Service Function Chaining D. Dolson
Internet-Draft Sandvine
Intended status: Informational S. Homma
Expires: November 26, 2015 NTT
D. Lopez
Telefonica I+D
May 25, 2015

Hierarchical Service Chaining
draft-dolson-sfc-hierarchical-00

Abstract

This document describes a network architecture for deploying service function chaining with multiple levels of administration within an organization.

The multiple levels of administration allow operators to compartmentalize a large network into multiple domains of responsibility, with each domain being independently managed and consequently easier to reason about.

Table of Contents

1. Introduction

Service Function Chaining (SFC) allows an operator to prescribe packet paths taken through their network. 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. We expect asymmetrical routing is inherent in the network, while recognizing that some Service Functions require bidirectional traffic for transport-layer sessions. We expect some paths need to be selected on the basis of application metadata accessible to the network, with 5-tuple stickiness to specific Service Function instances.

Difficult problems are often made easier by decomposing them in a hierarchical (nested) manner. So instead of considering an omniscient controller that can create complete paths from one end of the network to the other, we break the network into smaller pieces. Each piece may support a subset of the network applications or a subset of the users.

A previous example of simplifying a network by using multiple SF domains can be seen in draft-ietf-sfc-dc-use-cases [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 Chaining

A hierarchy has multiple conceptual levels. In Hierarchical Service Chaining, the top-most level encompasses the entire network domain to be managed. Lower levels encompass smaller portions of the network.

2.1. Top Level

Considering example 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 service function components.

Top-level service function paths carry packets from classifiers to egress via SFFs and sub-domains, with the operations within sub-domains being opaque to the higher levels.

Network-wide Service Chaining orchestration is only concerned with creating service paths from network edge points to sub-domains within data centers and configuring classifiers at a coarse level (e.g., based on source or destination host) to get traffic onto paths that will arrive at appropriate sub-domains. The figure shows one possible service chain passing from edge, through two sub-domains, to network egress.

At this high level, the number of SF Paths required is on the order of the number of ways in which a packet needs to traverse different sub-domains and egress the network.

It should be assumed that some service functions in the network require bidirectional symmetry of paths (see more in section Section 4). Therefore the classifiers at the top level need to ensure server-to-client packets take the reverse path of client-to-server packet 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 to SFF1 to SFF2 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 system unto itself.

Unlike the top level, however, data packets entering the sub-domain are already encapsulated within SFC transport. Figure 2 shows a sub-domain interfaced to a higher-level domain by means of an SF-Domain Gateway. It is the purpose of the SF Domain Gateway to remove packets from the SFC transport, apply Classification, and direct the packets to the selected local service function paths ending back at the SF Domain Gateway. The SF 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 SF Domain Gateway is concerned with higher-level paths, but only those traversing the sub-domain. The top-level controller configures top-level paths at the SF Domain Gateway, but the top-level paths are otherwise unknown within the sub-domain. The SF Domain Gateway provides adaptation between the levels.

```
  +----+    +-----+  +----------------------+   +-----+
  |    |SC#1| SFF |  | SF Domain Gateway 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. SF Domain Gateway

A network element termed "SF Domain Gateway" 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 SF Domain Gateway should be making more granular traffic classifications at the lower level than the traffic passed to it. This means that the number of SF Paths within the lower level is larger than the number of SF Paths arriving to the gateway.

The SF Domain Gateway 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. SF Domain Gateway 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.

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

1. Statefully per flow,

2. Pushing path identifier into meta-data,

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

3.1.1. Flow-Stateful SF Domain Gateway

An SF Domain Gateway can be flow-aware, returning packets to the correct higher-level SF path on the basis of 5-tuple of packets exiting the lower-level SF paths.

When packets are received by the SF Domain Gateway on a higher-level path, the encapsulated packets are parsed for IP and transport-layer (TCP or UDP) coordinates. State is created, indexed by the 5-tuple of {source-ip, destination-ip, source-port, destination-port and transport protocol}. The state contains critical fields of the encapsulating SFC header (or perhaps the entire header).

When a packet returns to the SF Domain Gateway 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 by the 5-tuple of the packet. The state contains the information required to forward the packet within the higher-level service chain.

In the stateful approach, there are issues caused by the state, such as how long the state should be retained, 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, since it can be re-created by each new packet arriving from the higher-level domain. For example, if an SF-Domain Gateway loses all flow state, the state is re-created by an end-point retransmitting a TCP packet.

If a network handles multiple routing domains, the 5-tuple may be augmented with a 6th parameter, perhaps using some meta-data to identify the routing domain.

In this stateful 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.

3.1.2. Saving Upper-Level Path in Meta-Data

An SF Domain Gateway can push the upper-level service path identifier (SPI) and service index (SI) into a meta-data field of the lower-level NSH encapsulation. When packets exit the lower-level path, the upper-level SPI and SI can be restored from the meta-data retrieved from the packet.

This approach requires the SFs in the path to be capable of forwarding the meta-data and to appropriately apply meta-data to any packets injected for a flow.

Using new meta-data may inflate packet size when variable-length meta-data (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 meta-data slots of the Mandatory Context Header could be repurposed within the lower-level domain. (And restored when leaving.)

In this meta-data 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).

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 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 SF Domain Gateway 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 meta-data, communicated by some organizational registry.

Another approach is to use well-known labels for path identifiers or meta-data, as an indirection to the actual identifiers. The actual identifiers can be assigned by control 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 SF Domain Gateway removes incoming packets from the higher-level encapsulation, 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 tractable 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 classifier to be aware of symmetric paths and 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 to 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. Controllers

Controllers have been mentioned in this document without being explained. Although controllers have not yet been standardized, from the point of view of hierarchical service chaining we have these expectations:

Each controller manages a single level of hierarchy.

Each controller is agnostic about other levels of hierarchy.

Sub-domain controllers are agnostic about controllers of other sub-domains.

6. Summary

The goals of the hierarchical SFC architecture are to make a large-scale network easier to reason about, simpler to control and allow independent domains of administration. This document has outlined an approach that serves those goals, with some suggested approaches to implementing the SF Domain Gateway.

7. 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.

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

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

Furthermore, hierarchical service 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.

10. References

10.1. Normative References

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

10.2. 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.

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