Hierarchical Service Function Chaining (hSFC)

Abstract

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

   The goals of hSFC are to make a large-scale network easier to design,
   simpler to control, and supportive of independent functional groups
   within large network 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. The SFC architecture is described in detail in [RFC7665] and
   is not repeated here.

   This document focuses on the difficult problem of implementing SFC
   across a large, geographically dispersed network, potentially
   comprised of millions of hosts and thousands of network-forwarding
   elements and which may involve multiple operational teams (with
   varying functional responsibilities). We recognize that some
   stateful 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 transport-layer coordinate (typically, the 5-tuple of source IP
   address, source port number, destination IP address, destination port
   number, and transport protocol) stickiness to specific stateful SF
   instances.

   Difficult problems are often made easier by decomposing them in a
   hierarchical (nested) manner. So, instead of considering a single
   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 domains operated by as many SFC control
   plane components (under the same administrative entity).
   Coordination between such components is further discussed in this
   document.

   Each subdomain may support a subset of the network applications or a
   subset of the users. Decomposing a network should be done with care
   to ease monitoring and troubleshooting of the network and services as
   a whole. The criteria for decomposing a domain into multiple SFC-
   enabled subdomains are beyond the scope of this document. These
   criteria are deployment specific.

   An example of simplifying a network by using multiple SFC-enabled
   domains is further discussed in [USE-CASES].

   We assume the SFC-aware nodes use the Network Service Header (NSH)
   [RFC8300] or a similar labeling mechanism. Examples are described in
   Appendix A.

   The SFC-enabled domains discussed in this document are assumed to be
   under the control of a single organization (an operator, typically),
   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 this document to
consider domains operated by different organizations or dwell on
interoperator considerations.

We introduce the concept of an Internal Boundary Node (IBN) that acts
as a gateway between the levels of the hierarchy. We also discuss
options for realizing this function.

1.1. Experiment Goals

This document defines an architecture that aims to solve
complications that may be encountered when deploying SFC in large
networks. A single network is therefore decomposed into multiple
subdomains, each treated as an SFC-enabled domain. Levels of
hierarchy are defined, together with SFC operations that are specific
to each level. In order to ensure consistent SFC operations when
multiple subdomains are involved, this document identifies and
analyzes various options for IBNs to glue the layers together
(Section 4.1).

Because it does not make any assumptions about (1) how subdomains are
defined, (2) whether one or multiple IBNs are enabled per subdomain,
(3) whether the same IBN is solicited at both the ingress and egress
of a subdomain for the same flow, (4) the nature of the internal
paths to reach SFs within a subdomain, or (5) the lack of deployment
feedback, this document does not call for a recommended option to
glue the SFC layers together.

Further experiments are required to test and evaluate the different
options. A recommendation for hSFC might be documented in a future
specification when the results of implementation and deployment of
the aforementioned options are available.

It is not expected that all the options discussed in this document
will be implemented and deployed. The lack of an implementation
might be seen as a signal to recommend against a given option.

2. Terminology

   This document makes use of the terms defined in Section 1.4 of
   [RFC7665] and Section 1.3 of [RFC8300].

   The following terms are defined:

   o Upper-level domain: the entire network domain to be managed.

   o Lower-level domain: a portion of the network (called a subdomain).

   o Internal Boundary Node (IBN): is responsible for bridging packets
      between upper and lower levels of SFC-enabled domains.

3. Hierarchical Service Function Chaining (hSFC)

   A hierarchy has multiple levels: the topmost level encompasses the
   entire network domain to be managed, and lower levels encompass
   portions of the network. These levels are discussed in the following
   subsections.

3.1. Upper 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)

   o Subdomains

```
                        +------------+
                        |Subdomain#1 |
                        |  in DC1    |
                        +----+-------+
                             |
                    .---- SFF1 ------.    +----+
        +----+     /      /    |        \--|CF#4|
    --->|CF#1|--/---->'   |           \  +----+
        +----+  /   SC#1   |            \
            |   |          |             |
            |             V      .------>|--->
            |        /     /     |
             \      |    /       /
        +----+  \   |   /      / +----+
        |CF#2|---\  |  /     /---|CF#3|
        +----+    '---- SFF2 ------'   +----+
                             |
                        +----+-------+
                        |Subdomain#2 |
                        |  in DC2    |
                        +------------+
```
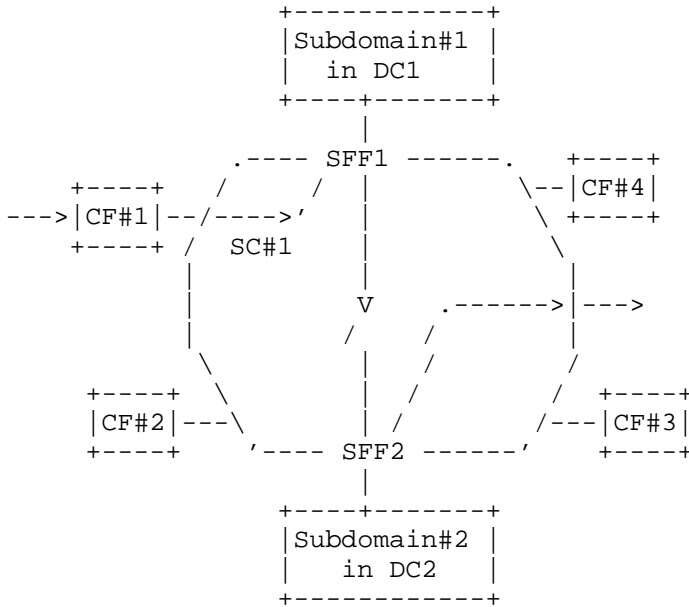
        Legend:
          SC#1: Service Chain 1
            DC: Data Center

     Figure 1: Network-Wide View of Upper Level of Hierarchy

   One path is shown from edge classifier (CF#1) to SFF1 to Subdomain#1
   (residing in Data Center 1) to SFF1 to SFF2 (residing in Data Center
   2) to Subdomain#2 to SFF2 to network egress.

   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 set of SFFs
   and subdomains, with the operations within subdomains being opaque to
   the upper levels.

   We expect the system to include a top-level control plane having
   responsibility for configuring forwarding policies and traffic-
   classification rules.

   The top-level Service Chaining control plane manages end-to-end
   service chains and associated service function paths from network
   edge points to subdomains. It also configures top-level classifiers

at a coarse level (e.g., based on source or destination host) to
forward traffic along paths that will transit across appropriate
subdomains.

Figure 1 shows one possible service chain passing from the edge
through two subdomains to network egress. The top-level control
plane does not configure traffic-classification rules or forwarding
policies within the subdomains.

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 subdomains and egress the network. Note that the
various paths that may be followed within a subdomain are not
represented by distinct network-wide SFPs; specific policies at the
ingress nodes of each subdomain bind flows to subdomain paths.

Packets are classified at the edge of the network to select the paths
by which subdomains are to be traversed. At the ingress of each
subdomain, packets are reclassified to paths directing them to the
required SFs of the subdomain. At the egress of each subdomain,
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 subdomain.

It should be assumed that some SFs require bidirectional symmetry of
paths (see more in Section 5). Therefore, the classifiers at the top
level must be configured with policies ensuring outgoing packets take
the reverse path of incoming packets through subdomains.

3.2. Lower Levels

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

   Figure 2 shows a subdomain interfaced with an upper-level domain by
   means of an Internal Boundary Node (IBN). An IBN acts as an SFC-
   aware SF in the upper-level domain and as a classifier in the lower-
   level domain. As such, data packets entering the subdomain are
   already SFC encapsulated. Also, 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. Finally, the egress IBN
   restores packets to the original SFC shim and hands them off to SFFs.

   Each subdomain intersects a subset of the total paths that are
   possible in the upper-level domain. An IBN is concerned with upper-
   level paths, but only those traversing its subdomain.

Each subdomain is likely to have a control plane that can operate
independently of the top-level control plane, managing
classification, forwarding paths, etc., within the level of the
subdomain, with the details being opaque to the upper-level control
elements. Section 4 provides more details about the behavior of an
IBN.

The subdomain control plane configures the classification rules in
the IBN, where SFC encapsulation of the top-level domain is converted
to/from SFC encapsulation of the lower-level domain. The subdomain
control plane also configures the forwarding rules in the SFFs of the
subdomain.

```
    +----+    +-----+  +---------------------+   +-----+
    |    |    | SFF |  |   IBN 1  (in DC 1)  |   | SFF |
    |    |SC#1|     |  | +----------------+  |   |     |
 ->|    |===============>|     SFF        |================>
    |    |    +-----+  | +----------------+  |   +-----+
    | CF |             |  |               ^  |
    |    |             |  v               |  |
    |    |             |+------------------+| Upper 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|        *
             *  +---+  +------+  +---+  +------+        *
             * * * * * * * * * * * * * * * * * * * * * *
```

Legend:
      *** Subdomain boundary
      === upper-level chain
      --- lower-level chain

      Figure 2: Example of a Subdomain within an Upper-Level Domain

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

4. Internal Boundary Node (IBN)

   As mentioned in the previous section, a network element termed an
   "Internal Boundary Node" (or IBN) is responsible for bridging packets
   between upper and lower layers of SFC-enabled domains. It behaves as
   an SF to the upper level (Section 3.1) and looks like a classifier
   and end of chain to the lower level (Section 3.2).

   To achieve the benefits of hierarchy, the IBN should be applying
   fine-grained traffic-classification rules at a 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 SFPs must be returned to the upper-
   level SFPs and forwarded to the next hop in the upper-level domain.

   When different metadata schemes are used at different levels, the IBN
   has further responsibilities: when packets enter the subdomain, the
   IBN translates upper-level metadata into lower-level metadata; and
   when packets leave the subdomain at the termination of lower-level
   SFPs, the IBN translates lower-level metadata into upper-level
   metadata.

   Appropriately configuring IBNs is key to ensuring the consistency of
   the overall SFC operation within a given domain that enables hSFC.
   Classification rules (or lack thereof) in the IBN classifier can, of
   course, impact upper levels.

4.1. IBN Path Configuration

   The lower-level domain may be provisioned with valid upper-level
   paths or allow any upper-level paths.

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

   At the termination of an SFP in the subdomain, packets can be
   restored to an original upper-level SFP by implementing one of these
   methods:

   1. Saving the SPI and SI in transport-layer flow state
       (Section 4.1.1).

   2. Pushing the SPI and SI into a metadata header (Section 4.1.2).

   3. Using unique lower-level paths per upper-level path coordinates
       (Section 4.1.3).

   4. Nesting NSH headers, encapsulating the upper-level NSH headers
       within the lower-level NSH headers (Section 4.1.4).

   5. Saving the upper level with a flow identifier (ID) and placing an
       hSFC Flow ID into a metadata header (Section 4.1.5).

4.1.1. Flow-Stateful IBN

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

   When packets are received by the IBN on an upper-level path, the
   classifier parses encapsulated packets for IP and transport-layer
   (TCP, UDP, etc.) coordinates. State is created, indexed by some or
   all transport coordinates (typically, {source-IP, destination-IP,
   source-port, destination-port, and transport protocol}). The state
   contains, at minimum, the critical fields of the encapsulating SFC
   header (SPI, SI, MD Type, flags); additional information carried in
   the packet (metadata, TTL) may also be extracted and saved as state.
   Note that some fields of a packet may be altered by an SF of the
   subdomain (e.g., source IP address).

   Note that this state is only accessed by the classifier and
   terminator functions of the subdomain. Neither the SFFs nor SFs have
   knowledge of this state; in fact they may be agnostic about being in
   a subdomain.

   One approach is to ensure that packets are terminated at the end of
   the chain at the same IBN that classified the packet at the start of
   the chain. If the packet is returned to a different egress IBN,
   state must be synchronized between the IBNs.

   When a packet returns to the IBN at the end of a chain (which is the
   SFP-terminating node of the lower-level chain), the SFC header is
   removed, the packet is parsed for flow-identifying information, and
   state is retrieved from within the IBN using the flow-identifying
   information as index.

   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 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 recreated by each new packet arriving from the upper-
level domain. For example, if an IBN loses all flow state, the state
is recreated by an endpoint 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 subdomain's
control plane to modify paths when upper-level paths are changed.
The complexity of the upper-level domain does not cause complexity in
the lower-level domain.

Since it doesn't depend on NSH in the lower-level domain, this flow-
stateful approach can be applied to translation methods of converting
NSH to other forwarding techniques (refer to Section 7).

4.1.2. Encoding Upper-Level Paths in Metadata

An IBN can push the upper-level SPI and 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 the 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 a new metadata header may inflate packet size when variable-
length metadata (NSH MD Type 0x2) is used.

It is conceivable that the MD Type 0x1 Fixed-Length Context Header
field of the NSH is not all relevant to the lower-level domain. In
this case, 32 bits of the Fixed-Length Context Header field could be
repurposed within the lower-level domain and restored when leaving.

If flags or TTL (see Section 4.4) from the original header also need
to be saved, more metadata space will be consumed.

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

4.1.3. Using Unique Paths per Upper-Level Path

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

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

A downside of this approach is that the number of paths in the lower-
level domain is multiplied by the number of paths in the upper-level
domain that traverse the lower-level domain. That is, a subpath must
be created for each combination of upper SPI/SI and lower chain. The
number of paths required for lower-level domains will increase
exponentially as hierarchy becomes deep.

A further downside of this approach is that it requires upper and
lower levels to utilize the same metadata configuration.

Furthermore, this approach does not allow any information to be
stashed away in state or embedded in metadata. For example, the TTL
modifications by the lower level cannot be hidden from the upper
level.

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

When packets arrive at an 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.

```
+-------------------------------+
|   Outer-Transport Encapsulation   |
+-------------------------------+
|         Lower-NSH Header         |
+-------------------------------+
|         Upper-NSH Header         |
+-------------------------------+
|          Original Packet         |
+-------------------------------+
```
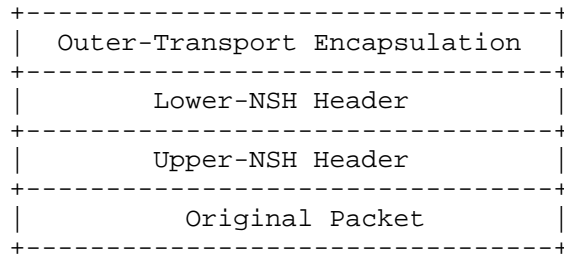
Figure 3: Encapsulation of NSH within NSH

The traffic with this stack of two NSH headers 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. The packet will be forwarded in the top-level
domain 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 metadata. All header
fields, including flags and TTL, are easily restored when the chains
of the subdomain terminate.

However, the downside is that it does require SFC-aware SFs in the
lower-level domain to be able to parse multiple NSH layers. If an
SFC-aware SF injects packets, it must also be able to deal with
adding appropriate multiple layers of headers to injected packets.

By increasing packet overhead, nesting may lead to fragmentation or
decreased MTU in some networks.

4.1.5. 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".

The hSFC Flow ID is placed, for example, in the NSH Fixed-Length
Context Header field of the packet in the lower-level domain, as
shown in Figure 4. Likewise, hSFC Flow ID may be encoded as a
Variable-Length Context Header field when MD Type 0x2 is used.

When packets exit the lower-level domain, the IBN uses the hSFC Flow
ID to retrieve the appropriate NSH encapsulation for returning the
packet to the upper domain. The hSFC Flow ID Context Header is then
stripped by the IBN.

```
    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|U|    TTL    |   Length  |U|U|U|U|MD Type| Next Protocol |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Service Path Identifier              | Service Index |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        hSFC Flow ID                           |
   |              Zero Padding or other fields                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
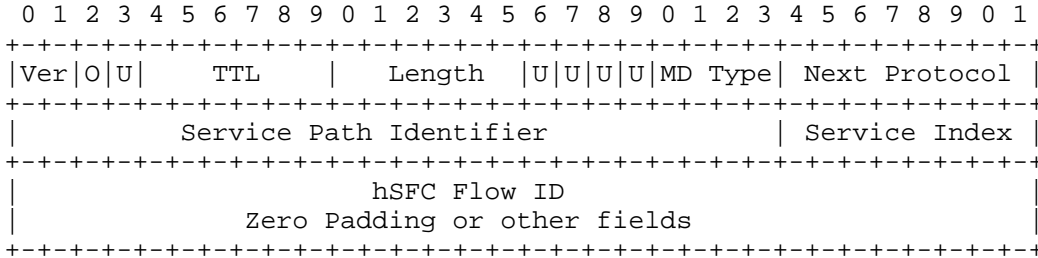
             Figure 4: Storing hSFC Flow ID in Lower-Level NSH
          Fixed-Length Context Header Field ([RFC8300], Section 2.4)

Advantages of this approach include:

o It does not require state to be based on a 5-tuple, so it works
   with SFs that change the IP addresses or port numbers of a packet,
   such as NATs.

o It does not require all domains to have the same metadata scheme.

o It can be used to restore any upper-domain information, including
   metadata, flags, and TTL, not just the service path.

o The lower-level domain only requires a single item of metadata
   regardless of the number of items of metadata used in the upper
   domain.

o The SFC-related functionality required by this approach in an SFC-
   aware SF is able to preserve and apply metadata, which is a
   requirement that was already present in [RFC8300].

Disadvantages include those of other stateful approaches, including
state timeout and synchronization, mentioned in Section 4.1.1.

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 hSFC Flow IDs cannot be
created or old hSFC Flow IDs are discarded while still in use.

4.2. Gluing Levels Together

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

In some cases, the meanings of the various path IDs and metadata must
be coordinated between domains for the sake of proper end-to-end SFC
operation.

One approach is to use well-known identifier values in metadata,
maintained in a global 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
subdomain classifier could have a policy, "if pathID = classA then
chain packet to path 1234"; the upper-level controller would be
expected to configure the concrete upper-level "pathID" for "classA".

4.3. Decrementing Service Index

Because the IBN acts as an SFC-aware SF to the upper-level domain, it
must decrement the Service Index in the NSH headers of the upper-
level path. This operation should be undertaken when the packet is
first received by the IBN, before applying any of the strategies of
Section 4.1, immediately prior to classification.

4.4. Managing TTL

The NSH base header contains a TTL field [RFC8300]. There is a
choice:

    a subdomain may appear as a pure service function, which should
    not decrement the TTL from the perspective of the upper-level
    domain, or

    all of the TTL changes within the subdomain may be visible to the
    upper-level domain.

Some readers may recognize this as a choice between "pipe" and
"uniform" models, respectively [RFC3443].

The network operator should be given control of this behavior,
choosing whether to expose the lower-level topology to the upper
layer. An implementation may support per-packet policy, allowing
some users to perform a layer-transcending trace route, for example.

The choice affects whether the methods of restoring the paths in
Section 4.1 restore a saved version of the TTL or propagate it with
the packet. The method of Section 4.1.3 does not permit topology
hiding. The other methods of Sections 4.1.1, 4.1.2, 4.1.4, and 4.1.5
have unique methods for restoring saved versions of the TTL.

5. Subdomain Classifier

Within the subdomain (referring to Figure 2), as the classifier
receives incoming packets, the upper-level encapsulation is treated
according to one of the methods described in Section 4.1 to either
statefully store, encode, or nest header information. The classifier
then selects the path and metadata for the packet within the
subdomain.

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 SF instances 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 SFs in and out. All of the complexities of
load-balancing among multiple SFs can be handled within a subdomain,
under control of the classifier, allowing the upper-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 subdomain be
received by the same classifier or a coordinated cluster of
classifiers. There are both stateful and stateless approaches to
ensuring bidirectional symmetry.

6. Control Plane Elements

   Although SFC control protocols have not yet been standardized (as of
   2018), from the point of view of hierarchical service function
   chaining, we have these expectations:

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

   o Each control plane is agnostic about other levels of the
      hierarchy. This aspect allows humans to reason about the system
      within a single domain and control-plane algorithms to use only
      domain-local inputs. Top-level control does not need visibility
      to subdomain policies, nor does subdomain control need visibility
      to upper-level policies. (Top-level control considers a subdomain
      as though it were an SF.)

   o Subdomain control planes are agnostic about the control planes of
      other subdomains. This allows both humans and machines to
      manipulate subdomain policy without considering policies of other
      domains.

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

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

7. Extension for Adapting 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 forwarding 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                   .
         .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
```
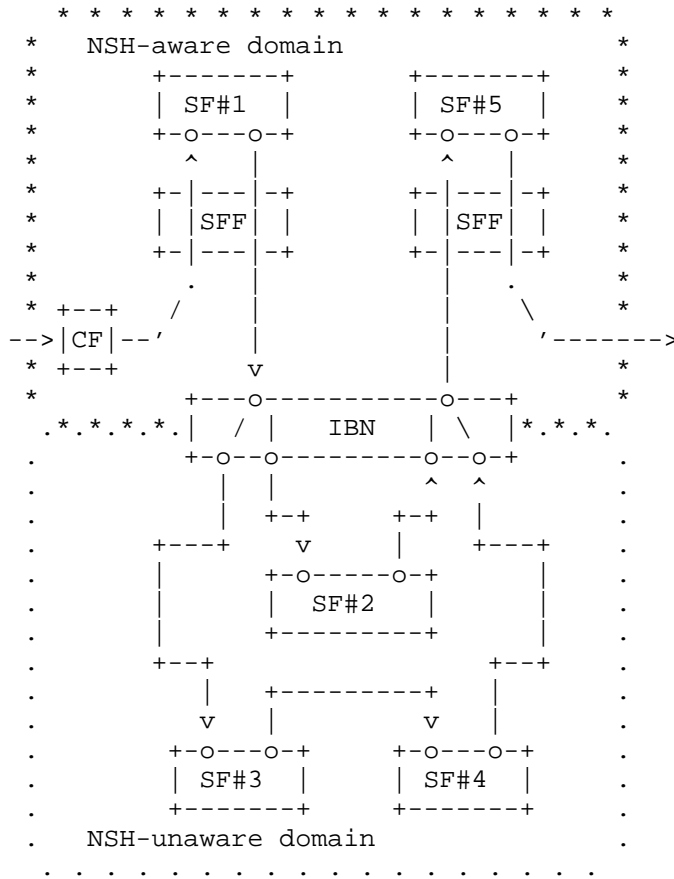
   SF#1 and SF#5 are NSH aware; SF#2, SF#3, and SF#4 are NSH unaware.
   In the NSH-unaware domain, packets are conveyed in a format supported
   by SFs that are deployed there.

            Figure 5: Dividing NSH-Aware and NSH-Unaware Domains

7.1. 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

   o Supporting multitenancy

7.2. Requirements for an 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 upper-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 4.1.1)

   o Using unique lower-level paths per upper-level NSH coordinates
      (refer to Section 4.1.3)

   Using the unique paths approach would be especially 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 the NSH to the format appropriate to the packet in the lower-level
   path. For example, some legacy SFs identify subscribers based on
   information about the network topology, such as the VLAN ID (VID),
   and the IBN would be required to create a VLAN to packets from
   metadata if the subscriber identifier is conveyed as metadata in
   upper-level domains.

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

   It is useful to permit metadata to be transferred between levels of a
   hierarchy. Metadata from an upper level may be useful within a
   subdomain, and a subdomain may augment metadata for consumption in an
   upper domain. However, allowing uncontrolled metadata between
   domains may lead to forwarding failures.

      In order to prevent SFs of lower-level SFC-enabled domains from
      supplying (illegitimate) metadata, IBNs may be instructed to only
      permit specific metadata types to exit the subdomain. Such
      control over the metadata in the upper level is the responsibility
      of the upper-level control plane.

To limit unintentional metadata reaching SFs of lower-level SFC-
enabled subdomains, IBNs may be instructed to only permit specific
metadata types into the subdomain. Such control of metadata in
the lower-level domain is the responsibility of the lower-level
control plane.

8. IANA Considerations

   This document has no IANA actions.

9. Security Considerations

   hSFC makes use of service chaining architecture; hence, it inherits
   the security considerations described in the architecture document
   [RFC7665].

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

   This document describes systems that may be managed by distinct teams
   that all belong to the same administrative entity. Subdomains must
   have consistent configurations in order to properly forward traffic.
   Any protocol designed to distribute the configurations must be secure
   from tampering. The means of preventing attacks from within a
   network must be enforced. For example, continuously monitoring the
   network may allow detecting such misbehaviors. hSFC adheres to the
   same security considerations as [RFC8300]. Those considerations must
   be taken into account.

   The options in Sections 4.1.2 and 4.1.5 assume the use of a dedicated
   context header to store information to bind a flow to its upper-level
   SFP. Such a context header is stripped by the IBN of a subdomain
   before exiting a subdomain. Additional guards to prevent leaking
   unwanted context information when entering/exiting a subdomain are
   discussed in Section 7.2.

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

9.1. Control Plane

   Security considerations related to the control plane are discussed in
   the corresponding control specification documents (e.g.,
   [BGP-CONTROL], [PCEP-EXTENSIONS], or [RADIUS]).

9.2. Infinite Forwarding Loops

   Distributing policies among multiple domains may lead to forwarding
   loops. NSH supports the ability to detect loops (as described in
   Sections 4.3 and 4.4 of [RFC8300]), but the means of ensuring the
   consistency of the policies should be enabled at all levels of a
   domain. Within the context of hSFC, it is the responsibility of the
   Control Elements at all levels to prevent such (unwanted) loops.

10. References

10.1. Normative References

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

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

10.2. Informative References

   [BGP-CONTROL]
              Farrel, A., Drake, J., Rosen, E., Uttaro, J., and L.
              Jalil, "BGP Control Plane for NSH SFC", Work in Progress,
              draft-ietf-bess-nsh-bgp-control-plane-04, July 2018.

   [PCEP-EXTENSIONS]
              Wu, Q., Dhody, D., Boucadair, M., Jacquenet, C., and J.
              Tantsura, "PCEP Extensions for Service Function Chaining
              (SFC)", Work in Progress,
              draft-wu-pce-traffic-steering-sfc-12, June 2017.

   [RADIUS]   Maglione, R., Trueba, G., and C. Pignataro, "RADIUS
              Attributes for NSH", Work in Progress,
              draft-maglione-sfc-nsh-radius-01, October 2016.

   [RFC3443] Agarwal, P. and B. Akyol, "Time To Live (TTL) Processing
              in Multi-Protocol Label Switching (MPLS) Networks",
              RFC 3443, DOI 10.17487/RFC3443, January 2003,
              <https://www.rfc-editor.org/info/rfc3443>.

   [USE-CASES]
            Kumar, S., Tufail, M., Majee, S., Captari, C., and S.
            Homma, "Service Function Chaining Use Cases In Data
            Centers", Work in Progress,
            draft-ietf-sfc-dc-use-cases-06, February 2017.

Appendix A. Examples of Hierarchical Service Function Chaining

   The advantage of hSFC 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.

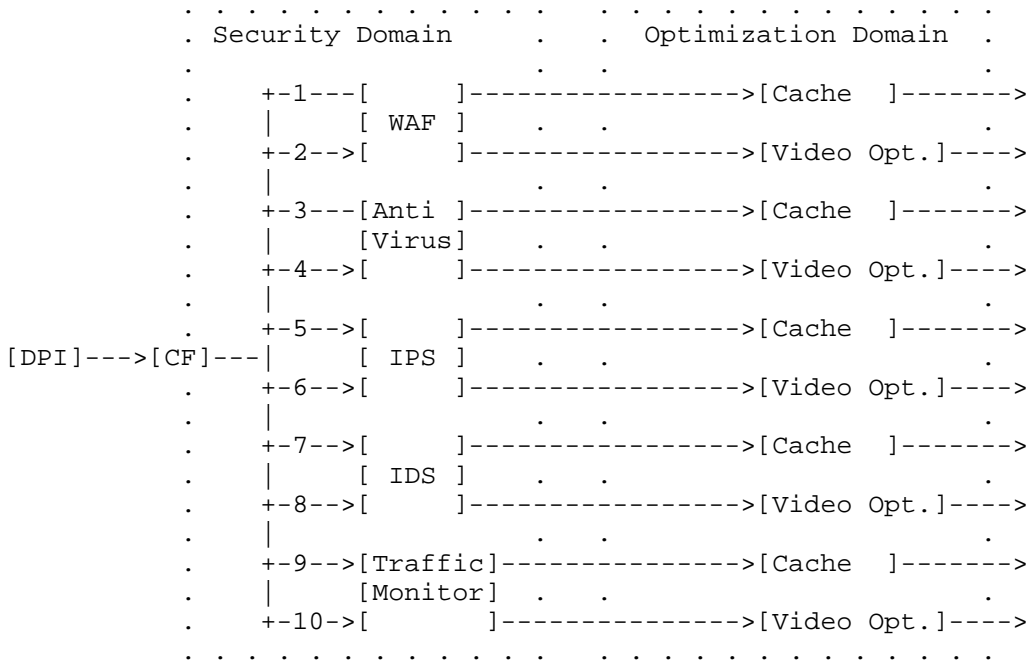A.1. Reducing the Number of Service Function Paths

   In this case, hSFC is used to simplify service function chaining
   management by reducing the number of SFPs.

   As shown in Figure 6, there are two domains, each with different
   concerns: a Security Domain that selects SFs based on network
   conditions and an Optimization Domain that selects SFs 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 hSFC, only five SFPs in Security Domain and two
   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
   SFs in all of the domains. In the hSFC model, the number of SFPs is
   the sum of the number of SFs. For example, adding a "bypass" path in
   the Optimization Domain would cause the flat model to require 15
   paths (five 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.]---->
        . . . . . . . . . . . .   . . . . . . . . . . . .
```

    Legend:
       IDS: Intrusion Detection System
       IPS: Intrusion Prevention System
       WAF: Web Application Firewall
       DPI: Deep Packet Inspection

    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 ]        .
    . . . . . . . . . . . . . . .
```
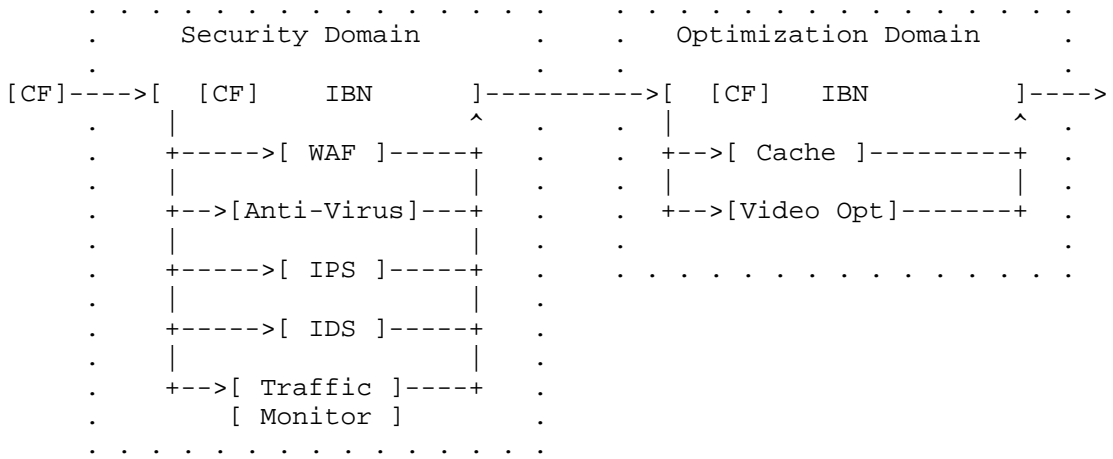
                Figure 7: Simplified Path Management with hSFC

A.2. Managing a Distributed DC Network

   Hierarchical service function chaining can be used to simplify inter-
   DC SFC management. In the example of Figure 8, 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 SFs are deployed as dedicated hardware appliances, and there
      is a desire to lower the cost (both CAPEX and OPEX) of deploying
      such SFs in all data centers.

   o Some SFs are being trialed or introduced, or they otherwise handle
      a relatively small amount of traffic. It may be cheaper to manage
      these SFs in a single central data center and steer packets to the
      central data center than to manage these SFs in all data centers.
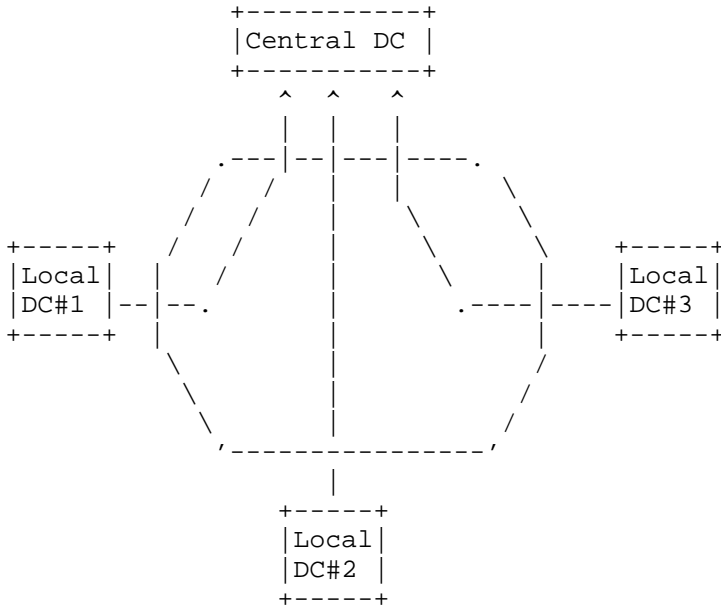
```
                    +-----------+
                    |Central DC |
                    +-----------+
                       ^   ^   ^
                       |   |   |
                   .---|--|---|----.
                  /   /   |   |     \
                 /   /    |    \     \
  +-----+      /    /     |     \     \      +-----+
  |Local|     |    /      |      \     |     |Local|
  |DC#1 |--|--.     |     |       .----|----|DC#3 |
  +-----+   |        \    |      /     |     +-----+
            \         \   |     /
             \         \  |    /
              \         \ |   /
               '----------------'
                          |
                      +-----+
                      |Local|
                      |DC#2 |
                      +-----+
```

                  Figure 8: Simplify Inter-DC SFC Management

   For large DC operators, one local DC may have tens of thousands of
   servers and hundreds 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 a large-scale DC, using flat SFC is very complex, requiring a
   super controller to configure all DCs. For example, any changes to
   SFs or SFPs in the central DC (e.g., deploying a new SF) would
   require updates to all of the SFPs 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 DC can be managed
   independently to significantly reduce management complexity. SFPs
   between DCs can represent abstract notions without regard to details
   within DCs. Independent controllers can be used for the top level
   (getting packets to pass the correct DCs) and local levels (getting
   packets to specific SF instances).

Authors' Addresses

    David Dolson
    Sandvine
    Waterloo, ON
    Canada

    Email: ddolson@acm.org

    Shunsuke Homma
    NTT
    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