ALTO
Internet-Draft
Intended status: Standards Track
Expires: 28 April 2022

K. Gao
Sichuan University
Y. Lee
Samsung
S. Randriamasy
Nokia Bell Labs
Y.R. Yang
Yale University
J. Zhang
Tongji University

25 October 2021

An ALTO Extension: Path Vector
draft-ietf-alto-path-vector-19

Abstract

This document is an extension to the base Application-Layer Traffic Optimization (ALTO) protocol. It extends the ALTO Cost Map ~~service~~ and ALTO Property Map services so that ~~the~~ an application client can decide which endpoint(s) to connect based on not only numerical/ordinal cost values but also details of the paths. This is useful for applications whose performance is impacted by specified components of a network on the end-to-end paths, e.g., they may infer that several paths share common links and prevent traffic bottlenecks by avoiding such paths. This extension introduces a new abstraction called Abstract Network Element (ANE) to represent these components and encodes a network path as a vector of ANEs. Thus, it provides a more complete but still abstract graph representation of the underlying network(s) for informed traffic optimization among endpoints.

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 https://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 28 April 2022.

Copyright Notice
Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.
This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
Table of Contents

1. Introduction

Network performance metrics are crucial to assess the Quality of Experience (QoE) of ~~today's~~ applications. The ALTO protocol allows Internet Service Providers (ISPs) to provide guidance, such as topological distance between different end hosts, to overlay applications. Thus, the overlay applications can potentially improve the perceived QoE by better orchestrating their traffic to utilize the resources in the underlying network infrastructure.

Existing ALTO Cost Map (Section X of [RFC7285]) and Endpoint Cost Service (Section x of [RFC7285]) provide only cost information on an end-to-end path defined by its <source, destination> endpoints: The base protocol [RFC7285] allows the services to expose the topological distances of end-to-end paths, while various extensions have been proposed to extend the capability of these services, e.g., to express other performance metrics such as [I-D.ietf-alto-performance-metrics], to query multiple costs simultaneously [RFC8189], ~~and~~ or to obtain the time-varying values [RFC8896].

While the existing extensions are sufficient for many overlay applications, the QoE of some overlay applications depends not only on the cost information of end-to-end paths, but also on particular components of a network on the paths and their properties. For example, job completion time, which is an important QoE metric for a large-scale data analytics application, is impacted by shared bottleneck links inside the carrier network as link capacity may impact the rate of data input/output to the job. We refer to such components of a network as Abstract Network Elements (ANE). Predicting such information can be very complex without the help of ~~the~~ ISPs [BOXOPT]. With proper guidance from the ISP, an overlay application may be able to schedule its traffic for better QoE. In the meantime, it may be helpful as well for ISPs if applications could avoid using bottlenecks or challenging the network with poorly scheduled traffic.

Mis en forme : Surlignage

Mis en forme : Surlignage

Commenté [BMI1]: Wonder whether it is useful to mention that this information is dynamic and the assessment where this useful to share is not trivial.

Despite the claimed benefits, ISPs are not likely to expose details on their network paths: first for the sake of ~~confidentiality~~topology hiding requirement, second because it may increase volume and computation overhead, and last because it is difficult for ISPs to figure out what information and what details an application needs. Likewise, applications do not necessarily need all the network path details and are likely not able to understand them.

Therefore, it is beneficial for both parties if an ALTO server provides ALTO clients with an "abstract network state" that provides the necessary details to applications, while hiding the network complexity and confidential information. An "abstract network state" is a selected set of abstract representations of Abstract Network Elements traversed by the paths between <source, destination> pairs combined with properties of these Abstract Network Elements that are relevant to ~~the~~ an overlay applications' QoE. Both an application via its ALTO client and the ISP via the ALTO server can achieve better confidentiality and resource utilization by appropriately abstracting relevant Abstract Network Elements. Server scalability can also be improved by combining Abstract Network Elements and their properties in a single response.

This document extends [RFC7285] to allow an ALTO server to convey "abstract network state"~~,~~ for paths defined by their <source, destination> pairs. To this end, it introduces a new cost type called "Path Vector". A Path Vector is an array of identifiers that identifies an Abstract Network Element, which can be associated with various properties. The associations between ANEs and their properties are encoded in an ALTO information resource called Unified Property Map, which is specified in [I-D.ietf-alto-unified-props-new].

For better confidentiality, this document aims to minimize information exposure. In particular, this document enables and recommends that first ANEs are constructed on demand, and second an ANE is only associated with properties that are requested by an ALTO client. A Path Vector response involves two ALTO Maps: the Cost Map that contains the Path Vector results and the up-to-date Unified Property Map that contains the properties requested for these ANEs. To enforce consistency and improve server scalability, this document uses the "multipart/related" message defined in [RFC2387] to return the two maps in a single response.

~~The rest of the document is organized as follows. Section 3 introduces the extra terminologies that are used in this document. Section 4 uses an illustrative example to introduce the additional requirements of the ALTO framework, and discusses potential use cases. Section 5 gives an overview of the protocol design.~~

Commenté [BMI2]: Which ones?

Commenté [BMI3]: Redundant with the table of content.

~~Section 6 and Section 7 specify the extension to the ALTO IRD and the information resources, with some concrete examples presented in Section 8. Section 9 discusses the backward compatibility with the base protocol and existing extensions. Security and IANA considerations are discussed in Section 11 and Section 12 respectively.~~

2. Requirements Languages

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

When the words appear in lower case, they are to be interpreted with their natural language meanings.

3. Terminology

~~NOTE: This document depends on the Unified Property Map extension [I-D.ietf-alto-unified-props-new] and should be processed after the Unified Property Map document.~~

This document extends the ALTO base protocol [RFC7285] and the Unified Property Map extension [I-D.ietf-alto-unified-props-new]. In addition to the terms defined in these documents, this document also uses the following additional terms:

* Abstract Network Element (ANE): ~~An Abstract Network Element~~ is an abstract representation for a component in a network that handles data packets and whose properties can potentially have an impact on the end-to-end performance of traffic. An ANE can be a physical device such as a router, a link or an interface, or an aggregation of devices such as a subnetwork, or a data center.

  The definition of Abstract Network Element is similar to the Network Element defined in [RFC2216] in the sense that they both provide an abstract representation of ~~particular~~ specific components of a network.

  However, they have different criteria on how these ~~particular~~ components are selected. Specifically, a Network Element requires the components to be capable of exercising QoS control, while Abstract Network Element only requires the components to have an impact on the end-to-end performance.

* ANE Name: An ANE can be constructed either statically in advance or on demand based on the requested information. Thus, different ANEs may only be valid within a particular scope, either ephemeral

or persistent. Within each scope, an ANE is uniquely identified by an ANE Name, as defined in Section 6.1. Note that an ALTO client must not assume ANEs in different scopes but with the same ANE Name refer to the same component(s) of the network.

* Path Vector: ~~A Path Vector~~, or an ANE Path Vector, is a JSON array of ANE Names. It is a generalization of BGP path vector. While standard BGP path vector specifies a sequence of autonomous systems for a destination IP prefix, the Path Vector defined in this extension specifies a sequence of ANEs either for a source Provider-Defined Identifier (PID) and a destination PID as in the CostMapData object (Section 11.2.3.6 in [RFC7285]), or for a source endpoint and a destination endpoint as in the EndpointCostMapData object (Section 11.5.1.6 in [RFC7285]).
* Path Vector resource: ~~A Path Vector resource~~ refers to an ALTO resource which supports the extension defined in this document.
* Path Vector cost type: ~~The Path Vector cost type~~ is a special cost type, which is specified in Section 6.5. When this cost type is present in an IRD entry, it indicates that the information resource is a Path Vector resource. When this cost type is present in a Filtered Cost Map request or an Endpoint Cost Service request, it indicates each cost value must be interpreted as a Path Vector.
* Path Vector request: ~~A Path Vector request~~ refers to the POST message sent to an ALTO Path Vector resource.
* Path Vector response: ~~A Path Vector response~~ refers to the multipart/related message returned by a Path Vector resource.

4. ~~Problem Statement~~Requirements and Uses Cases

4.1. Design Requirements

This section gives an illustrative example of how an overlay application can benefit from the extension defined in this document. Assume that an application has control over a set of flows, which may go through shared links/nodes ~~or switches~~ and share bottlenecks. The application ~~hopes~~ seeks to schedule the traffic among multiple flows to get better performance. The capacity region information for those flows will benefit the scheduling. However, ~~existing~~ cost maps as defined in [RFC7285] can not reveal such information.

Specifically, consider a network as shown in Figure 1. The network has 7 switches (sw1 to sw7) forming a dumb-bell topology. Switches "sw1/sw3" provide access on one side, "sw2/sw4" provide access on the

Commenté [BMI4]: I guess you meant AS path here. Please add a pointer to rfc4271#section-5.1.2

Commenté [BMI5]: I don't find this term in 7285. Can you please cite the section where this is defined?

Commenté [BMI6]: As this can be generalized to any node type, including routers.

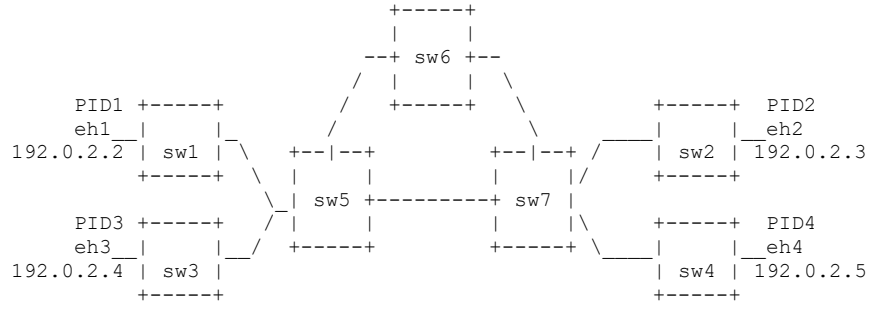
Commenté [BMI7]: ?

Not found in 7285

Commenté [BMI8]: I would reword to indicate that these are access switches.

"side" does not parse well to me.

other side, and sw5-sw7 form the backbone. End hosts eh1 to eh4 are connected to access switches sw1 to sw4 respectively. Assume that the bandwidth of link eh1 -> sw1 and link sw1 -> sw5 is 150 Mbps, and the bandwidth of the other links is 100 Mbps.



bw(eh1--sw1) = bw(sw1--sw5) = 150 Mbps
bw(eh2--sw2) = bw(eh3--sw3) = bw(eh4--sw4) = 100 Mbps
bw(sw1--sw5) = bw(sw3--sw5) = bw(sw2--sw7) = bw(sw4--sw7) = 100 Mbps
bw(sw5--sw6) = bw(sw5--sw7) = bw(sw6--sw7) = 100 Mbps

Figure 1: Raw Network Topology

The ~~single-node~~ ALTO topology abstraction of the network is shown in Figure 2. Assume the cost map returns ~~a~~ an hypothetical cost type representing the available bandwidth between a source and a destination.

Figure 2: Base ~~Single-Node~~ Topology Abstraction

Now assume the application wants to maximize the total rate of the traffic among a set of <source, destination> pairs, say "eh1 -> eh2" and "eh1 -> eh4". Let "x" denote the transmission rate of "eh1 -> eh2" and y denote the rate of "eh1 -> eh4". The objective function is

max(x + y).

With the ALTO Cost Map, the cost between PID1 and PID2 and between PID1 and PID4 will be 100 Mbps. ~~And T~~the client can get a capacity region of

x <= 100 Mbps,
y <= 100 Mbps.

With this information, the client may mistakenly think it can achieve a maximum total rate of 200 Mbps. However, ~~one can easily see that~~ this rate is infeasible~~,~~ as there are only two potential cases:

* Case 1: "eh1 -> eh2" and "eh1 -> eh4" take different path segments from sw5 to sw7. For example, if "eh1 -> eh2" uses path "eh1 -> sw1 -> sw5 -> sw6 -> sw7 -> sw2 -> eh2" and "eh1 -> eh4" uses path "eh1 -> sw1 -> sw5 -> sw7 -> sw4 -> eh4", then the shared bottleneck links are "eh1 -> sw1" and "sw1 -> sw5". In this case, the capacity region is :

  x <= 100 Mbps
  y <= 100 Mbps
  x + y <= 150 Mbps

  and the real optimal total rate is 150 Mbps.

* Case 2: "eh1 -> eh2" and "eh1 -> eh4" take the same path segment from "sw5 to sw7. For example, if zeh1 -> eh2z uses path zeh1 -> sw1 -> sw5 -> sw7 -> sw2 -> eh2z and zeh1 -> eh4z also uses path zeh1 -> sw1 -> sw5 -> sw7 -> sw4 -> eh4z, then the shared bottleneck link is zsw5 -> sw7z. In this case, the capacity region is:

  x <= 100 Mbps
  y <= 100 Mbps
  x + y <= 100 Mbps

  and the real optimal total rate is 100 Mbps.

Clearly, with more accurate and fine-grained information, the application can gain a better prediction of its traffic and may orchestrate its resources accordingly. However, to provide such information, the network needs to expose more details beyond the simple cost map abstraction. In particular:

* The ALTO server must ~~give~~ expose more details about the network paths that are traversed by the traffic between a source and a destination beyond a simple numerical value, which allows the overlay application to distinguish between Cases 1 and ~~Case~~ 2 and to compute the optimal total rate, accordingly.
* The ALTO server must allow the client to distinguish the common ANE shared by "eh1 -> eh2" and "eh1 -> eh4", e.g., "eh1 - sw1" and "sw1 - sw5" in Case 1.
* The ALTO server must ~~give~~ expose details on the properties of the ANEs

used by "eh1 -> eh2" and "eh1 -> eh4", e.g., the available bandwidth between "eh1 - sw1", "sw1 - sw5", "sw5 - sw7", "sw5 - sw6", "sw6 - sw7", "sw7 - sw2", s"w7 - sw4", "sw2 - eh2", "sw4 - eh4" in Case 1.

In general, we can conclude that to support the multiple flow scheduling use case, the ALTO framework must be extended to satisfy the following additional requirements:

AR1: An ALTO server must provide essential information on ANEs on the path of a <source, destination> pair that are critical to the QoE of the overlay application.

AR2: An ALTO server must provide essential information on how the paths of different <source, destination> pairs share a common ANE.

AR3: An ALTO server must provide essential information on the properties associated with the ANEs.

The extension defined in this document ~~proposes~~ specifies a solution to ~~provide~~expose these details.

4.2. Sample Use Cases

While the multiple flow scheduling problem is used to help identify the additional requirements, the extension defined in this document can be applied to a wide range of applications. This section highlights some real use cases that are reported.

4.2.1. Exposing Network Bottlenecks

An important use case of the Path Vector extension is to expose network bottlenecks. Applications such as large-scale data analytics can benefit from being aware of the resource constraints exposed by this extension even if they may have different optimization objectives.

Commenté [BMI9]: How "essential" is technical characterized?

Commenté [BMI10]: Were those reported by "real" deployments?

If not, I would suggest we keep the wording factual.

Figure 3 illustrates an example of using ALTO Path Vector as an standard interface between the job optimizer for a data analytics system and the network manager. In particular, we assume the objective of the job optimizer is to minimize the job completion time.

In this such a setting, the network-aware job optimizer (e.g., [CLARINET]) takes a query and generates multiple query execution plans (QEB). It can encode the QEBs as Path Vector requests and that are send sent to the an ALTO server. The ALTO server obtains the routing information for the flows in a QEP QEB and finds links, routers, or middleboxes (e.g., a stateful firewall) that can potentially become bottlenecks of the QEBQEP (see, e.g., [NOVA] and [G2] for mechanisms to identify bottleneck links under different settings). The resource constraint information is encoded in a Path Vector response and returned to the ALTO client. With the network resource constraints, the job optimizer may choose the QEP QEB with the optimal job completion time to be executed. It must be noted that the ALTO framework itself does not offer the capability to control the traffic. However, certain network managers may offer ways to enforce resource guarantees, such as on-demand tunnels (e.g., [SWAN]), demand vector (e.g., [HUG], [UNICORN]), etc. The traffic control interfaces and mechanisms are out of the scope of this document.

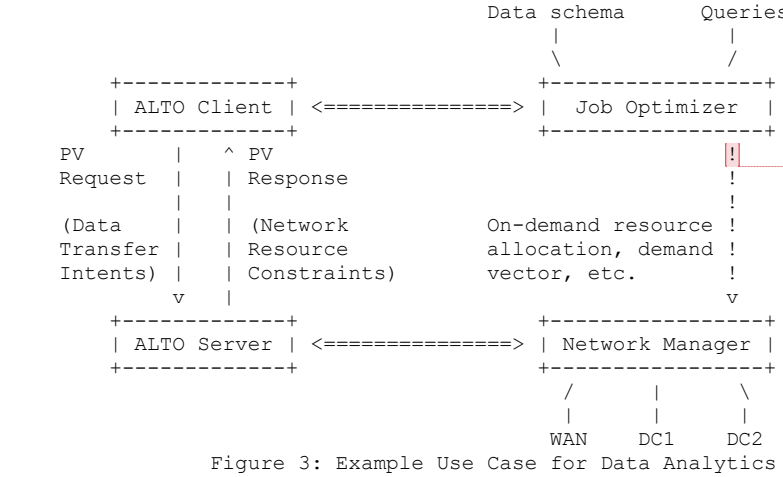
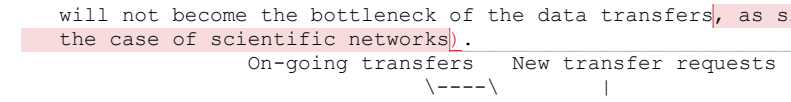


Figure 3: Example Use Case for Data Analytics

Another example is as illustrated in Figure 4. Consider a network consisting of multiple sites and a non-blocking core network (, i.e., the links in the core network have sufficient bandwidth that they will not become the bottleneck of the data transfers, as similar to the case of scientific networks).



Commenté [BMI11]: Any specific meaning compared to other links?

Commenté [BMI12]: Not sure I would keep this.

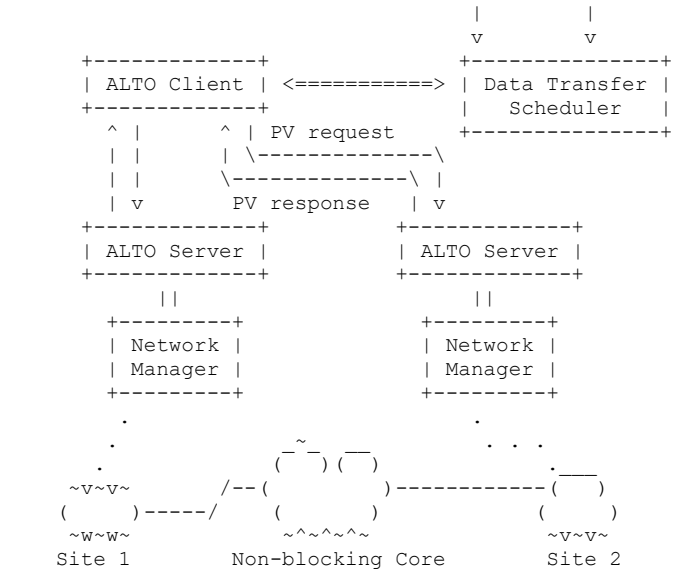


Figure 4: Example Use Case for Cross-site Bottleneck Discovery

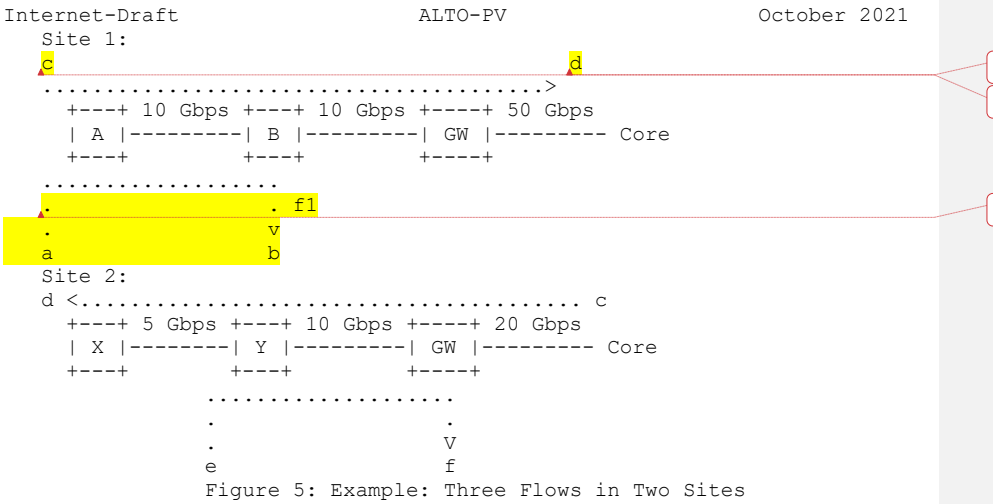


Figure 5: Example: Three Flows in Two Sites

With the Path Vector extension, a site can reveal the bottlenecks inside its own network with necessary information (such as link capacities) to the ALTO client, instead of providing the full topology and routing information. The bottleneck information can be used to analyze the impact of adding/removing data transfer flows, e.g., using the [G2] framework. For example, assume hosts a, b, c are in site 1 hosts d, e, f are in site 2, and there are 3 flows in two sites: a -> b, c -> d, e -> f. For these flows, site 1 returns:

a: { b: [ane1] },
c: { d: [ane1, ane2, ane3] }
ane1: bw = 10 Gbps (link: A->B)
ane2: bw = 10 Gbps (link: B->GW)
ane3: bw = 50 Gbps (link: GW->Core)

and site 2 returns:

Mis en forme : Surlignage
Mis en forme : Surlignage
Mis en forme : Surlignage

c: { d: [anei, aneii, aneiii] }
e: { f: [aneiv] }
anei: bw = 5 Gbps (link Y->X)
aneii: bw = 10 Gbps (link GW->Y)
aneiii: bw = 20 Gbps (link Core->GW)
aneiv: bw = 10 Gbps (link Y->GW)

With the information, the data transfer scheduler can use algorithms such as the theory on bottleneck structure [G2] to predict the potential throughput of the flows.

4.2.2. Resource Exposure for CDN and Service Edge

A growing trend in today's applications (2021) is to bring storage and computation closer to the end users for better QoE, such as Content Delivery Network (CDN), AR/VR, and cloud gaming, as reported in various documents (e.g., [SEREDGE] and [MOWIE]). Internet Service Providers may deploy multiple layers of CDN caches, or more generally service edges, with different latency and available resources.

For example, ~~the figure below~~Figure6 illustrates a typical edge-cloud scenario. The "on-premise" edge nodes are closest to the end hosts and have the smallest latency, and the site-radio edge node and access central office (CO) have larger latency but more available resources.

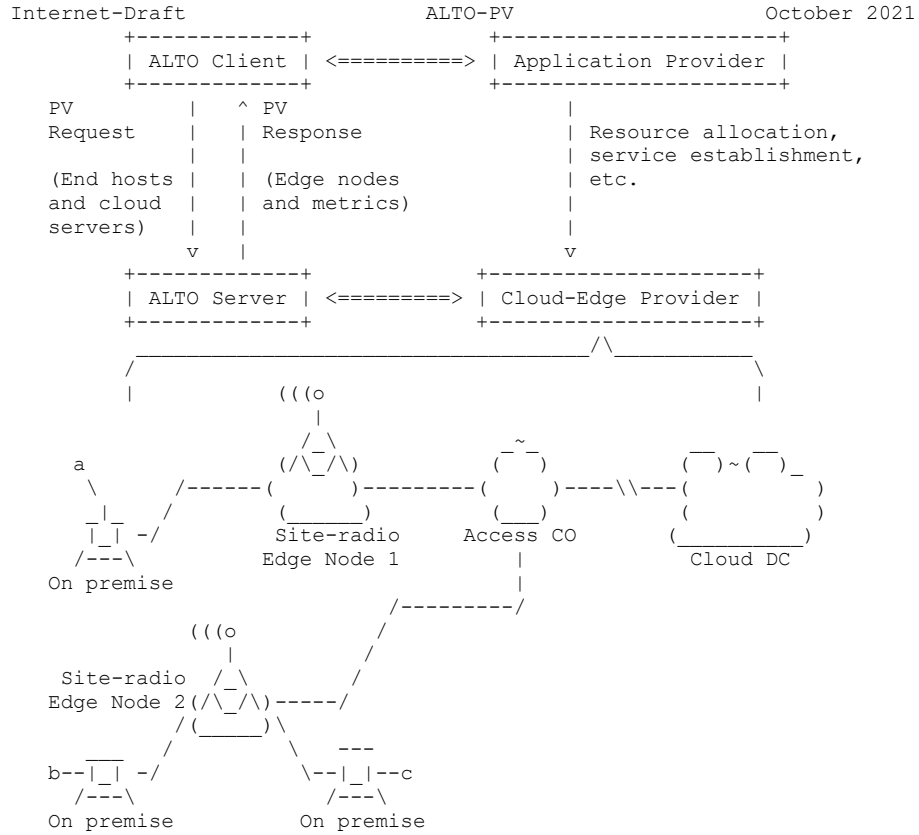


Figure 6: Example Use Case for Service Edge Exposure

```
   a: { b: [ane1, ane2, ane3, ane4, ane5],
        c: [ane1, ane2, ane3, ane4, ane6],
        DC: [ane1, ane2, ane3] }
   b: { c: [ane5, ane4, ane6], DC: [ane5, ane4, ane3] }
   ane1: latency=5ms cpu=2 memory=8G storage=10T
   (on premise, a)
   ane2: latency=20ms cpu=4 memory=8G storage=10T
   (Site-radio Edge Node 1)
   ane3: latency=100ms cpu=8 memory=128G storage=100T
   (Access CO)
   ane4: latency=20ms cpu=4 memory=8G storage=10T
   (Site-radio Edge Node 2)
   ane5: latency=5ms cpu=2 memory=8G storage=10T
   (on premise, b)
   ane6: latency=5ms cpu=2 memory=8G storage=10T
   (on premise, c)
```

Figure 7: Example Service Edge Query Results

With the extension defined in this document, an ALTO server can selectively reveal the CDNs and service edges that reside along the paths between different end hosts and/or the cloud servers, together with their properties such as capabilities (e.g., storage, GPU) and available Service Level Agreement (SLA) plans. See Figure 7 for an example where the query is made for sources [a, b] and destinations [b, c, DC]. Here each ANE represents a service edge and the properties include access latency, available resources, etc. Note the properties here are only used for illustration purposes and are not part of this extension.

With the service edge information, an ALTO client may better conduct CDN request routing or offload functionalities from the user equipment to the service edge, with considerations on customized quality of experience.

5. Path Vector Extension: Overview

This section ~~gives~~ provides a non-normative overview of the Path Vector extension ~~defined in this document~~. It is assumed that the readers are familiar with both the base protocol [RFC7285] and the Unified Property Map extension [I-D.ietf-alto-unified-props-new].

Commenté [BMI13]: Call-out the meaning and the units in the description text.

To ~~satisfies~~ satisfy the additional requirements listed in Section 4.1, this extension:

1. introduces the concept of Abstract Network Element (ANE) as the abstraction of components in a network whose properties may have an impact on the end-to-end performance of the traffic handled by those ~~component~~components,
2. extends the Cost Map and Endpoint Cost Service to convey the ANEs traversed by the path of a <source, destination> pair as Path Vectors, and
3. uses the Unified Property Map to convey the association between the ANEs and their properties.

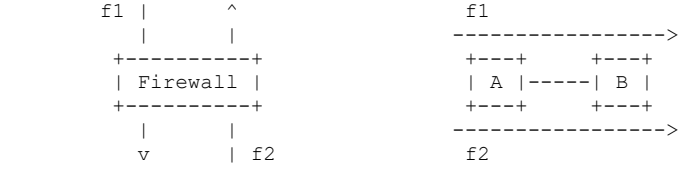
Thus, an ALTO client can learn about the ANEs that are critical to the QoE of a <source, destination> pair by investigating the corresponding Path Vector value (AR1), identify common ANEs if an ANE appears in the Path Vectors of multiple <source, destination> pairs (AR2), and retrieve the properties of the ANEs by searching the Unified Property Map (AR3).

Commenté [BMI14]: How these are identified as being critical?

5.1. Abstract Network Element (ANE)

This extension introduces ~~Abstract Network Element (~~ANE~~)~~ as an indirect and network-agnostic way to specify a component or an aggregation of components of a network whose properties have an impact on the end-to-end performance for application traffic between a ~~source and a destination~~application endpoints.

~~Abstract network elements~~ANEs allow ALTO servers to focus on common properties of different types of network components. For example, the throughput of a flow can be constrained by different components in a network: the capacity of a physical link, the maximum throughput of a firewall, the reserved bandwidth of an MPLS tunnel, etc. See the example below, assume the throughput of the firewall is 100 Mbps and the capacity for link (A, B) is also 100 Mbps, they result in the same constraint on the total throughput of f1 and f2. Thus, they are identical when treated as an ANE.



When an ANE is defined by ~~the~~ an ALTO server, it is assigned an identifier, i.e., a string of type ANEName ~~(as specified in~~ Section 6.1~~)~~, and a set of associated properties.

5.1.1. ANE Entity Domain

In this extension, the associations between an ANE and the properties are conveyed in a Unified Property Map. Thus, ANEs ~~must~~ constitute an entity domain (Section 5.1 of [I-D.ietf-alto-unified-props-new]), and each ANE property must be an entity property (Section 5.2 of [I-D.ietf-alto-unified-props-new]).

Specifically, this document defines a new entity domain called "ane" as specified in Section 6.2 and defines two initial properties for the ANE entity domain.

5.1.2. Ephemeral ~~ANE~~ and Persistent ANEs

By design, ANEs are ephemeral and not to be used in further requests to other ALTO resources. More precisely, the corresponding ANE names are no longer valid beyond the scope of ~~the~~ a Path Vector response or the incremental update stream for a Path Vector request. This has several benefits including better privacy of the ISPs and more flexible ANE computation.

For example, an ALTO server may define an ANE for each aggregated bottleneck link between the sources and destinations specified in the request. For requests with different sources and destinations, the bottlenecks may be different but can safely reuse the same ANE names. The client can still adjust its traffic based on the information but is difficult to infer the underlying topology with multiple queries.

However, sometimes an ISP may intend to selectively reveal some "persistent" network components which, opposite to being ephemeral, have a longer life cycle. For example, an ALTO server may define an ANE for each service edge cluster. Once a client chooses to use a service edge, e.g., by deploying some user-defined functions, it may want to stick to the service edge to avoid the complexity of state transition or synchronization, and continuously query the properties of the edge cluster.

This document provides a mechanism to expose such network components as persistent ANEs. A persistent ANE has a persistent ID that is registered in a Property Map, together with their properties. See Sections 6.2.4 and ~~Section~~ 6.4.2 for more detailed instructions on how to identify ephemeral ANEs and persistent ANEs.

Commenté [BMI15]: by the server. Please say so in the text.

Commenté [BMI16]: Not sure about this because cumulated expose data can be used to track a network/ANE status.

5.1.3. Property Filtering

Resource-constrained ALTO clients may benefit from the filtering of Path Vector query results at the ALTO server, as an ALTO client may only require a subset of the available properties.

Specifically, the available properties for a given resource are announced in the Information Resource Directory as a new capability called "ane-property-names". The selected properties are specified in a filter called "ane-property-names" in the request body, and the response includes and only includes the selected properties for the ANEs in the response.

The "ane-property-names" capability for Cost Map and for Endpoint Cost Service is specified in Sections 7.2.4 and ~~Section~~ 7.3.4 respectively. The "ane-property-names" filter for Cost Map and Endpoint Cost Service is specified in Sections 7.2.3 and ~~Section~~ 7.3.3 accordingly.

5.2. Path Vector Cost Type

For an ALTO client to correctly interpret the Path Vector, this extension specifies a new cost type called the Path Vector cost type. The Path Vector cost type must convey both the interpretation and semantics in the "cost-mode" and "cost-metric" respectively.

Unfortunately, a single "cost-mode" value cannot fully specify the interpretation of a Path Vector, which is a compound data type. For example, in programming languages such as C++, a Path Vector will have the type of JSONArray<ANEName>.

Instead of extending the "type system" of ALTO, this document takes a simple and backward compatible approach. Specifically, the "cost-mode" of the Path Vector cost type is "array", which indicates the value is a JSON array. Then, an ALTO client must check the value of the "cost-metric". If the value is "ane-path", it means that the JSON array should be further interpreted as a path of ANENames.

The Path Vector cost type is specified in Section 6.5.

5.3. Multipart Path Vector Response

For a basic ALTO information resource, a response contains only one type of ALTO resources, e.g., Network Map, Cost Map, or Property Map. Thus, only one round of communication is required: An ALTO client sends a request to an ALTO server, and the ALTO server returns a response, as shown in Figure 8.

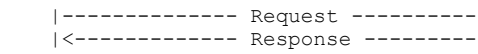


Figure 8: A Typical ALTO Request and Response

The extension defined in this document, on the other hand, involves two types of information resources: Path Vectors conveyed in an InfoResourceCostMap (defined in Section 11.2.3.6 of [RFC7285]) or an InfoResourceEndpointCostMap (defined in Section 11.5.1.6 of [RFC7285]), and ANE properties conveyed in an InfoResourceProperties (defined in Section 7.6 of [I-D.ietf-alto-unified-props-new]). Instead of two consecutive message exchanges, the extension defined in this document enforces one round of communication. Specifically, the ALTO client must include the source and destination pairs and the requested ANE properties in a single request, and the ALTO server must return a single response containing both the Path Vectors and properties associated with the ANEs in the Path Vectors, as shown in Figure 9. Since the two parts are bundled together in one response message, their orders are interchangeable. See Sections 7.2.6 and ~~Section~~ 7.3.6 for details.

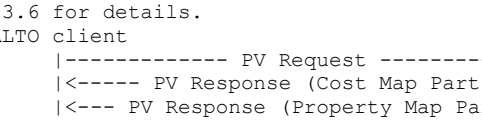


Figure 9: The Path Vector Extension Request and Response

This design is based on the following considerations:

1. Since ANEs may be constructed on demand, and potentially based on the requested properties (See Section 5.1 for more details). If sources and destinations are not in the same request as the properties, an ALTO server either cannot construct ANEs on-demand, or must wait until both requests are received.
2. As ANEs may be constructed on demand, mappings of each ANE to its underlying network devices and resources can be specific to the request. In order to respond to the Property Map request correctly, an ALTO server must store the mapping of each Path Vector request until the client fully retrieves the property information. The "stateful" behavior may substantially harm the server scalability and potentially lead to Denial-of-Service attacks.

One approach to realize the one-round communication is to define a new media type to contain both objects, but this violates modular design. This document follows the standard-conforming usage of "multipart/related" media type defined in [RFC2387] to elegantly combine the objects. Path Vectors are encoded in an InfoResourceCostMap or an InfoResourceEndpointCostMap, and the Property Map is encoded in an InfoResourceProperties. They are encapsulated as parts of a multipart message. The modular composition allows ALTO servers and clients to reuse the data models of the existing information resources. Specifically, this document addresses the following practical issues using "multipart/related".

5.3.1. Identifying the Media Type of the Root Object

ALTO uses media type to indicate the type of an entry in the Information Resource Directory (IRD) (e.g., "application/alto-costmap+json" for Cost Map and "application/alto-endpointcost+json" for Endpoint Cost Service). Simply putting "multipart/related" as the media type, however, makes it impossible for an ALTO client to identify the type of service provided by related entries.

To address this issue, this document uses the "type" parameter to indicate the root object of a multipart/related message. For a Cost Map resource, the "media-type" field in the IRD entry is "multipart/related" with the parameter "type=application/alto-costmap+json"; for an Endpoint Cost Service, the parameter is "type=application/alto-endpointcost+json".

5.3.2. References to Part Messages

As the response of a Path Vector resource is a multipart message with two different parts, it is important that each part can be uniquely identified. Following the designs of [RFC8895], this extension requires that an ALTO server assigns a unique identifier to each part of the multipart response message. This identifier, referred to as a Part Resource ID (See Section 6.6 for details), is present in the part message's "Content-ID" header. By concatenating the Part Resource ID to the identifier of the Path Vector request, an ALTO server/client can uniquely identify the Path Vector Part or the Property Map part.

6. Specification: Basic Data Types

6.1. ANE Name

An ANE Name is encoded as a JSON string with the same format as that of the type PIDName (Section 10.1 of [RFC7285]).

The type ANEName is used in this document to indicate a string of this format.

6.2. ANE Domain

The ANE domain associates property values with the Abstract Network Elements in a Property Map. Accordingly, the ANE domain always depends on a Property Map.

It must be noted that the term "domain" here does not refer to a network domain. Rather, it is inherited from the "entity domain" defined in ~~Sec~~ Section 3.2 in [I-D.ietf-alto-unified-props-new] that represents the set of valid entities defined by an ALTO information resource (called the defining information resource).

Commenté [BMI17]: Why not simply using « entity domain" and avoid this confusion?

6.2.1. Entity Domain Type

ane

6.2.2. Domain-Specific Entity Identifier

The entity identifiers are the ANE Names in the associated Property Map.

6.2.3. Hierarchy and Inheritance

There is no hierarchy or inheritance for properties associated with ANEs.

6.2.4. Media Type of Defining Resource

The defining resource for entity domain type "ane" MUST be a Property Map, i.e., the media type of defining resources is: application/alto-propmap+json

Specifically, for ephemeral ANEs that appear in a Path Vector response, their entity domain names MUST be exactly ".ane" and the defining resource of these ANEs is the Property Map part of the multipart response. Meanwhile, for persistent ANEs whose entity domain name has the format of "PROPMAP.ane" where PROPMAP is the name of a Property Map resource, PROPMAP is the defining resource of these ANEs. Persistent entities are "persistent" because standalone queries can be made by an ALTO client to their defining resources when the connection to the Path Vector service is closed.

For example, the defining resource of an ephemeral ANE whose entity identifier is ".ane:NET1" is the Property Map part that contains this identifier. The defining resource of a persistent ANE whose entity identifier is "dc-props.ane:DC1" is the Property Map with the resource ID "dc-props".

6.3. ANE Property Name

An ANE Property Name is encoded as a JSON string with the same format as that of Entity Property Name (Section 5.2.2 of [I-D.ietf-alto-unified-props-new]).

6.4. Initial ANE Property Types

~~In this document, two~~ Two initial ANE property types are specified~~,~~ : "max-reservable-bandwidth" and "persistent-entity-id".

Note that these two property types ~~defined in this document~~ do not depend on any information resource~~,~~. As such, ~~so~~ their ResourceID part must be empty.

6.4.1. Maximum Reservable Bandwidth

The maximum reservable bandwidth property ("max-reservable-bandwidth") stands for the maximum bandwidth that can be reserved for all the traffic that traverses an ANE. The value MUST be encoded as a non-negative numerical cost value as defined in Section 6.1.2.1 of [RFC7285] and the unit is bit per second (bps). If this property is requested by the ALTO client but not present for an ANE in the server response, it MUST be interpreted as that the property is not defined for the ANE.

This property can be offered in a setting where the ALTO server is part of a network system that provides on-demand resource allocation and the ALTO client is part of a user application. One existing example is [NOVA]: the ALTO server is part of an SDN controller and exposes a list of traversed network elements and associated link bandwidth to the client. The encoding in [NOVA] differs from the Path Vector response defined in this document that the Path Vector part and Property Map part are put in the same JSON object.

In such a framework, the ALTO server exposes resource (e.g., reservable bandwidth) availability information to the ALTO client. How the client makes resource requests based on the information and how the resource allocation is achieved respectively depend on interfaces between the management system and the users or a higher-layer protocol (e.g., SDN network intents or MPLS tunnels), which are out of the scope of this document.

6.4.2. Persistent Entity ID

The persistent entity ID property is the entity identifier of the persistent ANE which an ephemeral ANE presents (See Section 5.1.2 for details). The value of this property is encoded with the format EntityID defined in Section 5.1.3 of [I-D.ietf-alto-unified-props-new].

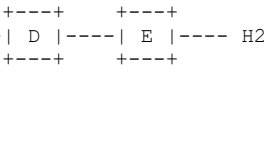
In this format, the entity ID combines:

* a defining information resource for the ANE on which a "persistent-entity-id" is queried, which is the Property Map resource defining the ANE as a persistent entity, together with the properties;
* the persistent name of the ANE in that Property Map.

With this format, the client has all the needed information for further standalone query properties on the persistent ANE.

6.4.3. Examples

To illustrate the use of "max-reservable-bandwidth", consider the following network with 5 nodes. Assume the client wants to query the maximum reservable bandwidth from H1 to H2. An ALTO server may split the network into two ANEs: "ane1" that represents the subnetwork with routers A, B, and C, and "ane2" that represents the subnetwork with routers B, D and E. The maximum reservable bandwidth for "ane1" is 15 Mbps (using path A->C->B) and the maximum reservable bandwidth for "ane2" is 20 Mbps (using path B->D->E).



To illustrate the use of "persistent-entity-id", consider the scenario in Figure 6. As the life cycle of service edges are typically long, they may contain information that is not specific to the query. Such information can be stored in an individual unified property map and later be accessed by an ALTO client.

For example, "ane1" in Figure 7 represents the on-premise service edge closest to host a. Assume the properties of the service edges are provided in a unified property map called "se-props" and the ID of the on-premise service edge is "9a0b55f7-7442-4d56-8a2c-b4cc6a8e3aa1", the "persistent-entity-id" of "ane1" will be "se-props.ane:9a0b55f7-7442-4d56-8a2c-b4cc6a8e3aa1". With this persistent entity ID, an ALTO client may send queries to the "se-props" resource with the entity ID ".ane:9a0b55f7-7442-4d56-8a2c-b4cc6a8e3aa1".

6.5. Path Vector Cost Type

This document defines a new cost type, which is referred to as the Path Vector cost type. An ALTO server MUST offer this cost type if it supports the extension defined in this document.

6.5.1. Cost Metric: ane-path

The cost metric "ane-path" indicates the value of such a cost type conveys an array of ANE names, where each ANE name uniquely represents an ANE traversed by traffic from a source to a destination.

An ALTO client MUST interpret the Path Vector as if the traffic between a source and a destination logically traverses the ANEs in the same order as they appear in the Path Vector.

6.5.2. Cost Mode: array

The cost mode "array" indicates that every cost value in the response body of a (Filtered) Cost Map or an Endpoint Cost Service MUST be interpreted as a JSON array object.

Note that this cost mode only requires the cost value to be a JSON array of JSONValue. However, an ALTO server that enables this extension MUST return a JSON array of ANEName (Section 6.1) when the cost metric is "ane-path".

6.6. Part Resource ID and Part Content ID

A Part Resource ID is encoded as a JSON string with the same format as that of the type ResourceID (Section 10.2 of [RFC7285]).

Even though the client-id assigned to a Path Vector request and the Part Resource ID MAY contain up to 64 characters by their own definition, their concatenation (~~see~~ Section 5.3.2) MUST also conform to the same length constraint. The same requirement applies to the resource ID of the Path Vector resource, too. Thus, it is RECOMMENDED to limit the length of resource ID and client ID related to a Path Vector resource to 31 characters.

A Part Content ID conforms to the format of "msg-id" as specified in [RFC2387] and [RFC5322]. Specifically, it has the following format: "<" PART-RESOURCE-ID "@" DOMAIN-NAME ">"

PART-RESOURCE-ID: PART-RESOURCE-ID has the same format as the Part Resource ID. It is used to identify whether a part message is a Path Vector or a Property Map.

DOMAIN-NAME: DOMAIN-NAME has the same format as dot-atom-text specified in Section 3.2.3 of [RFC5322]. It must be the domain name of the ALTO server.

7. Specification: Service Extensions

7.1. Notations

This document uses the same syntax and notations as introduced in Section 8.2 of RFC 7285 [RFC7285] to specify the extensions to existing ALTO resources and services.

7.2. Multipart Filtered Cost Map for Path Vector

This document introduces a new ALTO resource called multipart Filtered Cost Map resource, which allows an ALTO server to provide other ALTO resources associated with the Cost Map resource in the same response.

7.2.1. Media Type

The media type of the multipart Filtered Cost Map resource is "multipart/related;type=application/alto-costmap+json".

7.2.2. HTTP Method

The multipart Filtered Cost Map is requested using the HTTP POST method.

7.2.3. Accept Input Parameters

The input parameters of the multipart Filtered Cost Map are supplied in the body of an HTTP POST request. This document extends the input parameters to a Filtered Cost Map, which is defined as a JSON object of type ReqFilteredCostMap in Section 4.1.2 of ~~RFC 8189~~ [RFC8189], with a data format indicated by the media type "application/alto-costmapfilter+json", which is a JSON object of type PVReqFilteredCostMap:

```
object {
  [EntityPropertyName ane-property-names<0..*>;]
} PVReqFilteredCostMap : ReqFilteredCostMap;
```

with fields:

ane-property-names: A list of selected ANE properties to be included in the response. Each property in this list MUST match one of the supported ANE properties indicated in the resource's "ane-property-names" capability (~~See~~ Section 7.2.4). If the field is ~~NOT~~ not present, it MUST be interpreted as an empty list.

Example: Consider the network in Figure 1. If an ALTO client wants to query the "max-reservable-bandwidth" between PID1 and PID2, it can submit the following request.

```
POST /costmap/pv HTTP/1.1
Host: alto.example.com
Accept: multipart/related;type=application/alto-costmap+json,
        application/alto-error+json
Content-Length: 201
Content-Type: application/alto-costmapfilter+json
{
  "cost-type": {
    "cost-mode": "array",
    "cost-metric": "ane-path"
  },
  "pids": {
    "srcs": [ "PID1" ],
    "dsts": [ "PID2" ]
  },
  "ane-property-names": [ "max-reservable-bandwidth" ]
}
```

7.2.4. Capabilities

The multipart Filtered Cost Map resource extends the capabilities defined in Section 4.1.1 of [RFC8189]. The capabilities are defined by a JSON object of type PVFilteredCostMapCapabilities:

```
object {
  [EntityPropertyName ane-property-names<0..*>;]
} PVFilteredCostMapCapabilities : FilteredCostMapCapabilities;
```

with fields:

cost-type-names: The "cost-type-names" field MUST include the Path Vector cost type, unless explicitly documented by a future extension. This also implies that the Path Vector cost type MUST be defined in the "cost-types" of the Information Resource Directory's "meta" field.

cost-constraints: If the "cost-type-names" field includes the Path Vector cost type, "cost-constraints" field MUST be "false" or not present unless specifically instructed by a future document.

testable-cost-type-names (Section 4.1.1 of [RFC8189]): If the "cost-type-names" field includes the Path Vector cost type and the "testable-cost-type-names" field is present, the Path Vector cost type MUST NOT be included in the "testable-cost-type-names" field unless specifically instructed by a future document.

ane-property-names: Defines a list of ANE properties that can be returned. If the field is ~~NOT~~ not present, it MUST be interpreted as an empty list, indicating the ALTO server cannot provide any ANE property.

Commenté [BMI18]: Explicit where the extension was defined.

7.2.5. Uses

This member MUST include the resource ID of the network map based on which the PIDs are defined. If this resource supports "persistent-entity-id", it MUST also include the defining resources of persistent ANEs that may appear in the response.

7.2.6. Response

The response MUST indicate an error, using ALTO protocol error handling, as defined in Section 8.5 of [RFC7285], if the request is invalid.

The "Content-Type" header of the response MUST be "multipart/related" as defined by [RFC2387] with the following parameters:

type: The type parameter MUST be "application/alto-costmap+json". Note that [RFC2387] permits both parameters with and without the double quotes.

start: The start parameter is as defined in [RFC2387]. If present, it MUST have the same value as the "Content-ID" header of the Path Vector part.

boundary: The boundary parameter is as defined in [RFC2387].

The body of the response MUST consist of two parts:

* The Path Vector part MUST include "Content-ID" and "Content-Type" in its header. The "Content-Type" MUST be "application/alto-costmap+json". The value of "Content-ID" MUST have the same format as the Part Content ID as specified in Section 6.6.
  The body of the Path Vector part MUST be a JSON object with the same format as defined in Section 11.2.3.6 of [RFC7285] when the "cost-type" field is present in the input parameters and MUST be a JSON object with the same format as defined in Section 4.1.3 of [RFC8189] if the "multi-cost-types" field is present. The JSON object MUST include the "vtag" field in the "meta" field, which provides the version tag of the returned CostMapData. The resource ID of the version tag MUST follow the format of
  resource-id '.' part-resource-id
  where "resource-id" is the resource Id of the Path Vector resource, and "part-resource-id" has the same value as the PART-RESOURCE-ID in the "Content-ID" of the Path Vector part. The "meta" field MUST also include the "dependent-vtags" field, whose value is a single-element array to indicate the version tag of the network map used, where the network map is specified in the "uses" attribute of the multipart Filtered Cost Map resource in IRD.
* The Unified Property Map part MUST also include "Content-ID" and "Content-Type" in its header. The "Content-Type" MUST be "application/alto-propmap+json". The value of "Content-ID" MUST have the same format as the Part Content ID as specified in Section 6.6.
  The body of the Unified Property Map part is a JSON object with the same format as defined in Section 4.6 of [I-D.ietf-alto-unified-props-new]. The JSON object MUST include the "dependent-vtags" field in the "meta" field. The value of the "dependent-vtags" field MUST be an array of VersionTag objects as defined by Section 10.3 of [RFC7285]. The "vtag" of the Path

Vector part MUST be included in the "dependent-vtags". If "persistent-entity-id" is requested, the version tags of the dependent resources that MAY expose the entities in the response MUST also be included.

The PropertyMapData has one member for each ANEName that appears in the Path Vector part, which is an entity identifier belonging to the self-defined entity domain as defined in Section 5.1.2.3 of [I-D.ietf-alto-unified-props-new]. The EntityProps for each ANE has one member for each property that is both 1) associated with the ANE, and 2) specified in the "ane-property-names" in the request. If the Path Vector cost type is not included in the "cost-type" field or the "multi-cost-type" field, the "property-map" field MUST be present and the value MUST be an empty object ({}).

A complete and valid response MUST include both the Path Vector part and the Property Map part in the multipart message. If any part is NOT present, the client MUST discard the received information and send another request if necessary.

According to [RFC2387], the Path Vector part, whose media type is the same as the "type" parameter of the multipart response message, is the root object. Thus, it is the element the application processes first. Even though the "start" parameter allows it to be placed anywhere in the part sequence, it is RECOMMENDED that the parts arrive in the same order as they are processed, i.e., the Path Vector part is always put as the first part, followed by the Property Map part. When doing so, an ALTO server MAY choose not to set the "start" parameter, which implies the first part is the root object.

Example: Consider the network in Figure 1. The response of the example request in Section 7.2.3 is as follows, where "ANE1" represents the aggregation of all the switches in the network.

```
   HTTP/1.1 200 OK
   Content-Length: 821
   Content-Type: multipart/related; boundary=example-1;
                 type=application/alto-costmap+json
   --example-1
   Content-ID: <costmap@alto.example.com>
   Content-Type: application/alto-costmap+json
   {
     "meta": {
       "vtag": {
         "resource-id": "filtered-cost-map-pv.costmap",
         "tag": "d827f484cb66ce6df6b5077cb8562b0a"
       },
       "dependent-vtags": [
         {
           "resource-id": "my-default-networkmap",
           "tag": "75ed013b3cb58f896e839582504f6228"
         }
       ],
       "cost-type": { "cost-mode": "array", "cost-metric": "ane-path" }
     },
     "cost-map": {
       "PID1": { "PID2": ["ANE1"] }
     }
   }
   --example-1
   Content-ID: <propmap@alto.example.com>
   Content-Type: application/alto-propmap+json
   {
     "meta": {
       "dependent-vtags": [
         {
           "resource-id": "filtered-cost-map-pv.costmap",
           "tag": "d827f484cb66ce6df6b5077cb8562b0a"
         }
       ]
     },
     "property-map": {
       ".ane:ANE1": { "max-reservable-bandwidth": 100000000 }
     }
   }
```

7.3. Multipart Endpoint Cost Service for Path Vector

This document introduces a new ALTO resource called multipart Endpoint Cost Service, which allows an ALTO server to provide other ALTO resources associated with the Endpoint Cost Service resource in the same response.

7.3.1. Media Type

The media type of the multipart Endpoint Cost Service resource is "multipart/related;type=application/alto-endpointcost+json".

7.3.2. HTTP Method

The multipart Endpoint Cost Service resource is requested using the HTTP POST method.

7.3.3. Accept Input Parameters

The input parameters of the multipart Endpoint Cost Service resource are supplied in the body of an HTTP POST request. This document extends the input parameters to an Endpoint Cost Service, which is defined as a JSON object of type ReqEndpointCost in Section 4.2.2 in RFC 8189 [RFC8189], with a data format indicated by the media type "application/alto-endpointcostparams+json", which is a JSON object of type PVReqEndpointCost:

```
object {
  [EntityPropertyName ane-property-names<0..*>;]
} PVReqEndpointcost : ReqEndpointcostMap;
```

with fields:

ane-property-names: This document defines the "ane-property-names" in PVReqEndpointcost as the same as in PVReqFilteredCostMap. See Section 7.2.3.

Example: Consider the network in Figure 1. If an ALTO client wants to query the "max-reservable-bandwidth" between eh1 and eh2, it can submit the following request.

```
POST /ecs/pv HTTP/1.1
Host: alto.example.com
Accept: multipart/related;type=application/alto-endpointcost+json,
        application/alto-error+json
Content-Length: 222
Content-Type: application/alto-endpointcostparams+json
{
  "cost-type": {
    "cost-mode": "array",
    "cost-metric": "ane-path"
  },
  "endpoints": {
    "srcs": [ "ipv4:192.0.2.2" ],
    "dsts": [ "ipv4:192.0.2.18" ]
  },
  "ane-property-names": [ "max-reservable-bandwidth" ]
}
```

7.3.4. Capabilities

The capabilities of the multipart Endpoint Cost Service resource are defined by a JSON object of type PVEndpointcostCapabilities, which is defined as the same as PVFilteredCostMapCapabilities. See Section 7.2.4.

7.3.5. Uses

If this resource supports "persistent-entity-id", it MUST also include the defining resources of persistent ANEs that may appear in the response.

7.3.6. Response

The response MUST indicate an error, using ALTO protocol error handling, as defined in Section 8.5 of [RFC7285], if the request is invalid.

The "Content-Type" header of the response MUST be "multipart/related" as defined by [RFC7285] with the following parameters:

type: The type parameter MUST be "application/alto-endpointcost+json".

start: The start parameter is as defined in Section 7.2.6.

boundary: The boundary parameter is as defined in [RFC2387].

The body MUST consist of two parts:

* The Path Vector part MUST include "Content-ID" and "Content-Type" in its header. The "Content-Type" MUST be "application/alto-endpointcost+json". The value of "Content-ID" MUST have the same format as the Part Content ID as specified in Section 6.6.
  The body of the Path Vector part MUST be a JSON object with the same format as defined in Section 11.5.1.6 of [RFC7285] when the "cost-type" field is present in the input parameters and MUST be a JSON object with the same format as defined in Section 4.1.3 of [RFC8189] if the "multi-cost-types" field is present. The JSON object MUST include the "vtag" field in the "meta" field, which provides the version tag of the returned EndpointCostMapData. The resource ID of the version tag MUST follow the format of
  resource-id '.' part-resource-id
  where "resource-id" is the resource Id of the Path Vector resource, and "part-resource-id" has the same value as the PART-RESOURCE-ID in the "Content-ID" of the Path Vector part.
* The Unified Property Map part MUST also include "Content-ID" and "Content-Type" in its header. The "Content-Type" MUST be "application/alto-propmap+json". The value of "Content-ID" MUST have the same format as the Part Content ID as specified in Section 6.6.
  The body of the Unified Property Map part MUST be a JSON object with the same format as defined in Section 4.6 of [I-D.ietf-alto-unified-props-new]. The JSON object MUST include the "dependent-vtags" field in the "meta" field. The value of the "dependent-vtags" field MUST be an array of VersionTag objects as defined by Section 10.3 of [RFC7285]. The "vtag" of the Path Vector part MUST be included in the "dependent-vtags". If "persistent-entity-id" is requested, the version tags of the dependent resources that MAY expose the entities in the response MUST also be included.

The PropertyMapData has one member for each ANEName that appears in the Path Vector part, which is an entity identifier belonging to the self-defined entity domain as defined in Section 5.1.2.3 of [I-D.ietf-alto-unified-props-new]. The EntityProps for each ANE has one member for each property that is both 1) associated with the ANE, and 2) specified in the "ane-property-names" in the request. If the Path Vector cost type is not included in the "cost-type" field or the "multi-cost-type" field, the "property-map" field MUST be present and the value MUST be an empty object ({}).

A complete and valid response MUST include both the Path Vector part and the Property Map part in the multipart message. If any part is ~~NOT~~ not present, the client MUST discard the received information and send another request if necessary.

According to [RFC2387], the Path Vector part, whose media type is the same as the "type" parameter of the multipart response message, is the root object. Thus, it is the element the application processes first. Even though the "start" parameter allows it to be placed anywhere in the part sequence, it is RECOMMENDED that the parts arrive in the same order as they are processed, i.e., the Path Vector part is always put as the first part, followed by the Property Map part. When doing so, an ALTO server MAY choose not to set the "start" parameter, which implies the first part is the root object.

Example: Consider the network in Figure 1. The response of the example request in Section 7.3.3 is as follows.

```
   HTTP/1.1 200 OK
   Content-Length: 810
   Content-Type: multipart/related; boundary=example-1;
                 type=application/alto-endpointcost+json
   --example-1
   Content-ID: <ecs@alto.example.com>
   Content-Type: application/alto-endpointcost+json
   {
     "meta": {
       "vtag": {
         "resource-id": "ecs-pv.ecs",
         "tag": "d827f484cb66ce6df6b5077cb8562b0a"
       },
       "dependent-vtags": [
         {
           "resource-id": "my-default-networkmap",
           "tag": "75ed013b3cb58f896e839582504f6228"
         }
       ],
       "cost-type": { "cost-mode": "array", "cost-metric": "ane-path" }
     },
     "cost-map": {
       "ipv4:192.0.2.2": { "ipv4:192.0.2.18": ["ANE1"] }
     }
   }
   --example-1
   Content-ID: <propmap@alto.example.com>
   Content-Type: application/alto-propmap+json
   {
     "meta": {
       "dependent-vtags": [
         {
           "resource-id": "ecs-pv.ecs",
           "tag": "d827f484cb66ce6df6b5077cb8562b0a"
         }
       ]
     },
     "property-map": {
       ".ane:ANE1": { "max-reservable-bandwidth": 100000000 }
     }
   }
```

Commenté [BMI19]: Folding ?

8. Examples

This section lists some examples of Path Vector queries and the corresponding responses. Some long lines are truncated for better readability.

8.1. ~~Example:~~Sample Setup

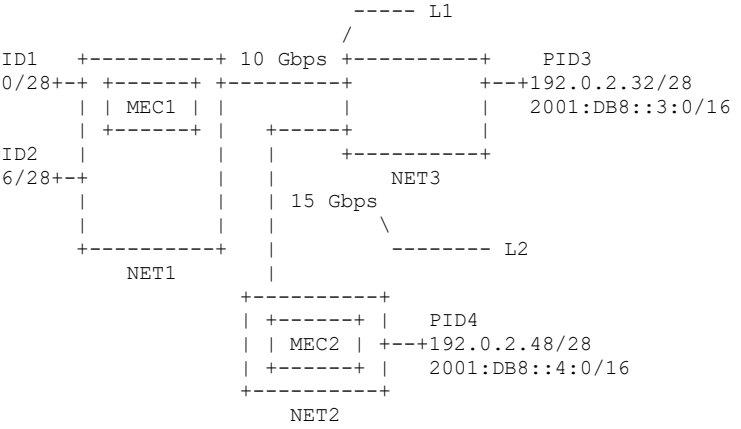


Figure 10: Examples of ANE Properties

In this document, Figure 10 is used to illustrate the message contents. There are 3 sub-networks (NET1, NET2 and NET3) and two interconnection links (L1 and L2). It is assumed that each sub-network has sufficiently large bandwidth to be reserved.

8.2. ~~Example:~~ Information Resource Directory

To give a comprehensive example of the extension defined in this document, we consider the network in Figure 10. Assume that the ALTO server provides the following information resources:

* "my-default-networkmap": A Network Map resource which contains the PIDs in the network.
* "filtered-cost-map-pv": A Multipart Filtered Cost Map resource for Path Vector, which exposes the "max-reservable-bandwidth" property for the PIDs in "my-default-networkmap".

* "ane-props": A filtered Unified Property resource that exposes the information for persistent ANEs in the network.
* "endpoint-cost-pv": A Multipart Endpoint Cost Service for Path Vector, which exposes the "max-reservable-bandwidth" and the "persistent-entity-id" properties.
* "update-pv": An Update Stream service, which provides the incremental update service for the "endpoint-cost-pv" service.
* "multicost-pv": A Multipart Endpoint Cost Service with both Multi-Cost and Path Vector.

Below is the Information Resource Directory of the example ALTO server. To enable the extension defined in this document, the "path-vector" cost type (Section 6.5) is defined in the "cost-types" of the "meta" field, and is included in the "cost-type-names" of resources "filtered-cost-map-pv" and "endpoint-cost-pv".

```
{
  "meta": {
    "cost-types": {
      "path-vector": {
        "cost-mode": "array",
        "cost-metric": "ane-path"
      },
      "num-rc": {
        "cost-mode": "numerical",
        "cost-metric": "routingcost"
      }
    }
  },
  "resources": {
    "my-default-networkmap": {
      "uri": "https://alto.example.com/networkmap",
      "uri" : "https://alto.example.com/networkmap",
      "media-type": "application/alto-networkmap+json"
      "media-type" : "application/alto-networkmap+json"
    },
    "filtered-cost-map-pv": {
      "uri": "https://alto.example.com/costmap/pv",
      "media-type": "multipart/related;
                     type=application/alto-costmap+json",
      "accepts": "application/alto-costmapfilter+json",
      "capabilities": {
        "cost-type-names": [ "path-vector" ],
        "ane-property-names": [ "max-reservable-bandwidth" ]
      },
      "uses": [ "my-default-networkmap" ]
    },
```

```
      "ane-props": {
        "uri": "https://alto.example.com/ane-props",
        "media-type": "application/alto-propmap+json",
        "accepts": "application/alto-propmapparams+json",
        "capabilities": {
          "mappings": {
            ".ane": [ "cpu" ]
          }
        }
      },
      "endpoint-cost-pv": {
        "uri": "https://alto.exmaple.com/endpointcost/pv",
        "media-type": "multipart/related;
                       type=application/alto-endpointcost+json",
        "accepts": "application/alto-endpointcostparams+json",
        "capabilities": {
          "cost-type-names": [ "path-vector" ],
          "ane-property-names": [
            "max-reservable-bandwidth", "persistent-entity-id"
          ]
        },
        "uses": [ "ane-props" ]
      },
      "update-pv": {
        "uri": "https://alto.example.com/updates/pv",
        "media-type": "text/event-stream",
        "uses": [ "endpoint-cost-pv" ],
        "accepts": "application/alto-updatestreamparams+json",
        "capabilities": {
          "support-stream-control": true
        }
      },
      "multicost-pv": {
        "uri": "https://alto.exmaple.com/endpointcost/mcpv",
        "media-type": "multipart/related;
                       type=application/alto-endpointcost+json",
        "accepts": "application/alto-endpointcostparams+json",
        "capabilities": {
          "cost-type-names": [ "path-vector", "num-rc" ],
          "max-cost-types": 2,
          "testable-cost-type-names": [ "num-rc" ],
          "ane-property-names": [
            "max-reservable-bandwidth", "persistent-entity-id"
          ]
        },
        "uses": [ "ane-props" ]
      }
    }
```

```
   }
```

8.3. ~~Example:~~ Multipart Filtered Cost Map

The following examples demonstrate the request to the "filtered-cost-map-pv" resource and the corresponding response.

The request uses the "path-vector" cost type in the "cost-type" field. The "ane-property-names" field is missing, indicating that the client only requests for the Path Vector but not the ANE properties.

The response consists of two parts. The first part returns the array of ANEName for each source and destination pair. There are two ANEs, where "L1" represents the interconnection link L1, and "L2" represents the interconnection link L2.

The second part returns an empty Property Map. Note that the ANE entries are omitted since they have no properties (See Section 3.1 of [I-D.ietf-alto-unified-props-new]).

```
POST /costmap/pv HTTP/1.1
Host: alto.example.com
Accept: multipart/related;type=application/alto-costmap+json,
        application/alto-error+json
Content-Length: 153
Content-Type: application/alto-costmapfilter+json
{
  "cost-type": {
    "cost-mode": "array",
    "cost-metric": "ane-path"
  },
  "pids": {
    "srcs": [ "PID1" ],
    "dsts": [ "PID3", "PID4" ]
  }
}
HTTP/1.1 200 OK
Content-Length: 860
Content-Type: multipart/related; boundary=example-1;
              type=application/alto-costmap+json
--example-1
Content-ID: <costmap@alto.example.com>
Content-Type: application/alto-costmap+json
```

```
   {
     "meta": {
       "vtag": {
         "resource-id": "filtered-cost-map-pv.costmap",
         "tag": "d827f484cb66ce6df6b5077cb8562b0a"
       },
       "dependent-vtags": [
         {
           "resource-id": "my-default-networkmap",
           "tag": "75ed013b3cb58f896e839582504f6228"
         }
       ],
       "cost-type": {
         "cost-mode": "array",
         "cost-metric": "ane-path"
       }
     },
     "cost-map": {
       "PID1": {
         "PID3": [ "L1" ],
         "PID4": [ "L1", "L2" ]
       }
     }
   }
   --example-1
   Content-ID: <propmap@alto.example.com>
   Content-Type: application/alto-propmap+json

   {
     "meta": {
       "dependent-vtags": [
         {
           "resource-id": "filtered-cost-map-pv.costmap",
           "tag": "d827f484cb66ce6df6b5077cb8562b0a"
         }
       ]
     },
     "property-map": {
     }
   }
```

8.4. ~~Example:~~ Multipart Endpoint Cost Service Resource

The following examples demonstrate the request to the "endpoint-cost-pv" resource and the corresponding response.

The request uses the Path Vector cost type in the "cost-type" field, and queries the Maximum Reservable Bandwidth ANE property and the Persistent Entity property for two IPv4 source and destination pairs (192.0.2.34 -> 192.0.2.2 and 192.0.2.34 -> 192.0.2.50) and one IPv6 source and destination pair (2001:DB8::3:1 -> 2001:DB8::4:1).

The response consists of two parts. The first part returns the array of ANEName for each valid source and destination pair. As one can see in Figure 10, flow 192.0.2.34 -> 192.0.2.2 traverses NET2, L1 and NET1, and flows 192.0.2.34 -> 192.0.2.50 and 2001:DB8::3:1 -> 2001:DB8::4:1 traverse NET2, L2 and NET3.

The second part returns the requested properties of ANEs. Assume NET1, NET2 and NET3 has sufficient bandwidth and their "max-reservable-bandwidth" values are set to a sufficiently large number (50 Gbps in this case). On the other hand, assume there are no prior reservation on L1 and L2, and their "max-reservable-bandwidth" values are the corresponding link capacity (10 Gbps for L1 and 15 Gbps for L2).

Both NET1 and NET2 have a mobile edge deployed, i.e., MEC1 in NET1 and MEC2 in NET2. Assume the ANEName for MEC1 and MEC2 are "MEC1" and "MEC2" and their properties can be retrieved from the Property Map "ane-props". Thus, the "persistent-entity-id" property of NET1 and NET3 are "ane-props.ane:MEC1" and "ane-props.ane:MEC2" respectively.

```
   POST /endpointcost/pv HTTP/1.1
   Host: alto.example.com
   Accept: multipart/related;
           type=application/alto-endpointcost+json,
           application/alto-error+json
   Content-Length: 362
   Content-Type: application/alto-endpointcostparams+json

   {
     "cost-type": {
       "cost-mode": "array",
       "cost-metric": "ane-path"
     },
     "endpoints": {
       "srcs": [
         "ipv4:192.0.2.34",
         "ipv6:2001:DB8::3:1"
       ],
       "dsts": [
         "ipv4:192.0.2.2",
         "ipv4:192.0.2.50",
         "ipv6:2001:DB8::4:1"
       ]
     },
     "ane-property-names": [
       "max-reservable-bandwidth",
       "persistent-entity-id"
     ]
   }
   HTTP/1.1 200 OK
   Content-Length: 1433
   Content-Type: multipart/related; boundary=example-2;
                 type=application/alto-endpointcost+json
   --example-2
   Content-ID: <ecs@alto.example.com>
   Content-Type: application/alto-endpointcost+json
   {
     "meta": {
       "vtags": {
         "resource-id": "endpoint-cost-pv.ecs",
         "tag": "bb6bb72eafe8f9bdc4f335c7ed3b10822a391cef"
       },
       "cost-type": {
         "cost-mode": "array",
         "cost-metric": "ane-path"
```

```
      }
    },
    "endpoint-cost-map": {
      "ipv4:192.0.2.34": {
        "ipv4:192.0.2.2":   [ "NET3", "L1", "NET1" ],
        "ipv4:192.0.2.50":   [ "NET3", "L2", "NET2" ]
      },
      "ipv6:2001:DB8::3:1": {
        "ipv6:2001:DB8::4:1": [ "NET3", "L2", "NET2" ]
      }
    }
  }
  --example-2
  Content-ID: <propmap@alto.example.com>
  Content-Type: application/alto-propmap+json
  {
    "meta": {
      "dependent-vtags": [
        {
          "resource-id": "endpoint-cost-pv.ecs",
          "tag": "bb6bb72eafe8f9bdc4f335c7ed3b10822a391cef"
        },
        {
          "resource-id": "ane-props",
          "tag": "bf3c8c1819d2421c9a95a9d02af557a3"
        }
      ]
    },
    "property-map": {
      ".ane:NET1": {
        "max-reservable-bandwidth": 50000000000,
        "persistent-entity-id": "ane-props.ane:MEC1"
      },
      ".ane:NET2": {
        "max-reservable-bandwidth": 50000000000,
        "persistent-entity-id": "ane-props.ane:MEC2"
      },
      ".ane:NET3": {
        "max-reservable-bandwidth": 50000000000
      },
      ".ane:L1": {
        "max-reservable-bandwidth": 10000000000
      },
      ".ane:L2": {
        "max-reservable-bandwidth": 15000000000
      }
    }
```

```
   }
```

As mentioned in Section 6.5.1, an advanced ALTO server may obfuscate the response in order to preserve its own privacy or conform to its own policies. For example, an ALTO server may choose to aggregate NET1 and L1 as a new ANE with ANE name "AGGR1", and aggregate NET2 and L2 as a new ANE with ANE name "AGGR2". The "max-reservable-bandwidth" of "AGGR1" takes the value of L1, which is smaller than that of NET1, and the "persistent-entity-id" of "AGGR1" takes the value of NET1. The properties of "AGGR2" are computed in a similar way and the obfuscated response is as shown below. Note that the obfuscation of Path Vector responses is implementation-specific and is out of the scope of this document, and developers may refer to Section 11 for further references.

```
   HTTP/1.1 200 OK
   Content-Length: 1280
   Content-Type: multipart/related; boundary=example-2;
                 type=application/alto-endpointcost+json
   --example-2
   Content-ID: <ecs@alto.example.com>
   Content-Type: application/alto-endpointcost+json
   {
     "meta": {
       "vtags": {
         "resource-id": "endpoint-cost-pv.ecs",
         "tag": "bb6bb72eafe8f9bdc4f335c7ed3b10822a391cef"
       },
       "cost-type": {
         "cost-mode": "array",
         "cost-metric": "ane-path"
       }
     },
     "endpoint-cost-map": {
       "ipv4:192.0.2.34": {
         "ipv4:192.0.2.2":   [ "NET3", "AGGR1" ],
         "ipv4:192.0.2.50":   [ "NET3", "AGGR2" ]
       },
       "ipv6:2001:DB8::3:1": {
         "ipv6:2001:DB8::4:1": [ "NET3", "AGGR2" ]
       }
     }
   }
   --example-2
   Content-ID: <propmap@alto.example.com>
   Content-Type: application/alto-propmap+json
```

```
   {
     "meta": {
       "dependent-vtags": [
         {
           "resource-id": "endpoint-cost-pv.ecs",
           "tag": "bb6bb72eafe8f9bdc4f335c7ed3b10822a391cef"
         },
         {
           "resource-id": "ane-props",
           "tag": "bf3c8c1819d2421c9a95a9d02af557a3"
         }
       ]
     },
     "property-map": {
       ".ane:AGGR1": {
         "max-reservable-bandwidth": 10000000000,
         "persistent-entity-id": "ane-props.ane:MEC1"
       },
       ".ane:AGGR2": {
         "max-reservable-bandwidth": 15000000000,
         "persistent-entity-id": "ane-props.ane:MEC2"
       },
       ".ane:NET3": {
         "max-reservable-bandwidth": 50000000000
       }
     }
   }
```

8.5. ~~Example:~~ Incremental Updates

In this example, an ALTO client subscribes to the incremental update for the multipart Endpoint Cost Service resource "endpoint-cost-pv".

```
   POST /updates/pv HTTP/1.1
   Host: alto.example.com
   Accept: text/event-stream
   Content-Type: application/alto-updatestreamparams+json
   Content-Length: 112
   {
     "add": {
       "ecspvsub1": {
         "resource-id": "endpoint-cost-pv",
         "input": <ecs-input>
       }
     }
   }
```

Based on the server-side process defined in [RFC8895], the ALTO server will send the "control-uri" first using Server-Sent Event (SSE), followed by the full response of the multipart message.

```
HTTP/1.1 200 OK
Connection: keep-alive
Content-Type: text/event-stream
event: application/alto-updatestreamcontrol+json
data: {"control-uri": "https://alto.example.com/updates/streams/123"}
event: multipart/related;boundary=example-3;
       type=application/alto-endpointcost+json,ecspvsub1
data: --example-3
data: Content-ID: <ecsmap@alto.example.com>
data: Content-Type: application/alto-endpointcost+json
data:
data: <endpoint-cost-map-entry>
data: --example-3
data: Content-ID: <propmap@alto.example.com>
data: Content-Type: application/alto-propmap+json
data:
data: <property-map-entry>
data: --example-3--
```

When the contents change, the ALTO server will publish the updates for each node in this tree separately.

```
event: application/merge-patch+json, ecspvsub1.ecsmap
data: <Merge patch for endpoint-cost-map-update>
event: application/merge-patch+json, ecspvsub1.propmap
data: <Merge patch for property-map-update>
```

8.6. ~~Example:~~ Multi-cost

The following examples demonstrate the request to the "multicost-pv" resource and the corresponding response.

The request asks for two cost types: the first is the Path Vector cost type, and the second is a numerical routing cost. It also queries the Maximum Reservable Bandwidth ANE property and the Persistent Entity property for two IPv4 source and destination pairs (192.0.2.34 -> 192.0.2.2 and 192.0.2.34 -> 192.0.2.50) and one IPv6 source and destination pair (2001:DB8::3:1 -> 2001:DB8::4:1).

The response consists of two parts. The first part returns a JSONArray that contains two JSONValue for each requested source and destination pair: the first JSONValue is a JSONArray of ANENames, which is the value of the Path Vector cost type, and the second JSONValue is a JSONNumber which is the value of the routing cost. The second part is the same as in Section 8.4

```
POST /endpointcost/mcpv HTTP/1.1
Host: alto.example.com
Accept: multipart/related;
        type=application/alto-endpointcost+json,
        application/alto-error+json
Content-Length: 433
Content-Type: application/alto-endpointcostparams+json
{
  "multi-cost-types": [
    { "cost-mode": "array", "cost-metric": "ane-path" },
    { "cost-mode": "numerical", "cost-metric": "routingcost" }
  ],
  "endpoints": {
    "srcs": [
      "ipv4:192.0.2.34",
      "ipv6:2001:DB8::3:1"
    ],
    "dsts": [
      "ipv4:192.0.2.2",
      "ipv4:192.0.2.50",
      "ipv6:2001:DB8::4:1"
    ]
  },
  "ane-property-names": [
    "max-reservable-bandwidth",
    "persistent-entity-id"
  ]
}
HTTP/1.1 200 OK
Content-Length: 1366
Content-Type: multipart/related; boundary=example-4;
              type=application/alto-endpointcost+json
--example-4
Content-ID: <ecs@alto.example.com>
Content-Type: application/alto-endpointcost+json
{
  "meta": {
```

```
       "vtags": {
         "resource-id": "endpoint-cost-pv.ecs",
         "tag": "bb6bb72eafe8f9bdc4f335c7ed3b10822a391cef"
       },
       "multi-cost-types": [
         { "cost-mode": "array", "cost-metric": "ane-path" },
         { "cost-mode": "numerical", "cost-metric": "routingcost" }
       ]
     },
     "endpoint-cost-map": {
       "ipv4:192.0.2.34": {
         "ipv4:192.0.2.2":   [[ "NET3", "AGGR1" ], 1],
         "ipv4:192.0.2.50":   [[ "NET3", "AGGR2" ], 1]
       },
       "ipv6:2001:DB8::3:1": {
         "ipv6:2001:DB8::4:1": [[ "NET3", "AGGR2" ], 1]
       }
     }
   }
   --example-4
   Content-ID: <propmap@alto.example.com>
   Content-Type: application/alto-propmap+json
   {
     "meta": {
       "dependent-vtags": [
         {
           "resource-id": "endpoint-cost-pv.ecs",
           "tag": "bb6bb72eafe8f9bdc4f335c7ed3b10822a391cef"
         },
         {
           "resource-id": "ane-props",
           "tag": "bf3c8c1819d2421c9a95a9d02af557a3"
         }
       ]
     },
     "property-map": {
       ".ane:AGGR1": {
         "max-reservable-bandwidth": 10000000000,
         "persistent-entity-id": "ane-props.ane:MEC1"
       },
       ".ane:AGGR2": {
         "max-reservable-bandwidth": 15000000000,
         "persistent-entity-id": "ane-props.ane:MEC2"
       },
       ".ane:NET3": {
         "max-reservable-bandwidth": 50000000000
       }
```

```
    }
  }
```

9. Compatibility with Other ALTO Extensions

9.1. Compatibility with Legacy ALTO Clients/Servers

The multipart Filtered Cost Map resource and the multipart Endpoint Cost Service resource has no backward compatibility issue with legacy ALTO clients and servers. Although these two types of resources reuse the media types defined in the base ALTO protocol for the accept input parameters, they have different media types for responses. If the ALTO server provides these two types of resources, but the ALTO client does not support them, the ALTO client will ignore the resources without incurring any incompatibility problem.

9.2. Compatibility with Multi-Cost Extension

The extension defined in this document is compatible with the multi-cost extension [RFC8189]. Such a resource has a media type of either "multipart/related; type=application/alto-costmap+json" or "multipart/related; type=application/alto-endpointcost+json". Its "cost-constraints" field must either be "false" or not present and the Path Vector cost type must be present in the "cost-type-names" capability field but must not be present in the "testable-cost-type-names" field, as specified in Section 7.2.4 and Section 7.3.4.

9.3. Compatibility with Incremental Update

ALTO clients and servers MUST follow the specifications given in Section 5.2 of [RFC8895] to support incremental updates for a Path Vector resource.

9.4. Compatibility with Cost Calendar

The extension specified in this document is compatible with the Cost Calendar extension [RFC8896]. When used together with the Cost Calendar extension, the cost value between a source and a destination is an array of Path Vectors, where the k-th Path Vector refers to the abstract network paths traversed in the k-th time interval by traffic from the source to the destination.

When used with time-varying properties, e.g., maximum reservable bandwidth ~~(maxresbw)~~, a property of a single ANE may also have different values in different time intervals. In this case, if such an ANE has different property values in two-time ~~two time~~ intervals, it MUST be treated as two different ANEs, i.e., with different entity identifiers. However, if it has the same property values in two time intervals, it MAY use the same identifier.

This rule allows the Path Vector extension to represent both changes of ANEs and changes of the ANEs' properties in a uniform way. The Path Vector part is calendared in a compatible way, and the Property Map part is not affected by the calendar extension.

The two extensions combined together can provide the historical network correlation information for a set of source and destination pairs. A network broker or client may use this information to derive other resource requirements such as Time-Block-Maximum Bandwidth, Bandwidth-Sliding-Window, and Time-Bandwidth-Product (TBP) (See [SENSE] for details).

10. General Discussions

10.1. Constraint Tests for General Cost Types

The constraint test is a simple approach to query the data. It allows users to filter the query result by specifying some boolean tests. This approach is already used in the ALTO protocol. [RFC7285] and [RFC8189] allow ALTO clients to specify the "constraints" and "or-constraints" tests to better filter the result. However, the current syntax can only be used to test scalar cost types, and cannot easily express constraints on complex cost types, e.g., the Path Vector cost type defined in this document.

In practice, developing a bespoke language for general-purpose boolean tests can be a complex undertaking, and it is conceivable that there are some existing implementations already (the authors have not done an exhaustive search to determine whether there are such implementations). One avenue to develop such a language may be to explore extending current query languages like XQuery [XQuery] or JSONiq [JSONiq] and integrating these with ALTO.

Filtering the Path Vector results or developing a more sophisticated filtering mechanism is beyond the scope of this document.

10.2. General Multi-Resource Query

Querying multiple ALTO information resources continuously is a general requirement. Enabling such a capability, however, must address general issues like efficiency and consistency. The incremental update extension [RFC8895] supports submitting multiple queries in a single request, and allows flexible control over the queries. However, it does not cover the case introduced in this document where multiple resources are needed for a single request. This extension gives an example of using a multipart message to encode the responses from two specific ALTO information resources: a Filtered Cost Map or an Endpoint Cost Service, and a Property Map. By packing multiple resources in a single response, the implication is that servers may proactively push related information resources to clients.

Thus, it is worth looking into the direction of extending the SSE mechanism as used in the incremental update extension [RFC8895], or upgrading to HTTP/2 [RFC7540] and HTTP/3 [I-D.ietf-quic-http], which provides the ability to multiplex queries and to allow servers proactively send related information resources.

Defining a general multi-resource query mechanism is out of the scope of this document.

11. Security Considerations

This document is an extension of the base ALTO protocol, so the Security Considerations [RFC7285] of the base ALTO protocol fully apply when this extension is provided by an ALTO server.

The Path Vector extension requires additional scrutiny on three security considerations discussed in the base protocol: confidentiality of ALTO information (Section 15.3 of [RFC7285]), potential undesirable guidance from authenticated ALTO information (Section 15.2 of [RFC7285]), and availability of ALTO service (Section 15.5 of [RFC7285]).

For confidentiality of ALTO information, a network operator should be aware of that this extension may introduce a new risk: the Path Vector information may make network attacks easier. For example, as the Path Vector information may reveal more fine-grained internal network structures than the base protocol, an ALTO client may detect the bottleneck link and start a distributed denial-of-service (DDoS) attack involving minimal flows to conduct the in-network congestion.

To mitigate this risk, the ALTO server should consider protection mechanisms to reduce information exposure or obfuscate the real information, in particular, in settings where the network and the application do not belong to the same trust domain. For example, in the multi-flow bandwidth reservation use case as introduced in Section 4, only the available bandwidth of the shared bottleneck link is crucial, and the ALTO server may only preserve the critical bottlenecks and can change the order of links appearing in the Path Vector response.

However, arbitrary reduction and obfuscation of information exposure may potentially introduce a risk on the integrity of the ALTO information, leading to infeasible or suboptimal decisions of ALTO clients,

To mitigate this risk, if an ALTO client finds that the traffic distribution based on the Path Vector information is not feasible (e.g., causing constant congestion) or not better than a distribution which does not fully conform to the information (e.g., by randomly choosing the source/destination for certain flows), it can follow the protection strategies for potential undesirable guidance from authenticated ALTO information, specified in Section 15.2.2 of RFC 7285 [RFC7285]. While repeatedly sending the same query can potentially detect the integrity problem for certain obfuscation methods (e.g., those based on time or randomness) under certain network conditions (e.g., where the routing and ANE properties are stable), an ALTO client must be aware that this behavior may be considered as a denial-of-service attack on the server and may lead to the rejection of further requests from the client.

On the other hand, this risk can also be mitigated from the server side. While the implementation of an ALTO server is beyond the scope of this document, implementations of ALTO servers involving reduction or obfuscation of the Path Vector information should consider reduction/obfuscation mechanisms that can preserve the integrity of ALTO information, for example, by using minimal feasible region compression algorithms [NOVA] or obfuscation protocols [RESA][MERCATOR].

For availability of ALTO service, an ALTO server should be cognizant that using Path Vector extension might have a new risk: frequent requesting for Path Vectors might consume intolerable amounts of the server-side computation and storage, which can break the ALTO server. For example, if an ALTO server implementation dynamically computes the Path Vectors for each request, the service providing Path Vectors may become an entry point for denial-of-service attacks on the availability of an ALTO server.

To mitigate this risk, an ALTO server may consider using optimizations such as precomputation-and-projection mechanisms [MERCATOR] to reduce the overhead for processing each query. Also, an ALTO server may also protect itself from malicious clients by monitoring the behaviors of clients and stopping serving clients with suspicious behaviors (e.g., sending requests at a high frequency).

12. IANA Considerations

12.1. ALTO Entity Domain Type Registry

This document registers a new entry to the ALTO Domain Entity Type Registry, as instructed by Section 12.2 of [I-D.ietf-alto-unified-props-new]. The new entry is as shown below in Table 1.

Identifier	Entity Address Encoding	Hierarchy & Inheritance
ane	See Section 6.2.2	None

Table 1: ALTO Entity Domain Type Registry

Identifier: See Section 6.2.1.

Entity Identifier Encoding: See Section 6.2.2.

Hierarchy: None

Inheritance: None

Media Type of Defining Resource: See Section 6.2.4.

Security Considerations: In some usage scenarios, ANE addresses carried in ALTO Protocol messages may reveal information about an ALTO client or an ALTO service provider. Applications and ALTO service providers using addresses of ANEs will be made aware of how (or if) the addressing scheme relates to private information and network proximity, in further iterations of this document.

12.2. ALTO Entity Property Type Registry

Two initial entries "max-reservable-bandwidth" and "persistent-entity-id" are registered to the ALTO Domain "ane" in the "ALTO Entity Property Type Registry", as instructed by Section 12.3 of [I-D.ietf-alto-unified-props-new]. The two new entries are shown below in Table 2 and their details can be found in Section 12.2.1 and Section 12.2.2.

Identifier	Intended Semantics	Media Type of Defining Resource
max-reservable-bandwidth	See Section 6.4.1	application/alto-propmap+json
persistent-entity-id	See Section 6.4.2	application/alto-propmap+json

Table 2: Initial Entries for ane Domain in the ALTO Entity Property Types Registry

12.2.1. New ANE Property Type: Maximum Reservable Bandwidth

Identifier: "max-reservable-bandwidth"

Intended Semantics: See Section 6.4.1.

Media Type of Defining Resource: application/alto-propmap+json

Security Considerations: This property is essential for applications such as large-scale data transfers or overlay network interconnection to make better choice of bandwidth reservation. It may reveal the bandwidth usage of the underlying network and can potentially be leveraged to reduce the cost of conducting denial-of-service attacks. Thus, the ALTO server MUST consider protection mechanisms including only providing the information to authorized clients, and information reduction and obfuscation as introduced in Section 11.

12.2.2. New ANE Property Type: Persistent Entity ID

Identifier: "persistent-entity-id"

Intended Semantics: See Section 6.4.2.

Media Type of Defining Resource: application/alto-propmap+json

Security Considerations: This property is useful when an ALTO server wants to selectively expose certain service points whose detailed properties can be further queried by applications. The entity IDs may consider sensitive information about the underlying network, and an ALTO server should follow the security considerations in Section 11 of [I-D.ietf-alto-unified-props-new].

13. Acknowledgments

The authors would like to thank discussions with Andreas Voellmy, Erran Li, Haibin Song, Haizhou Du, Jiayuan Hu, Qiao Xiang, Tianyuan Liu, Xiao Shi, Xin Wang, and Yan Luo. The authors thank Greg Bernstein, Dawn Chen, Wendy Roome, and Michael Scharf for their contributions to earlier drafts.

14. References

14.1. Normative References

[I-D.ietf-alto-unified-props-new] Roome, W., Randriamasy, S., Yang, Y. R., Zhang, J. J., and K. Gao, "ALTO Extension: Entity Property Maps", Work in Progress, Internet-Draft, draft-ietf-alto-unified-props-new-19, 25 October 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-alto-unified-props-new-19>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC2387] Levinson, E., "The MIME Multipart/Related Content-type", RFC 2387, DOI 10.17487/RFC2387, August 1998, <https://www.rfc-editor.org/rfc/rfc2387>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/rfc/rfc5322>.

[RFC7285] Alimi, R., Ed., Penno, R., Ed., Yang, Y., Ed., Kiesel, S., Previdi, S., Roome, W., Shalunov, S., and R. Woundy, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, DOI 10.17487/RFC7285, September 2014, <https://www.rfc-editor.org/rfc/rfc7285>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8189] Randriamasy, S., Roome, W., and N. Schwan, "Multi-Cost Application-Layer Traffic Optimization (ALTO)", RFC 8189, DOI 10.17487/RFC8189, October 2017, <https://www.rfc-editor.org/rfc/rfc8189>.

[RFC8895] Roome, W. and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Incremental Updates Using Server-Sent Events (SSE)", RFC 8895, DOI 10.17487/RFC8895, November 2020, <https://www.rfc-editor.org/rfc/rfc8895>.

[RFC8896] Randriamasy, S., Yang, R., Wu, Q., Deng, L., and N. Schwan, "Application-Layer Traffic Optimization (ALTO) Cost Calendar", RFC 8896, DOI 10.17487/RFC8896, November 2020, <https://www.rfc-editor.org/rfc/rfc8896>.

14.2. Informative References

[BOXOPT] Xiang, Q., Yu, H., Aspnes, J., Le, F., Kong, L., and Y.R. Yang, "Optimizing in the dark: Learning an optimal solution through a simple request interface", Proceedings of the AAAI Conference on Artificial Intelligence 33, 1674-1681, 2019.

[CLARINET] Viswanathan, R., Ananthanarayanan, G., and A. Akella, "CLARINET: WAN-Aware Optimization for Analytics Queries", In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), USENIX Association, Savannah, GA, 435-450, 2016.

[G2] Ros-Giralt, J., Bohara, A., Yellamraju, S., Langston, M.H., Lethin, R., Jiang, Y., Tassiulas, L., Li, J., Tan, Y., and M. Veeraraghavan, "On the Bottleneck Structure of Congestion-Controlled Networks", Proceedings of the ACM on Measurement and Analysis of Computing Systems, Volume 3, Issue 3, pp 1-31, 2019.

[HUG] Chowdhury, M., Liu, Z., Ghodsi, A., and I. Stoica, "HUG: Multi-Resource Fairness for Correlated and Elastic Demands", 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16) (Santa Clara, CA, 2016), 407-424, 2016.

[I-D.ietf-alto-performance-metrics]
Wu, Q., Yang, Y. R., Lee, Y., Dhody, D., Randriamasy, S., and L. M. C. Murillo, "ALTO Performance Cost Metrics", Work in Progress, Internet-Draft, draft-ietf-alto-performance-metrics-19, 23 October 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-alto-performance-metrics-19>.

[I-D.ietf-quic-http]
Bishop, M., "Hypertext Transfer Protocol Version 3 (HTTP/3)", Work in Progress, Internet-Draft, draft-ietf-quic-http-34, 2 February 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-quic-http-34>.

[JSONiq] "The JSON Query language", 2020, <https://www.jsoniq.org/>.

[MERCATOR] Xiang, Q., Zhang, J., Wang, X., Liu, Y., Guok, C., Le, F., MacAuley, J., Newman, H., and Y.R. Yang, "Toward Fine-Grained, Privacy-Preserving, Efficient Multi-Domain Network Resource Discovery", IEEE/ACM IEEE Journal on Selected Areas of Communication 37(8): 1924-1940, 2019.

[MOWIE] Zhang, Y., Li, G., Xiong, C., Lei, Y., Huang, W., Han, Y., Walid, A., Yang, Y.R., and Z. Zhang, "MoWIE: Toward Systematic, Adaptive Network Information Exposure as an Enabling Technique for Cloud-Based Applications over 5G and Beyond", In Proceedings of the Workshop on Network Application Integration/CoDesign, ACM, Virtual Event USA, 20-27, 2020.

[NOVA] Gao, K., Xiang, Q., Wang, X., Yang, Y.R., and J. Bi, "An objective-driven on-demand network abstraction for adaptive applications", IEEE/ACM Transactions on Networking (TON) Vol 27, no. 2 (2019): 805-818, 2019.

[RESA] Xiang, Q., Zhang, J., Wang, X., Liu, Y., Guok, C., Le, F., MacAuley, J., Newman, H., and Y.R. Yang, "Fine-grained, multi-domain network resource abstraction as a fundamental primitive to enable high-performance, collaborative data sciences", Proceedings of the Super Computing 2018, 5:1-5:13, 2019.

[RFC2216] Shenker, S. and J. Wroclawski, "Network Element Service Specification Template", RFC 2216, DOI 10.17487/RFC2216, September 1997, <https://www.rfc-editor.org/rfc/rfc2216>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <https://www.rfc-editor.org/rfc/rfc7540>.

[SENSE] "Services - SENSE", 2019, <http://sense.es.net/services>.

[SEREDGE] Contreras, L., Baliosian, J., Martínez-Julia, P., and J. Serrat, "Computing at the Edge: But, what Edge?", In proceedings of the NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium. pp. 1-9., 2020.

[SWAN] Hong, C., Kandula, S., Mahajan, R., Zhang, M., Gill, V., Nanduri, M., and R. Wattenhofer, "Achieving High Utilization with Software-driven WAN", In Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM (SIGCOMM '13), ACM, New York, NY, USA, 15-26., 2013.

[UNICORN] Xiang, Q., Chen, S., Gao, K., Newman, H., Taylor, I., Zhang, J., and Y.R. Yang, "Unicorn: Unified Resource Orchestration for Multi-Domain, Geo-Distributed Data Analytics", 2017 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computed, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI) (Aug. 2017), 1-6., 2017.

[XQuery] "XQuery 3.1: An XML Query Language", 2017, <https://www.w3.org/TR/xquery-31/>.

Commenté [BMI20]: I can't access this link