be
   managed independently to significantly reduce management complexity.
   Service Function Paths between data centers can represent abstract
   notions without regard to details within data centers. Independent
   controllers can be used for the top level (getting packets to pass
   the correct data centers) and local levels (getting packets to
   specific SF instances).

Authors' Addresses

   David Dolson
   Sandvine
   408 Albert Street
   Waterloo, ON N2L 3V3
   Canada

   Phone: +1 519 880 2400
   Email: ddolson@sandvine.com

   Shunsuke Homma
   NTT, Corp.
   3-9-11, Midori-cho
   Musashino-shi, Tokyo 180-8585
   Japan

   Email: homma.shunsuke@lab.ntt.co.jp

   Diego R. Lopez
   Telefonica I+D
   Don Ramon de la Cruz, 82
   Madrid 28006
   Spain

   Phone: +34 913 129 041
   Email: diego.r.lopez@telefonica.com

   Mohamed Boucadair
   Orange
   Rennes 35000
   France

   Email: mohamed.boucadair@orange.com

   Dapeng Liu
   Alibaba Group
   Beijing 100022
   China

   Email: max.ldp@alibaba-inc.com

Ting Ao
ZTE Corporation
No.889,Bibo Rd.,Zhangjiang Hi-tech Park
Shanghai   201203
China

Phone: +86-21-688976442
Email: ao.ting@zte.com.cn

Vu Anh Vu
Soongsil University
369 Sangdo-ro
Seoul, Dongjak-gu 06978
Korea

Email: vuva@dcn.ssu.ac.kr