ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: January 16, 2014

Y. Yang, Ed.
Yale University

July 15, 2013

ALTO Topology Considerations
draft-yang-alto-topology-00.txt

Abstract

The Application-Layer Traffic Optimization (ALTO) Service has defined Network and Cost maps to provide basic network information. In this document, we discuss some initial thinking on adding topology in ALTO.

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].

Status of this Memo

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

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

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

This Internet-Draft will expire on January 16, 2014.

Table of Contents

1. Introduction

Topology is a basic information component that a network can provide to network management tools and applications. Example tools and applications that can utilize network topology include traffic engineering, network services (e.g., VPN) provisioning, PCE, application overlays, among others [RFC5693,I-D.amante-i2rs-topology-use-cases, I-D.lee-alto-app-net-info-exchange].

A basic challenge in exposing network topology is that there can be multiple representations of the topology of the same network infrastructure, and each representation may be better suited for its own set of deployment scenarios. For example, the current base ALTO protocol [I-D.ietf-alto-protocol] is designed for a setting of exposing network topology using the extreme "my-Internet-view" representation, which does not report any internal network switches, and hence is a "single-switch" abstraction. We interpret the word "switch" in the generic sense of network equipment in this document, not limited to L2 devices. An issue of this abstraction is that there are applications who may need details about network elements (e.g., specific network switches and links), but these are not exposed in the single-switch topology abstraction. An opposite of the single-switch representation is the complete raw topology, spanning across multiple layers, to include all details of network states such as endhosts attachment, physical links, physical switch equipment, and logical structures (e.g., LSPs) already built on top of physical infrastructure devices. A problem of the raw topology representation, however, is that its exposure may violate privacy constraints. Also, a large raw topology may be overwhelming and unnecessary for specific applications.

In this document, we discuss an extension of ALTO for topology exposure. We focus on a particular network. We assume a raw network topology, i.e., the ground truth. How the raw topology information is collected is outside the scope of this document.

The organization of this document is not a typical normative document. In particular, we first introduce concepts through examples, to better motivate the design. Then we introduce a sketch of schema for exposing topology in ALTO. There are details of the schema that are not specified and the intention is to integrate with other designs such as [I-D.lee-alto-app-net-info-exchange]. Next we give a framework of topology transformations to help with the understanding of deriving multiple representations of the topology of the same network infrastructure. We finish by pointing out operations based on new ALTO topology exposure.

2. Motivation using Examples

We distinguish between endhosts and the network infrastructure of the network. Endhosts are sources and destinations of data that the network infrastructure carries. The network itself is neither the source or the destination of data.

For the given network, it provides "access ports" or access points where digital signal from endhosts enter and leave the network. One should understand "access ports" in a general sense. For example, an access port can be a physical Ethernet port connecting to a specific endhost, or it can be a port connecting to a CE which connects to a large number of endhosts. Let AP be the set of access ports that the network provides.

2.1. Single-Switch

A high-level abstraction of a network topology is only the set AP, and one can visualize the network as a single switch. At each ap in AP, a set of endhosts can be reached as destinations. Let dest(ap) denote the set of endhosts reachable at ap. The base ALTO protocol introduces PID to represent a partition of the set AP. Each subset in the partition is named as a PID, and the complete partition is conveyed as the Network Map. The ALTO base protocol then conveys the pair-wise connection properties from one PID to another PID through the "single-switch". This is the Cost Map.

2.2. Multiple Switches

Now, assume that the network actually consists of multiple switches, and the application needs to know more detailed topology. To help with the understanding, we consider the example case that the network has three switches, s1, s2 and s3. Each switch is connected to the other. The set AP is naturally divided as AP1, AP2, and AP3, denoting the access ports connected to the three switches respectively. The topology then exposed is simple to represent: there are three components: PIDs: {AP1, AP2, AP3}, Switches: {s1, s2, s3}, and Links: {s1->s2, s2->s1, ..., s2->s3, s3->s2}. It is straightforward to extend ALTO to represent the two additional components: Switches and Links.

2.3. Network Constraints/Policies of a Fixed E2E Path

Although the preceding 3-component representation is suited for some settings, e.g., traffic engineering who works on the raw topology, some other applications may need to or should only know a topology that encodes existing network constraints or policies. Note that such constraints may also come from another network tool or

application, to allow modular management composition.

For example, there can be a constraint, policy, or modular composition of the result of another application that endhosts from ap1 in AP1 connected to s1 must use the path s1 -> s2 -> s3 to reach endhosts at ap3 in AP3. To encode such a constraint to an application, there can be two choices: (1) create virtual switches and links still use the uniform graph-based representation; or (2) enumerate such a constraint in an end-to-end overlay representation.

2.4. Multi-Layer Topology

Now assume that the link s1 -> s2 is actually a given optical path, and s1 -> s3 is another given optical path, and the deployment scenario requires that this detail be exposed to the tool or application on top of topology exposure, for example, to evaluate reliability considering shared risk link groups. To handle such a case, one can encode the optical topology in a graph representation, and also include (layer 3) end-to-end entries s1 -> s2 and s1 -> s3 to specify the paths or some transformation of the paths such as encoded, opaque shared-risk-link group numbers for each of the s1 -> s2 and s1 -> s3 paths.

2.5. Multicast and Broadcast Topology

Next consider more complexity. Assume that the link from s1 -> s2 is actually a wireless link and the application may benefit in knowing that s1 -> s2 and s1 -> s3 can be active simultaneously. In other words, s1 -> [s2, s3] is a broadcast link. Knowing such links can be beneficial in settings such as wireless opportunistic routing.

3. Sketch of Schema

Given the preceding, we consider the following schema, which consists of EndhostMap, Topology, and Overlays.

EndhostMap: which encodes PIDs representing endhosts.

```
      object {

        VersionTag     map-vtag;
        EndhostMapData map;           // CHANGE: rename NetworkMap
                                      // to EndhostMap??

      } InfoResourceEndhostMap;

      object-map {
        PIDName -> EndpointAddrGroup; // already defined in base ALTO
      } EndhostMapData;
```

Topology: A network can define 0 to multiple topology maps, where each topology consists of switches and links:

```
      object {
        VersionTag     map-vtag;
        SwitchMapData  switches;
        LinkMapData    links;

      } InfoResourceTopology;

      object-map {
        JSONString ->  SwitchProperties; // switch name to properties
      } SwitchMapData;

      object {
        AccessLinks    alinks;    // between a PID to a switch
        TransportLinks tlinks;    // between two switches
      } LinkMapData;
```

(Overlay) paths: A network can define 0 to multiple overlays on top of a given topology, and path can be recursive:

```
      object {
        PathType        type;      // E2ECostMap; LSPs; ...
        [PathMapData    map;]      // depends on type,
                                   // if it is E2ECostMap,
                                   // it is InfoResourceCostMap
                                   // defined in [alto-protocol]
      } PathMap;
```

4. Graph Transformations to Build Topology/Overlays

The preceding sections give a top-down derivation. In this section, we give a graph transformation framework to build the schema from a raw topology G(0). The network conducts transformations on G(0) to obtain other topologies, with the following objectives:

1. Simplification: G(0) may have too many details that are unnecessary for the receiving app (assume intradomain, and hence no security problem); and

2. Preservation of privacy: there are details that the receiving app should not be allowed to see; and

3. Convey of logical structure (e.g., MPLS paths already computed); and

4. Convey of capability constraints (the network can have limitations, e.g., it uses only shortest path routing); and

5. Allow modular composition: path from one point to another point is delegated to another app.

The transformation of G(0) is to achieve/encode the preceding. For conceptual clarity, we assume that the network uses a given set of operators. Hence, given a sequence of operations and starting from G(0), the network builds G(1), to G(2), ...

Below is a list of basic operators that the network may use to transform from G(n-1) to G(n):

- O1: Deletion of a switch/port/link from G(n-1);

- O2: Switch aggregation: a set Vs of switches are merged as one new (logical) switch, links/ports connected to switches in Vs are now connected to the new logical switch, and then all switches in Vs are deleted;

- O3: Path representation: For a given extra path from A to R1 to R2 ... to B in G(n-1), a new (logical) link A -> B is added; if the constraint is that A -> must use the path, it will be put into the Overlay;

- O4: Switch split: A switch s in G(n-1) becomes two (logical) switches s1 and s2. The links connected to s1 is a subset of the original links connected to s; so is s2.

5. Operations on Exported Topology

Going beyond the basic topology exposure from the network and applications/tools, we anticipate that applications and tools can derive results and feed to topology. In particular, we consider the following operations:

- Instantiation of app guidance in real network: The details of instantiation will be outside the scope of this document. Example protocols include PCEP Extensions for Stateful PCE [I-D.ietf-pce-stateful-pce], RSVP LSP's and their associated characteristics, (i.e.: head and tail-end LSR's, bandwidth, priority, preemption, etc.). The reason that we choose the preceding operator set is that they are "implementable".

- We also anticipate topology guided mapping of other data: to allow applications to subscribe to statistics and link status from the derived topology.

6. Security Considerations

This document has not conducted its security analysis.

7. IANA Considerations

This document does not specified its IANA considerations, yet.

8. Acknowledgments

The author thanks discussions with Erran Li, Tianyuan Liu, Andreas Voellmy, Haibin Song, and Yan Luo.

9. References

9.1. Normative References

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

9.2. Informative References

[I-D.amante-i2rs-topology-use-cases]
Amante, S., Medved, J., Previdi, S., and T. Nadeau, "Topology API Use Cases",

draft-amante-i2rs-topology-use-cases-00 (work in progress), February 2013.

[I-D.ietf-alto-protocol]
Alimi, R., Penno, R., and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-17 (work in progress), July 2013.

[I-D.lee-alto-app-net-info-exchange]
Lee, Y., Bernstein, G., Choi, T., and D. Dhody, "ALTO Extensions to Support Application and Network Resource Information Exchange for High Bandwidth Applications", draft-lee-alto-app-net-info-exchange-02 (work in progress), July 2013.

[RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.

Author's Address

Y. Richard Yang (editor)
Yale University
51 Prospect St
New Haven CT
USA

Email: yry@cs.yale.edu