Hierarchical Service Function Chaining
draft-dolson-sfc-hierarchical-03

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 support independent functional
   groups within large operators.

Status of This Memo

   This Internet-Draft is submitted in full conformance with the
   provisions of BCP 78 and BCP 79.

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

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

   This Internet-Draft will expire on April 4, 2016.

Copyright Notice

Table of Contents

1. **Introduction**

   Service Function Chaining (SFC) is a technique for prescribing
   differentiated traffic forwarding policies within the SFC domain.
   SFC is described in detail in the SFC architecture document
   [I-D.ietf-sfc-architecture], 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 paths 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 Service Function 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 of the
underlay network) is not relevant to the current 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 paths 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 Service Functions and Service
Function Paths. Decomposition should also be implemented with
special care to ease monitoring and troubleshooting of the network 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 SF technology uses 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 here 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.

1.1. 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 RFC 2119 [RFC2119].

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 in Figure 1, a top-level network domain includes SFC 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 components.

Top-level service function paths 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. 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. The figure 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 service functions in the network
require bidirectional symmetry of paths (see more in Section 4).
Therefore the classifiers at the top level must be configured with
policies ensuring server-to-client packets take the reverse path of
client-to-server packet through sub-domains. (Recall the "path"
denotes the series of service functions; the precise physical network
path within the underlay network is not relevant here.)

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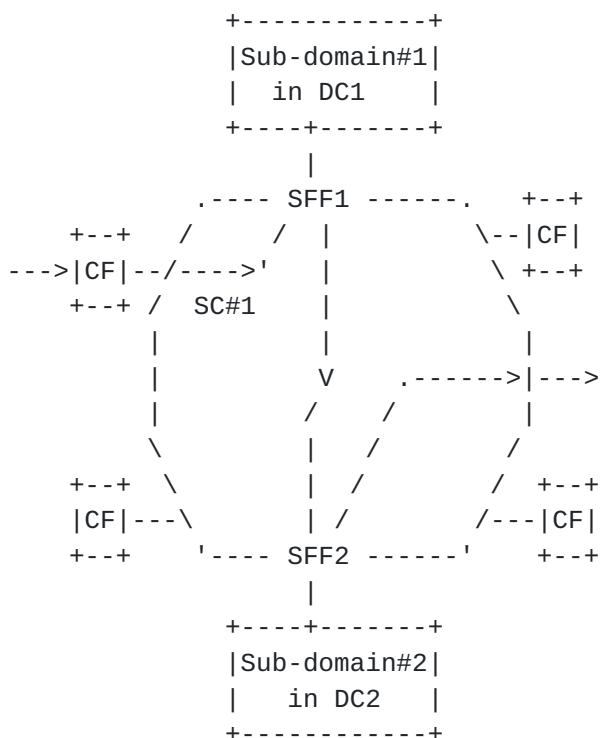
One path is shown from edge classifier to SFF1 to Sub-domain#1
(residing in data-center1) to SFF1 to SFF2 (residing in data-center
2) to Sub-domain#2 to SFF2 to network egress.

Figure 1: Network-wide view of Top Level of Hierarchy

2.2. Lower Levels

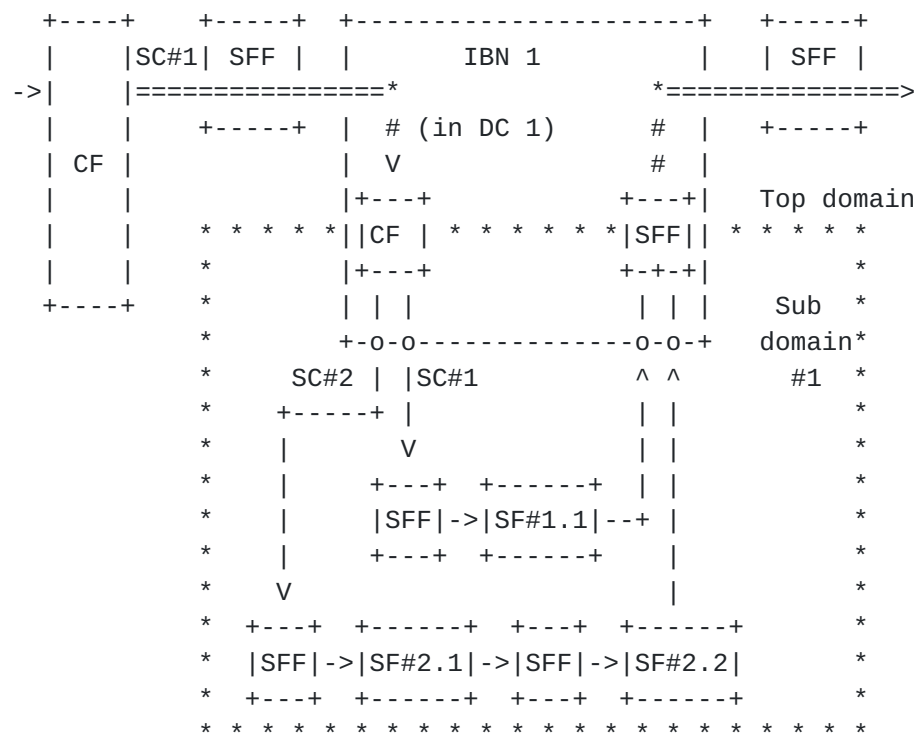Each of the sub-domains in Figure 1 is an SFC domain.

Unlike the top level, however, data packets entering the sub-domain
are already encapsulated within SFC transport. 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 remove packets
from the SFC encapsulation, apply Classification rules, and direct

the packets to the selected local service function paths terminating
at an egress IBN. The egress SFC Domain Gateway finally restores
packets to the original SFC transport 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 the sub-domain. A top-
level controller may configure the IBN as an SF (the IBN plays the SF
role in the top-level domain).

We expect each sub-domain 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
packets are moved from SFC encapsulation of the top-level domain to
and from SFC encapsulation of the lower-level domain.

```
   +----+    +-----+  +----------------------+   +-----+
   |    |SC#1| SFF |  |         IBN 1         |   | SFF |
 ->|    |================*                 *===============>
   |    |    +-----+  |   # (in DC 1)       #  |   +-----+
   | CF |             |   V                 #  |
   |    |             |+---+             +---+|   Top domain
   |    |    * * * * *||CF | * * * * * *|SFF|| * * * * *
   |    |    *        |+---+             +-+-+|         *
   +----+    *         | | |             | | |   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 looks like 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 SF Paths within
the lower level is greater than the number of SF Paths arriving to
the IBN.

The IBN is also the termination of lower-level SF paths. 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 SF 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 Path Identifier and Path Index
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
upper-level SF path by these methods:

1. Stateful per flow,

2. Pushing path identifier into metadata,

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

3.1.1. Flow-Stateful IBN

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

When packets are received by the IBN on a higher-level path, the
encapsulated packets are parsed for IP and transport-layer (TCP,
UDP...) coordinates. State is created, indexed by these coordinates
(a 5-tuple of {source-IP, destination-IP, source-port, destination-
port and transport protocol} in a typical case). The state contains
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., as done by a NAT) 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 the 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 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.

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
controller 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 path identifier (of the sub-domain) unambiguously indicates the
egress path (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 path identifier.

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 Path
identifier and lower path.

3.2. Gluing Levels Together

The path identifier 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 path 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.

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

   Controllers have been mentioned in this document without much
   explanation. 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. Acknowledgements

   The concept of Hierarchical Service Path Domains was introduced in
   draft-homma-sfc-forwarding-methods-analysis-01
   [I-D.homma-sfc-forwarding-methods-analysis] as a means to improve
   scalability of service chaining in large networks.

   The authors would like to thank the following individuals for taking
   the time to read and provide valuable feedback:

      Ron Parker

      Christian Jacquenet

      Jie Cao

7. IANA Considerations

   This memo includes no request to IANA.

8. 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.

9. References

9.1. 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,
              <http://www.rfc-editor.org/info/rfc2119>.

9.2. Informative References

   [I-D.homma-sfc-forwarding-methods-analysis]
              Homma, S., Kengo, K., Lopez, D., Stiemerling, M., and D.
              Dolson, "Analysis on Forwarding Methods for Service
              Chaining", draft-homma-sfc-forwarding-methods-analysis-01
              (work in progress), January 2015.

[I-D.ietf-sfc-architecture]
          Halpern, J. and C. Pignataro, "Service Function Chaining
          (SFC) Architecture", draft-ietf-sfc-architecture-07 (work
          in progress), March 2015.

[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.

[I-D.ietf-sfc-nsh]
          Quinn, P. and U. Elzur, "Network Service Header", draft-
          ietf-sfc-nsh-00 (work in progress), March 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 the advantage of hierarchical service function chaining.

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 3, 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.

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 4.

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.]---->
               . . . . . . . . . .   .  . . . . . . . . . . .
```
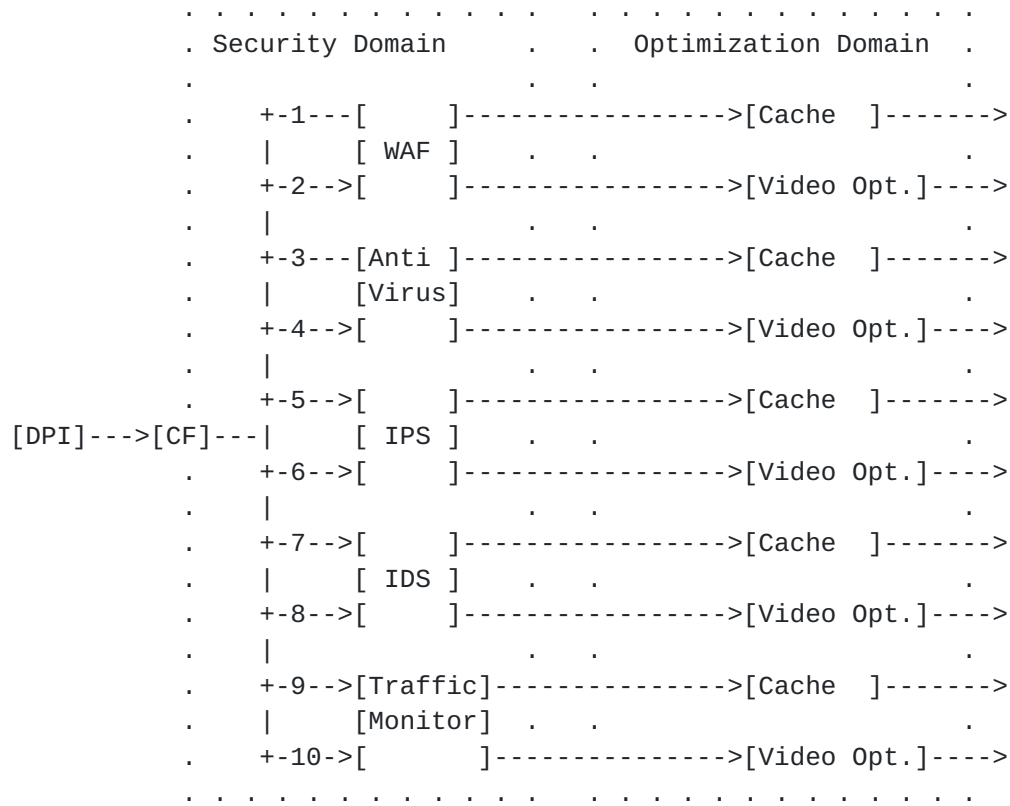
The classifier must select paths that determine the combination of
Security and Optimization concerns. 1:WAF+Cache, 2:WAF+VideoOpt,
3:AntiVirus+Cache, 4:AntiVirus+VideoOpt, 5: IPS+Cache,
6:IPS+VideoOpt, 7:IDS+Cache, 8:IDS+VideoOpt, 9:TrafficMonitor+Cache,
10:TrafficMonitor+VideoOpt

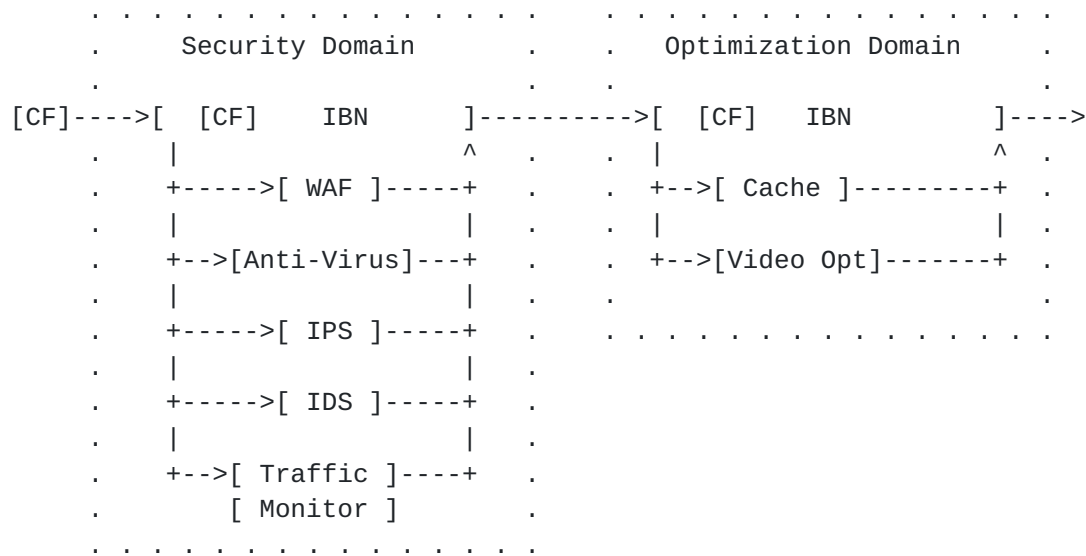   Figure 3: Flat SFC (Normal Branching)

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

              Figure 4: 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 5, 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 Consider the case when 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 |
                    +-----+
```
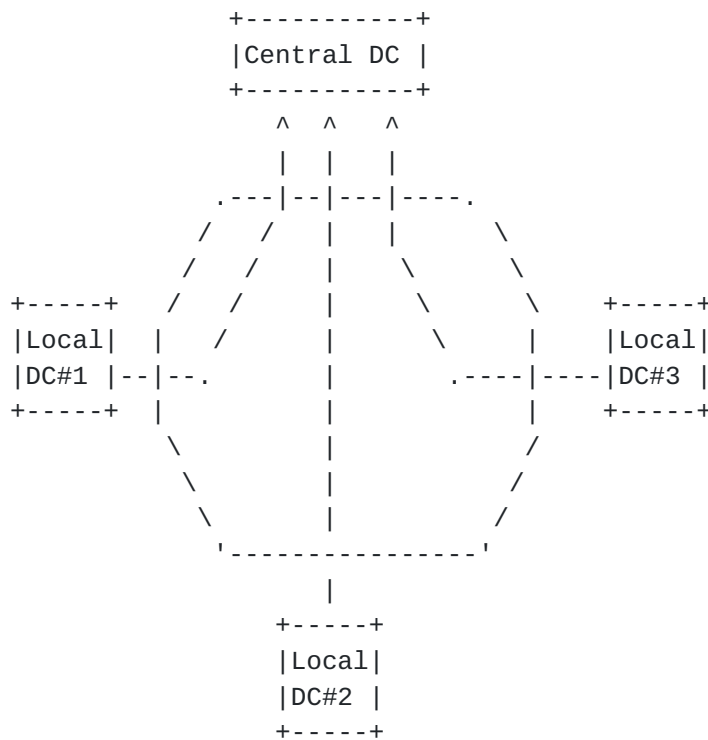
             Figure 5: Simplify Inter-DC SFC Management

   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 and the management complexity could be reduced
   significantly. 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 Group
   Rennes 35000
   France

   Email: mohamed.boucadair@orange.com

   Dapeng Liu
   Alibaba Group
   Beijing 100022
   China

   Email: max.ldp@alibaba-inc.com