ï»¿

Internet Engineering Task Force (IETF)
Request for Comments: 9275
Category: Experimental
ISSN: 2070-1721

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

September 2022

An Extension for Application-Layer Traffic Optimization (ALTO): Path Vector

Abstract

This document is an extension to the base Application-Layer Traffic Optimization (ALTO) protocol. It extends the ALTO cost map and ALTO property map services so that an application can decide to which endpoint(s) to connect based not only on numerical/ordinal cost values but also on fine-grained abstract information regarding the paths. This is useful for applications whose performance is impacted by specific 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 the "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 document is not an Internet Standards Track specification; it is published for examination, experimental implementation, and evaluation.

This document defines an Experimental Protocol for the Internet community. This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are candidates for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9275.

Table of Contents

1. Introduction

Network performance metrics are crucial for assessing the Quality of Experience (QoE) of applications. The Application-Layer Traffic Optimization (ALTO) protocol allows Internet Service Providers (ISPs) to provide guidance, such as topological distances 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.

The existing ALTO cost map (Section 11.2.3 of [RFC7285]) and Endpoint Cost Service (Section 11.5 of [RFC7285]) provide only cost information for 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 [ALTO-PERF-METRICS], to query multiple costs simultaneously [RFC8189], and to obtain time-varying values [RFC8896].

While numerical/ordinal cost values for end-to-end paths provided by the existing extensions are sufficient to optimize the QoE of many overlay applications, the QoE of some overlay applications also depends on the properties of particular components on the paths. 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 (ANEs).

Predicting such information can be very complex without the help of ISPs; for example, [BOXOPT] has shown that finding the optimal bandwidth reservation for multiple flows can be NP-hard without further information than whether a reservation succeeds. 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.

Despite the claimed benefits, ISPs are not likely to expose raw details on their network paths: first because ISPs have requirements to hide their network topologies, second because these details may increase volume and computation overhead, and last because applications do not necessarily need all the network path details and are likely not able to understand them.

Therefore, it is beneficial for both ISPs and applications if an ALTO server provides ALTO clients with an "abstract network state" that provides the necessary information to applications, while hiding network complexity and confidential information. An "abstract network state" is a selected set of abstract representations of ANEs traversed by the paths between <source, destination> pairs combined with properties of these ANEs that are relevant to the 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 ANEs. Server scalability can also be improved by combining ANEs and their properties in a single response.

This document extends the ALTO base protocol [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 a "Path Vector", following the cost metric registration specified in [RFC7285] and the updated cost mode registration specified in [RFC9274]. A Path Vector is an array of identifiers that identifies an ANE, which can be associated with various properties. The associations between ANEs and their properties are encoded in an ALTO information resource called the "entity property map", which is specified in [RFC9240].

For better confidentiality, this document aims to minimize information exposure of an ALTO server when providing Path Vector services. In particular, this document enables the capability, and also recommends that 1) ANEs be constructed on demand and 2) an ANE only be associated with properties that are requested by an ALTO client. A Path Vector response involves two ALTO maps: the cost map, which contains the Path Vector results; and the up-to-date entity property map, which contains the properties requested for these ANEs. To enforce consistency and improve server scalability, this document uses the "multipart/related" content type as defined in [RFC2387] to return the two maps in a single response.

As a single ISP may not have knowledge of the full Internet paths between arbitrary endpoints, this document is mainly applicable when

* there is a single ISP between the requested source and destination Provider-defined Identifiers (PIDs) or endpoints -- for example, ISP-hosted Content Delivery Network (CDN) / edge, tenant interconnection in a single public cloud platform, etc., or

* the Path Vectors are generated from end-to-end measurement data.

2. Requirements Language

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.

3. Terminology

This document extends the ALTO base protocol [RFC7285] and the entity property map extension [RFC9240]. In addition to the terms defined in those documents, this document also uses the following terms:

Abstract Network Element (ANE): 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 an ANE is similar to that for a network element as defined in [RFC2216] in the sense that they both provide an abstract representation of 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 an ANE only requires the components to have an impact on end-to-end performance.

ANE name: A string that uniquely identifies an ANE in a specific scope. 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 (or ANE Path Vector): Refers to a JSON array of ANE names. It is a generalization of a BGP path vector. While a

standard BGP path vector (Section 5.1.2 of [RFC4271]) specifies a sequence of Autonomous Systems (ASes) for a destination IP prefix, the Path Vector defined in this extension specifies a sequence of ANEs for either 1) a source PID and a destination PID, as in the CostMapData object (Section 11.2.3.6 of [RFC7285]) or 2) a source endpoint and a destination endpoint, as in the EndpointCostMapData object (Section 11.5.1.6 of [RFC7285]).

Path Vector resource: An ALTO information resource (Section 8.1 of [RFC7285]) that supports the extension defined in this document.

Path Vector cost type: A special cost type, which is specified in Section 6.5. When this cost type is present in an Information Resource Directory (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 that each cost value must be interpreted as a Path Vector.

Path Vector request: The POST message sent to an ALTO Path Vector resource.

Path Vector response: Refers to the multipart/related message returned by a Path Vector resource.

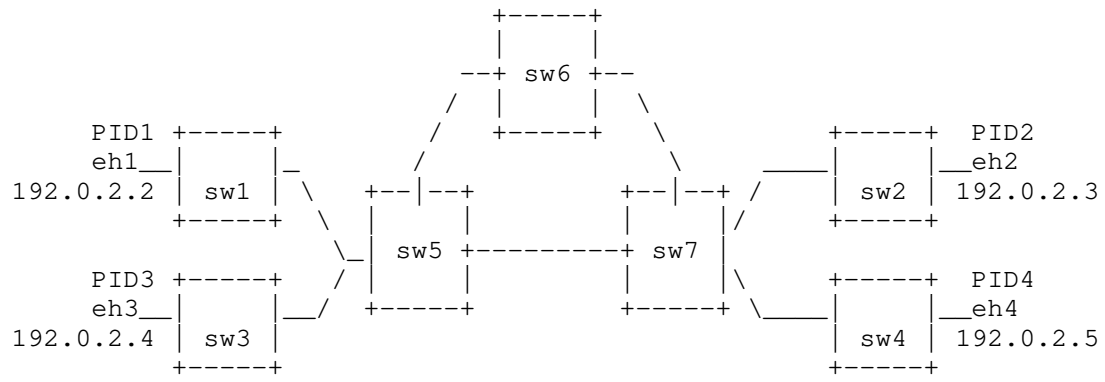
4. Requirements and Use Cases

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 and share bottlenecks. The application seeks to schedule the traffic among multiple flows to get better performance. The constraints of feasible rate allocations of those flows will benefit the scheduling. However, cost maps as defined in [RFC7285] cannot reveal such information.

Specifically, consider the example network shown in Figure 1. The network has seven switches ("sw1" to "sw7") forming a dumbbell topology. Switches "sw1", "sw2", "sw3", and "sw4" are access switches, 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 base ALTO topology abstraction of the network is shown in Figure 2. Assume that the cost map returns a hypothetical cost type representing the available bandwidth between a source and a

destination.

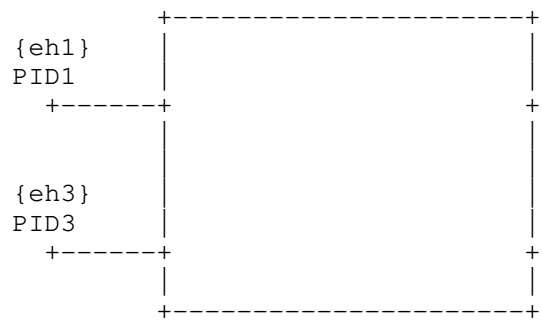


Figure 2: Base Topology Abstraction

Now, assume that 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 costs between PID1 and PID2 and between PID1 and PID4 will both be 100 Mbps. 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, 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 "eh1 -> eh2" uses path "eh1 -> sw1 -> sw5 -> sw7 -> sw2 -> eh2" and "eh1 -> eh4" also uses path "eh1 -> sw1 -> sw5 -> sw7 -> sw4 -> eh4", then the shared bottleneck link is "sw5 -> sw7". 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 better predict its traffic and may orchestrate its resources accordingly. However, to provide such information, the network needs to expose abstract information beyond the simple cost map abstraction. In particular:

* The ALTO server must expose abstract information 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 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 expose abstract information on the properties of the ANEs used by "eh1 -> eh2" and "eh1 -> eh4". For example, an ALTO server can either expose the available bandwidth between "eh1--sw1", "sw1--sw5", "sw5--sw7", "sw5--sw6", "sw6--sw7", "sw7--sw2", "sw7--sw4", "sw2--eh2", "sw4--eh4" in Case 1 or expose three abstract elements "A", "B", and "C", which represent the linear constraints that define the same capacity region in Case 1.

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

AR1: An ALTO server must provide the ANEs that are important for assessing the QoE of the overlay application on the path of a <source, destination> pair.

AR2: An ALTO server must provide information to identify how ANEs are shared on the paths of different <source, destination> pairs.

AR3: An ALTO server must provide information on the properties that are important for assessing the QoE of the application for ANEs.

The extension defined in this document specifies a solution to expose such abstract information.

4.2. Sample Use Cases

While the problem related to multiple flow scheduling 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 of the reported use cases.

4.2.1. Exposing Network Bottlenecks

One important use case for the Path Vector extension is to expose network bottlenecks. Applications that need to perform large-scale data transfers can benefit from being aware of the resource constraints exposed by this extension even if they have different objectives. One such example is the Worldwide LHC Computing Grid (WLCG) (where "LHC" means "Large Hadron Collider"), which is the largest example of a distributed computation collaboration in the research and education world.

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

In such a setting, the network-aware job optimizer (e.g., [CLARINET]) takes a query and generates multiple query execution plans (QEPs). It can encode the QEPs as Path Vector requests that are sent to an ALTO server. The ALTO server obtains the routing information for the flows in a QEP and finds links, routers, or middleboxes (e.g., a stateful firewall) that can potentially become bottlenecks for the QEP (e.g., see [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 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 vectors (e.g., [HUG], [UNICORN]), etc. The traffic control interfaces and mechanisms are out of scope for this document.

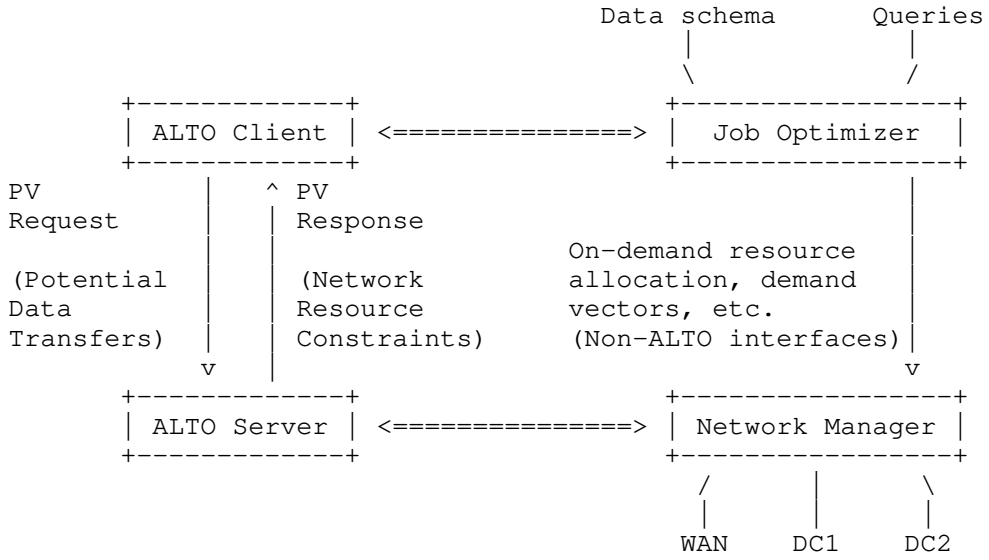


Figure 3: Example Use Case for Data Analytics

Another example is 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 a bottleneck for the data transfers.

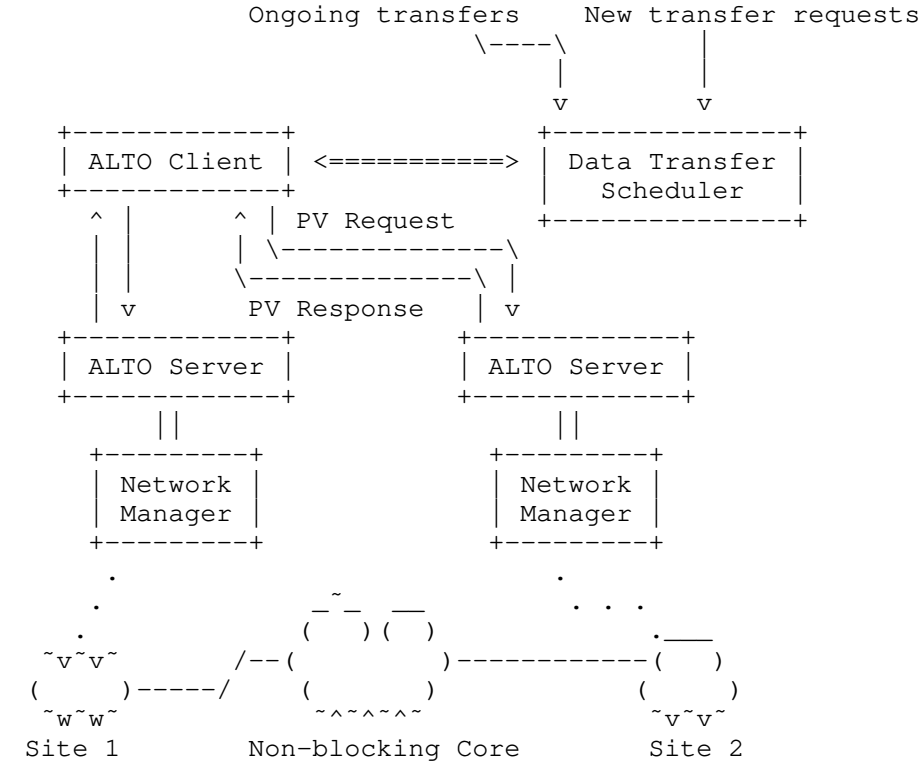


Figure 4: Example Use Case for Cross-Site Bottleneck Discovery

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, or no bottleneck information at all. The bottleneck information can be used to analyze the impact of adding/removing data transfer flows, e.g., using the framework defined in [G2]. For example, assume that hosts "a", "b", and "c" are in Site 1 and hosts "d", "e", and "f" are in Site 2, and there are three flows in two sites: "a -> b", "c -> d", and "e -> f" (Figure 5).

Site 1:

[c]

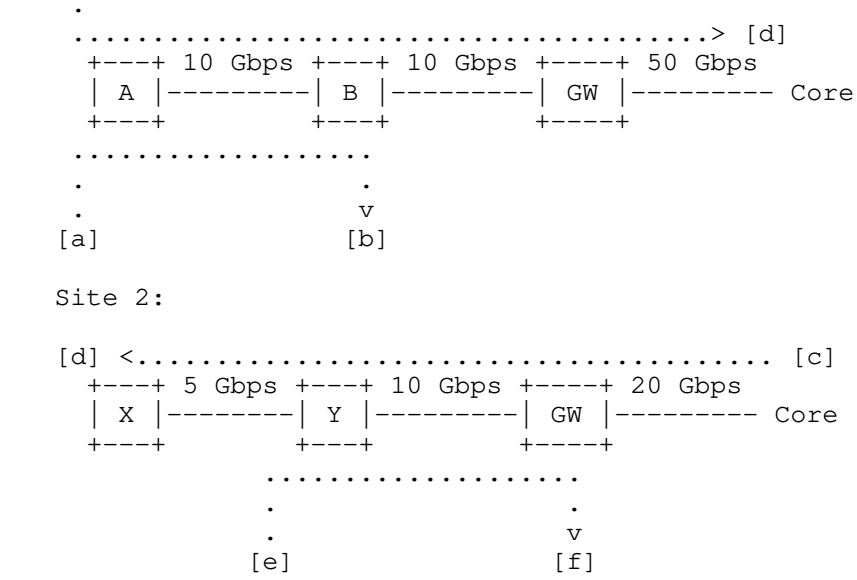


Figure 5: Example: Three Flows in Two Sites

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:

c: { d: [anei, aneii, aneiii] }
e: { f: [aneiv] }

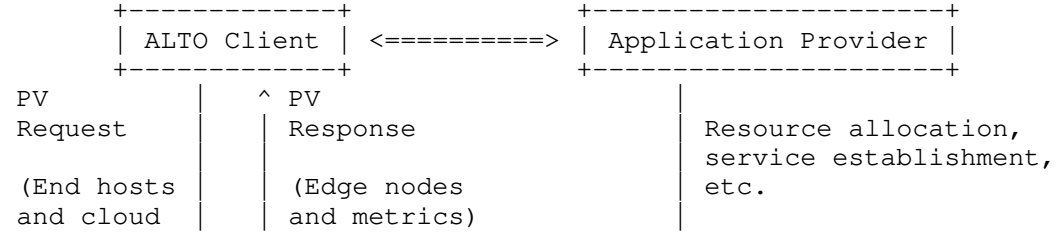
anei: bw = 5 Gbps (link Y->X)
aneii: bw = 10 Gbps (link GW->Y)
aneiii: bw = 20 Gbps (link Core->GW)
aneiv: bw = 10 Gbps (link Y->GW)

With this 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 CDNs and Service Edges

At the time of this writing, a growing trend in today's applications is to bring storage and computation closer to the end users for better QoE, such as CDNs, augmented reality / virtual reality, and cloud gaming, as reported in various documents (e.g., [SEREDGE] and [MOWIE]). ISPs may deploy multiple layers of CDN caches or, more generally, service edges, with different latencies and available resources, including the number of CPU cores, memory, and storage.

For example, Figure 6 illustrates a typical edge-cloud scenario where memory is measured in gigabytes (GB) and storage is measured in terabytes (TB). The "on-premise" edge nodes are closest to the end hosts and have the lowest latency, and the site-radio edge node and access central office (CO) have higher latencies but more available resources.



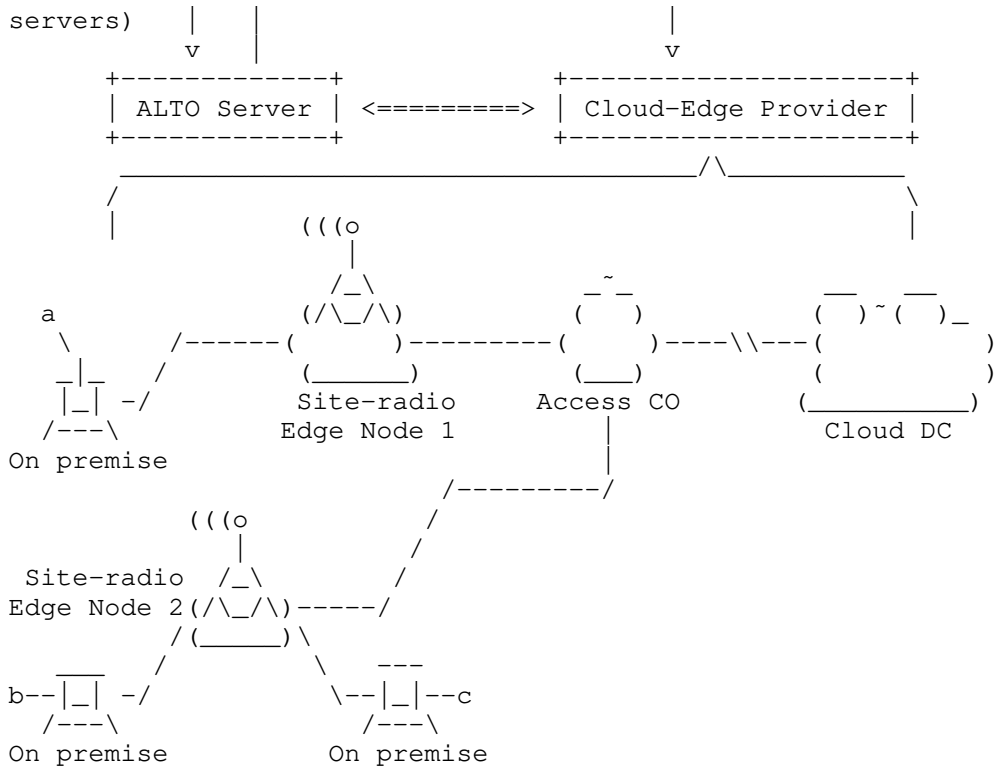


Figure 6: Example Use Case for Service Edge Exposure

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 (e.g., storage capabilities or Graphics Processing Unit (GPU) capabilities) 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 that the properties here are only used for illustration purposes and are not part of this extension.

```
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 = 5 ms  cpu = 2  memory = 8 GB  storage = 10 TB
(On premise, a)

ane2: latency = 20 ms  cpu = 4  memory = 8 GB  storage = 10 TB
(Site-radio Edge Node 1)

ane3: latency = 100 ms  cpu = 8  memory = 128 GB  storage = 100 TB
(Access CO)

ane4: latency = 20 ms  cpu = 4  memory = 8 GB  storage = 10 TB
(Site-radio Edge Node 2)

ane5: latency = 5 ms  cpu = 2  memory = 8 GB  storage = 10 TB
(On premise, b)

ane6: latency = 5 ms  cpu = 2  memory = 8 GB  storage = 10 TB
(On premise, c)
```

Figure 7: Example Service Edge Query Results

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 in place for customized quality of experience.

5. Path Vector Extension: Overview

This section provides a non-normative overview of the Path Vector extension defined in this document. It is assumed that readers are familiar with both the base protocol [RFC7285] and the entity property map extension [RFC9240].

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

1. introduces the concept of an ANE as the abstraction of components in a network whose properties may have an impact on end-to-end performance of the traffic handled by those 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

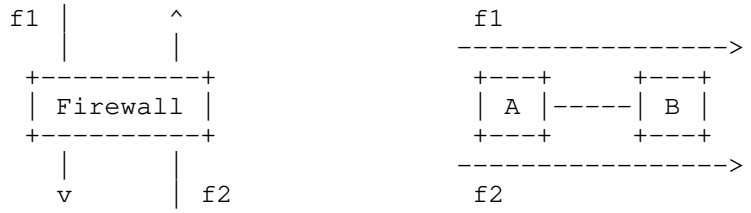
3. uses the entity property map to convey the association between the ANEs and their properties.

Thus, an ALTO client can learn about the ANEs that are important for assessing the QoE of different <source, destination> pairs by investigating the corresponding Path Vector value (AR1) and can also (1) identify common ANEs if an ANE appears in the Path Vectors of multiple <source, destination> pairs (AR2) and (2) retrieve the properties of the ANEs by searching the entity property map (AR3).

5.1. Abstract Network Element (ANE)

This extension introduces the 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 end-to-end performance for application traffic between endpoints.

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. In the example below, assume that 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 an ALTO server, it is assigned an identifier by the ALTO server, 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 ANEs and their properties are conveyed in an entity property map. Thus, ANEs must constitute an "entity domain" (Section 5.1 of [RFC9240]), and each ANE property must be an entity property (Section 5.2 of [RFC9240]).

Specifically, this document defines a new entity domain called "ane" as specified in Section 6.2; Section 6.4 defines two initial property types for the ANE entity domain.

5.1.2. Ephemeral 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 a Path Vector response or the incremental update stream for a Path Vector request. Compared with globally unique ANE names, ephemeral ANEs have several benefits, including better privacy for the ISP's internal structure 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 it is difficult to infer the underlying topology with multiple queries.

However, sometimes an ISP may intend to selectively reveal some "persistent" network components that, as opposed 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 its properties. See Sections 6.2.4 and 6.4.2 for more detailed instructions on how to identify ephemeral ANEs and persistent ANEs.

5.1.3. Property Filtering

Resource-constrained ALTO clients (see Section 4.1.2 of [RFC7285]) 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 (IRD) as a new filtering capability called "ane-property-names". The properties selected by a client as being of interest are specified in the subsequent Path Vector queries using the "ane-property-names" filter. The response only includes the selected properties for the ANEs.

The "ane-property-names" capability for the cost map and the Endpoint Cost Service is specified in Sections 7.2.4 and 7.3.4, respectively. The "ane-property-names" filter for the cost map and the Endpoint Cost Service is specified in Sections 7.2.3 and 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" parameters, 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++, if there existed a JSON array type named JSONArray, a Path Vector would 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 that the value is a JSON array. Then, an ALTO client must check the value of the "cost-metric" parameter. 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 resource, 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.

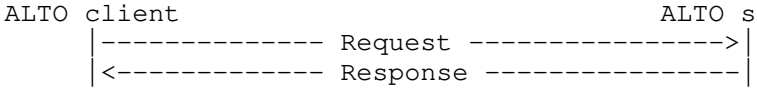


Figure 8: A Typical ALTO Request and Response

The extension defined in this document, on the other hand, involves two types of information resources: Path Vectors conveyed in an InfoResourceCostMap data component (defined in Section 11.2.3.6 of [RFC7285]) or an InfoResourceEndpointCostMap data component (defined in Section 11.5.1.6 of [RFC7285]), and ANE properties conveyed in an InfoResourceProperties data component (defined in Section 7.6 of [RFC9240]).

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 7.3.6 for details.

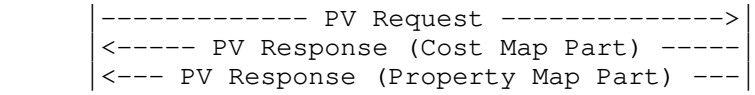


Figure 9: The Path Vector Extension Request and Response

This design is based on the following considerations:

1. 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. This "stateful" behavior may substantially harm server scalability and potentially lead to denial-of-service attacks.

One approach for realizing 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 the "multipart/related" media type as defined in [RFC2387] to elegantly combine the objects. Path Vectors are encoded in an InfoResourceCostMap data component or InfoResourceEndpointCostMap data component, and the property map is encoded in an InfoResourceProperties data component. They are encapsulated as parts of a multipart message. This 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 Object Root

ALTO uses a media type to indicate the type of an entry in the IRD (e.g., "application/alto-costmap+json" for the cost map and "application/alto-endpointcost+json" for the Endpoint Cost Service). Simply using "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 object root 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 design provided in [RFC8895], this extension requires that an ALTO server assign 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 field. 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 Entity Domain

The ANE entity domain associates property values with the ANEs in a property map. Accordingly, the ANE entity 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 as defined in Section 3.2 of [RFC9240]; the entity domain represents the set of valid entities defined by an ALTO information resource (called the "defining information resource").

6.2.1. Entity Domain Type

The entity domain type is "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 any persistent ANE whose defining resource is a property map resource, its entity domain name MUST have the format of "PROPMAP.ane", where PROPMAP is the resource ID of the defining resource. Persistent entities are "persistent" because standalone queries can be made by an ALTO client to their defining resource(s) 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 an entity property name (Section 5.2.2 of [RFC9240]).

6.4. Initial ANE Property Types

Two initial ANE property types are specified: "max-reservable-bandwidth" and "persistent-entity-id".

Note that these property types do not depend on any information resources. As such, the "EntityPropertyName" parameter MUST only have the EntityPropertyType part.

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 bits per second (bps). If this property is requested by the ALTO client but is not present for an ANE in the server response, it MUST be interpreted as meaning 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 a Software-Defined Networking (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 in that the Path Vector part and property map part are placed in the same JSON object.

In such a framework, the ALTO server exposes resource availability information (e.g., reservable bandwidth) 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 [INTENT-BASED-NETWORKING] or MPLS tunnels), which are out of scope for this document.

6.4.2. Persistent Entity ID

This document enables the discovery of a persistent ANE by exposing its entity identifier as the persistent entity ID property of an ephemeral ANE in the path vector response. The value of this property is encoded with the EntityID format defined in Section 5.1.3 of [RFC9240].

In this format, the entity ID combines:

* a defining information resource for the ANE on which a

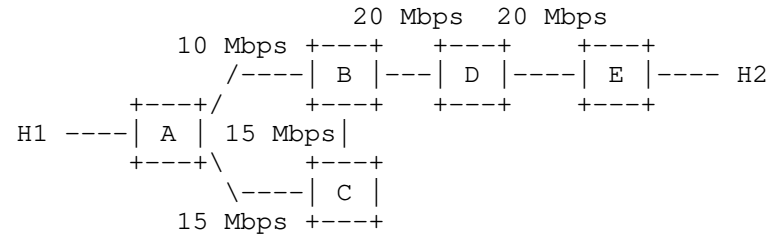
"persistent-entity-id" is queried, which is the property map resource defining the ANE as a persistent entity, together with the properties.

* 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 five nodes. Assume that the client wants to query the maximum reservable bandwidth from H1 to H2. An ALTO server may split the network into two ANEs: "ane1", which represents the subnetwork with routers A, B, and C; and "ane2", which 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 cycles of service edges are typically long, the service edges may contain information that is not specific to the query. Such information can be stored in an individual entity property map and can later be accessed by an ALTO client.

For example, "ane1" in Figure 7 represents the on-premise service edge closest to host "a". Assume that the properties of the service edges are provided in an entity property map called "se-props" and the ID of the on-premise service edge is "9a0b55f7-7442-4d56-8a2c-b4cc6a8e3aa1"; the "persistent-entity-id" setting for "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 that 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.

When the Path Vector procedures defined in this document are in use, an ALTO server using the "ane-path" cost metric and the "array" cost mode (see Section 6.5.2) MUST return as the cost value a JSON array of data type ANEName, and the client MUST also check that each element contained in the array is an ANEName (Section 6.1). Otherwise, the client MUST discard the response and SHOULD follow the guidance in Section 8.3.4.3 of [RFC7285] to handle the error.

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. While this cost mode can be applied to all cost metrics, additional specifications will be needed to clarify the semantics of the "array" cost mode when combined with cost metrics other than "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 the 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" as specified in Section 3.2.3 of [RFC5322]. It must be the domain name of the ALTO server.

7. Specification: Service Extensions

7.1. Notation

This document uses the same syntax and notation as those introduced in Section 8.2 of [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 the "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", and the required "type" parameter MUST have a value of "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 [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 field:

ane-property-names: This field provides 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 (Section 7.2.4). If the field is 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" setting 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: 212
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 field:

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

This extension also introduces additional restrictions for the following 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 IRD's "meta" field.

cost-constraints: If the "cost-type-names" field includes the Path Vector cost type, the "cost-constraints" field MUST be either "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.

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 field of the response MUST be "multipart/related" as defined by [RFC2387], with the following parameters:

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

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

boundary: The "boundary" parameter is as defined in Section 5.1.1 of [RFC2046] and is mandatory.

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 that 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 that 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 object. 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 the IRD.

* The entity 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 entity property map part is a JSON object with the same format as that defined in Section 7.6 of [RFC9240]. 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" field. 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 object 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 [RFC9240]. The EntityProps object 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" field 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.

The Path Vector part, whose media type is the same as the "type" parameter of the multipart response message, is the root body part as defined in [RFC2387]. Thus, it is the element that 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 placed 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 that the first part is the object root.

Example: Consider the network in Figure 1. The response to 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: 911
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": "fb20b76204814e9db37a51151faaaef2"
    },
    "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": "fb20b76204814e9db37a51151faaaef2"
      }
    ]
  },
```

```
    "property-map": {
      ".ane:ANE1": { "max-reservable-bandwidth": 100000000 }
    }
  }
  --example-1
```

7.3. Multipart Endpoint Cost Service for Path Vector

This document introduces a new ALTO resource called the "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", and the required "type" parameter MUST have a value of "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 ReqEndpointCostMap in Section 4.2.2 of [RFC8189], with a data format indicated by the media type "application/alto-endpointcostparams+json", which is a JSON object of type PVReqEndpointCostMap:

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

with field:

ane-property-names: This document defines the "ane-property-names" field in PVReqEndpointCostMap as being 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" setting 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: 238
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 being 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 field of the response MUST be "multipart/related" as defined by [RFC2387], with the following parameters:

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

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

boundary: The "boundary" parameter is as defined in Section 5.1.1 of [RFC2046] and is mandatory.

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-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 that 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 that defined in Section 4.2.3 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 object. 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 entity 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 entity property map part MUST be a JSON object with the same format as that defined in Section 7.6 of [RFC9240]. 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" field. 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 object 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 [RFC9240]. The EntityProps object 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" field 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.

The Path Vector part, whose media type is the same as the "type" parameter of the multipart response message, is the root body part as defined in [RFC2387]. Thus, it is the element that 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 placed 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 that the first part is the object root.

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

```
HTTP/1.1 200 OK
Content-Length: 899
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": "ec137bb78118468c853d5b622ac003f1"
    },
    "dependent-vtags": [
      {
        "resource-id": "my-default-networkmap",
        "tag": "677fe5f4066848d282ece213a84f9429"
      }
    ],
    "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": "ec137bb78118468c853d5b622ac003f1"
      }
    ]
  },
  "property-map": {
    ".ane:ANE1": { "max-reservable-bandwidth": 100000000 }
  }
}
--example-1
```

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. Sample Setup

Figure 10 illustrates the network properties and thus the message contents. There are three subnetworks (NET1, NET2, and NET3) and two interconnection links (L1 and L2). It is assumed that each subnetwork has sufficiently large bandwidth to be reserved.

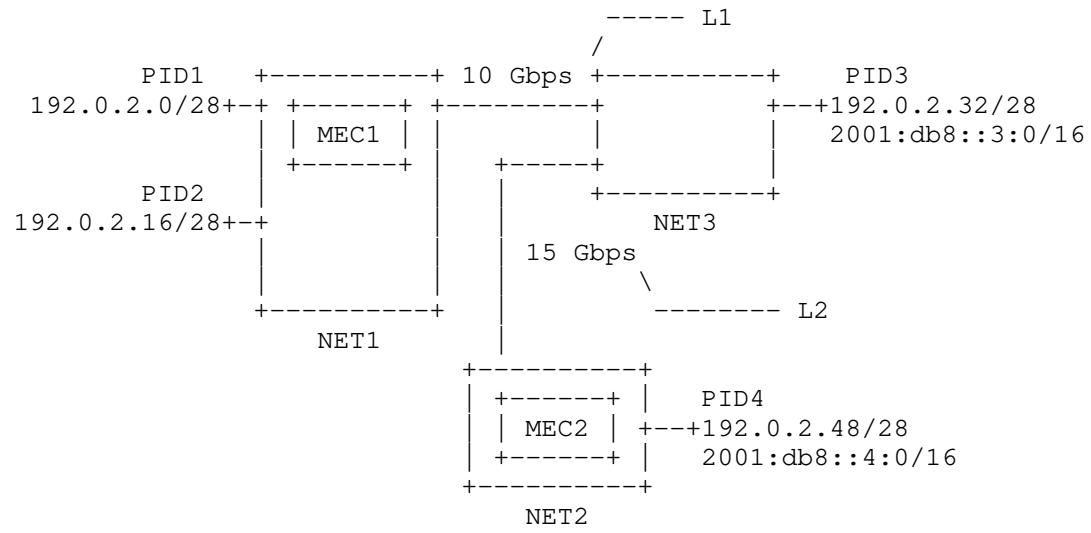


Figure 10: Examples of ANE Properties

8.2. 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 that contains the PIDs in the network.

"filtered-cost-map-pv": A multipart filtered cost map resource for the Path Vector. Exposes the "max-reservable-bandwidth" property for the PIDs in "my-default-networkmap".

"ane-props": A filtered entity property resource that exposes the information for persistent ANEs in the network.

"endpoint-cost-pv": A multipart Endpoint Cost Service for the Path Vector. Exposes the "max-reservable-bandwidth" and "persistent-entity-id" properties.

"update-pv": An update stream service that provides the incremental update service for the "endpoint-cost-pv" service.

"multicost-pv": A multipart Endpoint Cost Service with both the Multi-Cost extension and Path Vector extension enabled.

Below is the IRD of the example ALTO server. To enable the extension defined in this document, the Path Vector cost type (Section 6.5), represented by "path-vector" below, 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",
    "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. 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 the Path Vector and not the ANE properties.

The response consists of two parts:

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

* The second part returns the property map. Note that the properties of the ANE entries are equal to the literal string "{}" (see Section 8.3 of [RFC9240]).

```
POST /costmap/pv HTTP/1.1
Host: alto.example.com
Accept: multipart/related;type=application/alto-costmap+json,
        application/alto-error+json
Content-Length: 163
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: 952
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": "c04bc5da49534274a6daeee8ea1dec62"
      }
    ],
    "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": {
    ".ane:L1": {},
    ".ane:L2": {}
  }
}
--example-1
```

8.4. 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 ID 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 data type 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 NET3, 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 that NET1, NET2, and NET3 have 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 that there are no prior reservations 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 that the ANEName values 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 values for NET1 and NET2 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: 383
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: 1508
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
    }
  }
}
--example-2
```

In certain scenarios where the traversal order is not crucial, an ALTO server implementation may choose not to strictly follow the physical traversal order and may even obfuscate the order intentionally 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" property of "AGGR1" takes the value of L1, which is smaller than that of NET1, and the "persistent-entity-id" property of "AGGR1" takes the value of NET1. The properties of "AGGR2" are computed in a similar way; the obfuscated response is as shown below. Note that the obfuscation of Path Vector responses is implementation specific and is out of scope for this document. Developers may refer to Section 11 for further references.

```
HTTP/1.1 200 OK
Content-Length: 1333
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": "bb975862fbe3422abf4dae386b132c1d"
    },
    "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": "bb975862fbe3422abf4dae386b132c1d"
      },
      {
        "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
      }
    }
  }
  --example-2
```

8.5. 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: 120

{
  "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 a 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, based on Section 6.7.3 of [RFC8895].

```
event: application/merge-patch+json,
   ecspvsub1.ecsmap@alto.example.com
```

```
data: <Merge patch for endpoint-cost-map-update>

event: application/merge-patch+json,
   ecspvsub1.propmap@alto.example.com
data: <Merge patch for property-map-update>
```

8.6. 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 ID 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 entries 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 contains a property map that maps the ANEs to their requested properties.

```
POST /endpointcost/mcpv HTTP/1.1
Host: alto.example.com
Accept: multipart/related;
        type=application/alto-endpointcost+json,
        application/alto-error+json
Content-Length: 454
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: 1419
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": "84a4f9c14f9341f0983e3e5f43a371c8"
      },
      "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" ], 3],
        "ipv4:192.0.2.50":   [[ "NET3", "AGGR2" ], 2]
      },
      "ipv6:2001:db8::3:1": {
        "ipv6:2001:db8::4:1": [[ "NET3", "AGGR2" ], 2]
      }
    }
  }
  --example-4
  Content-ID: <propmap@alto.example.com>
  Content-Type: application/alto-propmap+json

  {
    "meta": {
      "dependent-vtags": [
        {
          "resource-id": "endpoint-cost-pv.ecs",
          "tag": "84a4f9c14f9341f0983e3e5f43a371c8"
        },
        {
          "resource-id": "ane-props",
          "tag": "be157afa031443a187b60bb80a86b233"
        }
      ]
    },
    "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
      }
    }
  }
  --example-4
```

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 have no backward-compatibility issues 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 problems.

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 be either "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 Sections 7.2.4 and 7.3.4.

9.3. Compatibility with Incremental Update Extension

This extension is compatible with the incremental update extension [RFC8895]. ALTO clients and servers MUST follow the specifications given in Sections 5.2 and 6.7.3 of [RFC8895] to support incremental updates for a Path Vector resource.

9.4. Compatibility with Cost Calendar Extension

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, 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 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 Cost 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 Discussion

10.1. Constraint Tests for General Cost Types

The constraint test is a simple approach for querying the data. It allows users to filter query results by specifying some boolean tests. This approach is already used in the ALTO Protocol. ALTO clients are permitted to specify either the "constraints" test [RFC7285] [RFC8189] or the "or-constraints" test [RFC8189] to better filter the results.

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 such implementations already exist (the authors have not done an exhaustive search to determine whether such implementations exist). One avenue for developing 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.

The extension specified in this document 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 extending the SSE mechanism as used in the incremental update extension [RFC8895]; or upgrading to HTTP/2 [RFC9113] and HTTP/3 [RFC9114], which provides the ability to multiplex queries and to allow servers to proactively send related information resources.

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

11. Security Considerations

This document is an extension of the base ALTO Protocol, so the security considerations provided for the base ALTO Protocol [RFC7285] fully apply when this extension is provided by an ALTO server.

The Path Vector extension requires additional scrutiny of 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 services (Section 15.5 of [RFC7285]).

For confidentiality of ALTO information, a network operator should be aware that this extension may introduce a new risk: the Path Vector information, when used together with sensitive ANE properties such as capacities of bottleneck links, 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 attacker may identify the bottleneck link or links and start a distributed denial-of-service (DDoS) attack involving minimal flows, triggering in-network congestion. Given the potential risk of leaking sensitive information, the Path Vector extension is mainly applicable in scenarios where 1) the ANE structures and ANE properties do not impose security risks on the ALTO service provider (e.g., they do not carry sensitive information) or 2) the ALTO server and client have established a reliable trust relationship (e.g., they operate in the same administrative domain or are managed by business partners with legal contracts).

Three risk types are identified in Section 15.3.1 of [RFC7285]:

(1) excess disclosure of the ALTO service provider’s data to an unauthorized ALTO client,

(2) disclosure of the ALTO service provider’s data (e.g., network topology information or endpoint addresses) to an unauthorized third party, and

(3) excess retrieval of the ALTO service provider’s data by collaborating ALTO clients.

To mitigate these risks, an ALTO server MUST follow the guidelines in Section 15.3.2 of [RFC7285]. Furthermore, an ALTO server MUST follow the following additional protections strategies for risk types (1) and (3).

For risk type (1), an ALTO server MUST use the authentication methods specified in Section 15.3.2 of [RFC7285] to authenticate the identity of an ALTO client and apply access control techniques to restrict the

retrieval of sensitive Path Vector information by unprivileged ALTO clients. For settings where the ALTO server and client are not in the same trust domain, the ALTO server should reach agreements with the ALTO client regarding protection of confidentiality before granting access to Path Vector services with sensitive information. Such agreements may include legal contracts or Digital Rights Management (DRM) techniques. Otherwise, the ALTO server MUST NOT offer Path Vector services that carry sensitive information to the clients, unless the potential risks are fully assessed and mitigated.

For risk type (3), an ALTO service provider must be aware that persistent ANEs may be used as "landmarks" in collaborative inferences. Thus, they should only be used when exposing public service access points (e.g., API gateways, CDN Interconnections) and/or when the granularity is coarse grained (e.g., when an ANE represents an AS, a data center, or a WAN). Otherwise, an ALTO server MUST use dynamic mappings from ephemeral ANE names to underlying physical entities. Specifically, for the same physical entity, an ALTO server SHOULD assign a different ephemeral ANE name when the entity appears in the responses to different clients or even for different requests from the same client. A RECOMMENDED assignment strategy is to generate ANE names from random numbers.

Further, to protect the network topology from graph reconstruction (e.g., through isomorphic graph identification [BONDY]), the ALTO server SHOULD consider protection mechanisms to reduce information exposure or obfuscate the real information. When doing so, the ALTO server must be aware that information reduction/obfuscation may lead to a potential risk of undesirable guidance from authenticated ALTO information (Section 15.2 of [RFC7285]).

Thus, 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]. However, these obfuscation methods are experimental, and their practical applicability to the generic capability provided by this extension has not been fully assessed. The ALTO server MUST carefully verify that the deployment scenario satisfies the security assumptions of these methods before applying them to protect Path Vector services with sensitive network information.

For availability of ALTO services, an ALTO server should be cognizant that using a Path Vector extension might introduce a new risk: frequent requests for Path Vectors might consume intolerable amounts of server-side computation and storage. This behavior can break the ALTO server. For example, if an ALTO server implementation dynamically computes the Path Vectors for each request, the service that provides the 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 such optimizations as precomputation-and-projection mechanisms [MERCATOR] to reduce the overhead for processing each query. An ALTO server may also protect itself from malicious clients by monitoring client behavior and stopping service to clients that exhibit suspicious behavior (e.g., sending requests at a high frequency).

The ALTO service providers must be aware that providing incremental updates of "max-reservable-bandwidth" may provide information about other consumers of the network. For example, a change in value may indicate that one or more reservations have been made or changed. To mitigate this risk, an ALTO server can batch the updates and/or add a random delay before publishing the updates.

12. IANA Considerations

12.1. "ALTO Cost Metrics" Registry

This document registers a new entry in the "ALTO Cost Metrics"

registry, per Section 14.2 of [RFC7285]. The new entry is as shown below in Table 1.

Identifier	Intended Semantics	Reference
ane-path	See Section 6.5.1	RFC 9275

Table 1: "ALTO Cost Metrics" Registry

12.2. "ALTO Cost Modes" Registry

This document registers a new entry in the "ALTO Cost Modes" registry, per Section 5 of [RFC9274]. The new entry is as shown below in Table 2.

Identifier	Description	Intended Semantics	Reference
array	Indicates that the cost value is a JSON array	See Section 6.5.2	RFC 9275

Table 2: "ALTO Cost Modes" Registry

12.3. "ALTO Entity Domain Types" Registry

This document registers a new entry in the "ALTO Entity Domain Types" registry, per Section 12.3 of [RFC9240]. The new entry is as shown below in Table 3.

Identifier	Entity Identifier Encoding	Hierarchy and Inheritance	Media Type of Defining Resource	Mapping to ALTO Address Type
ane	See Section 6.2.2	None	application/alto-propmap+json	false

Table 3: "ALTO Entity Domain Types" 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.

Mapping to ALTO Address Type: This entity type does not map to an ALTO address type.

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. If a naming schema is used to generate ANE names, either used privately or standardized by a future extension, how (or if) the naming schema relates to private information and network proximity must be explained to ALTO implementers and service providers.

12.4. "ALTO Entity Property Types" Registry

Two initial entries -- "max-reservable-bandwidth" and "persistent-entity-id" -- are registered for the ALTO domain "ane" in the "ALTO Entity Property Types" registry, per Section 12.4 of [RFC9240]. The

two new entries are shown below in Table 4, and their details can be found in Sections 12.4.1 and 12.4.2 of this document.

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 4: Initial Entries for the "ane" Domain in the "ALTO Entity Property Types" Registry

12.4.1. 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: To make better choices regarding bandwidth reservation, this property is essential for applications such as large-scale data transfers or an overlay network interconnection. 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 such protection mechanisms as providing the information to authorized clients only and applying information reduction and obfuscation as discussed in Section 11.

12.4.2. 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. As mentioned in Section 12.3.2 of [RFC9240], the entity IDs may reveal sensitive information about the underlying network. An ALTO server should follow the security considerations provided in Section 11 of [RFC9240].

13. References

13.1. Normative References

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, DOI 10.17487/RFC2046, November 1996, <https://www.rfc-editor.org/info/rfc2046>.

[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/info/rfc2119>.

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

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008,

<https://www.rfc-editor.org/info/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/info/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/info/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/info/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/info/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/info/rfc8896>.

[RFC9240] Roome, W., Randriamasy, S., Yang, Y., Zhang, J., and K. Gao, "An Extension for Application-Layer Traffic Optimization (ALTO): Entity Property Maps", RFC 9240, DOI 10.17487/RFC9240, July 2022, <https://www.rfc-editor.org/info/rfc9240>.

[RFC9274] Boucadair, M. and Q. Wu, "A Cost Mode Registry for the Application-Layer Traffic Optimization (ALTO) Protocol", RFC 9274, DOI 10.17487/RFC9274, July 2022, <https://www.rfc-editor.org/info/rfc9274>.

13.2. Informative References

[ALTO-PERF-METRICS] Wu, Q., Yang, Y., Lee, Y., Dhody, D., Randriamasy, S., and L. Contreras, "ALTO Performance Cost Metrics", Work in Progress, Internet-Draft, draft-ietf-alto-performance-metrics-28, 21 March 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-alto-performance-metrics-28>.

[BONDY] Bondy, J.A. and R.L. Hemminger, "Graph reconstruction--a survey", Journal of Graph Theory, Volume 1, Issue 3, pp. 227-268, DOI 10.1002/jgt.3190010306, 1977, <https://onlinelibrary.wiley.com/doi/10.1002/jgt.3190010306>.

[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, DOI 10.1609/aaai.v33i01.33011674, July 2019, <https://ojs.aaai.org//index.php/AAAI/article/view/3984>.

[CLARINET] Viswanathan, R., Ananthanarayanan, G., and A. Akella, "CLARINET: WAN-aware optimization for analytics queries", Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation (OSDI'16), Savannah, GA, pp. 435-450, November 2016, <https://dl.acm.org/doi/abs/10.5555/3026877.3026911>.

[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, DOI 10.1145/3366707, December 2019, <https://dl.acm.org/doi/10.1145/3366707>.

[HUG] Chowdhury, M., Liu, Z., Ghodsi, A., and I. Stoica, "HUG: multi-resource fairness for correlated and elastic demands", Proceedings of the 13th USENIX Conference on Networked Systems Design and Implementation (NSDI'16), Santa Clara, CA, pp. 407-424, March 2016, <https://dl.acm.org/doi/10.5555/2930611.2930638>.

[INTENT-BASED-NETWORKING] Clemm, A., Ciavaglia, L., Granville, L. Z., and J. Tantsura, "Intent-Based Networking - Concepts and Definitions", Work in Progress, Internet-Draft, draft-irtf-nmrg-ibn-concepts-definitions-09, 24 March 2022, <https://datatracker.ietf.org/doc/html/draft-irtf-nmrg-ibn-concepts-definitions-09>.

[JSONiq] JSONiq, "The JSON Query Language", 2022, <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 in Communications, Volume 37, Issue 8, pp. 1924-1940, DOI 10.1109/JSAC.2019.2927073, August 2019, <https://ieeexplore.ieee.org/document/8756056>.

[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", Proceedings of the Workshop on Network Application Integration/CoDesign (NAI '20), ACM, Virtual Event USA, pp. 20-27, DOI 10.1145/3405672.3409489, August 2020, <https://dl.acm.org/doi/10.1145/3405672.3409489>.

[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, Issue 2, pp. 805-818, DOI 10.1109/TNET.2019.2899905, April 2019, <https://doi.org/10.1109/TNET.2019.2899905>.

[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", SC18: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 58-70, DOI 10.1109/SC.2018.00008, November 2018, <https://ieeexplore.ieee.org/document/8665783>.

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

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/info/rfc4271>.

[RFC9113] Thomson, M., Ed. and C. Benfield, Ed., "HTTP/2", RFC 9113, DOI 10.17487/RFC9113, June 2022, <https://www.rfc-editor.org/info/rfc9113>.

[RFC9114] Bishop, M., Ed., "HTTP/3", RFC 9114, DOI 10.17487/RFC9114, June 2022, <https://www.rfc-editor.org/info/rfc9114>.

[SENSE] ESnet, "Software Defined Networking (SDN) for End-to-End Networked Science at the Exascale", 2019, <https://www.es.net/network-r-and-d/sense/>.

[SEREDGE] Contreras, L., Baliosian, J., MartÃ-nez-Julia, P., and J. Serrat, "Computing at the Edge: But, what Edge?", Proceedings of NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium, pp. 1-9, DOI 10.1109/NOMS47738.2020.9110342, April 2020, <https://ieeexplore.ieee.org/document/9110342>.

[SWAN] Hong, C., Kandula, S., Mahajan, R., Zhang, M., Gill, V., Nanduri, M., and R. Wattenhofer, "Achieving high utilization with software-driven WAN", Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM (SIGCOMM '13), New York, NY, pp. 15-26, DOI 10.1145/2486001.2486012, August 2013, <https://dl.acm.org/doi/10.1145/2486001.2486012>.

[UNICORN] Xiang, Q., Wang, T., Zhang, J., Newman, H., Yang, Y.R., and Y. Liu, "Unicorn: Unified resource orchestration for multi-domain, geo-distributed data analytics", Future Generation Computer Systems, Volume 93, pp. 188-197, DOI 10.1016/j.future.2018.09.048, April 2019, <https://www.sciencedirect.com/science/article/abs/pii/S0167739X18302413?via%3Dihub>.

[XQuery] Robie, J., Ed., Dyck, M., Ed., and J. Spiegel, Ed., "XQuery 3.1: An XML Query Language", W3C Recommendation, March 2017, <https://www.w3.org/TR/xquery-31/>.

Acknowledgments

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

The authors would also like to thank Tim Chown, Luis Contreras, Roman Danyliw, Benjamin Kaduk, Erik Kline, Suresh Krishnan, Murray Kucherawy, Warren Kumari, Danny Lachos, Francesca Palombini, Ã\211ric Vyncke, Samuel Weiler, and Qiao Xiang, whose feedback and suggestions were invaluable for improving the practicability and conciseness of this document; and Mohamed Boucadair, Martin Duke, Vijay Gurbani, Jan Seedorf, and Qin Wu, who provided great support and guidance.

Authors' Addresses

Kai Gao
Sichuan University
No.24 South Section 1, Yihuan Road
Chengdu
610000
China
Email: kaigao@scu.edu.cn

Young Lee
Samsung
Republic of Korea
Email: younglee.tx@gmail.com

Sabine Randriamasy
Nokia Bell Labs
Route de Villejust
91460 Nozay
France
Email: sabine.randriamasy@nokia-bell-labs.com

Yang Richard Yang
Yale University
51 Prospect Street
New Haven, CT 06511
United States of America
Email: yry@cs.yale.edu

Jingxuan Jensen Zhang
Tongji University
4800 Caoan Road
Shanghai
201804
China
Email: jingxuan.n.zhang@gmail.com