ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: June 21, 2018

G. Bernstein
Grotto Networking
S. Chen
Tongji University
K. Gao
Tsinghua University
Y. Lee
Huawei
W. Roome
M. Scharf
Nokia
Y. Yang
Yale University
J. Zhang
Tongji University

December 18, 2017

ALTO Extension: Path Vector Cost Type
draft-ietf-alto-path-vector-02.txt

Abstract

The Application-Layer Traffic Optimization (ALTO) protocol [RFC7285] has defined several resources and services to provide clients with basic network information. However, the base ALTO protocol and latest extensions only provide end-to-end metrics, which are insufficient to satisfy the demands of solving more complex network optimization problems. This document introduces an extension to the base ALTO protocol, namely the path-vector extension, which allows ALTO clients to query information such as capacity regions for a given set of flows. A non-normative example called multi-flow scheduling is presented to illustrate the limitations of existing ALTO (endpoint) cost maps. After that, details of the extension are defined.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

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

This Internet-Draft will expire on June 21, 2018.

Table of Contents

1. Introduction

The base ALTO protocol [RFC7285] is designed for exposing network information through services such as the Network Map service and the Cost Map service. These services use an extreme "single-node" network view abstraction, which represents the whole network with a single node and hosts with "endpoint groups" directly connected to the node.

Although the "single-node" network view abstraction works well in many settings, it lacks the ability to support new emerging use cases, such as inter-datacenter flow scheduling, scientific high-performance computing data transfers and end-to-end paths crossing heterogeneous technologies. For these use cases, more powerful network view abstraction is required. To provide a better network view abstraction, ALTO services need to support the following additional functionalities:

- Providing path vector rather than a simple path cost of endpoint to endpoint. The path vector exposes the network elements (e.g., links, switches, middle boxes and their aggregations) that endpoint to endpoint traffic goes through.

- Providing information of the network elements in the path vector. The information can be "bandwidth" for links, "delay" between neighboring switches and other properties of network elements. These information may help the application avoid network congestion, achieving better application performance.

To support these new functionalities, this document proposes the path-vector extension, which introduces a qualitative cost type listing selected groups of one or more abstracted network elements in an e2e path and optionally conveys some of their properties.

The rest of this document is organized as follows. Section 3 gives an example of flow scheduling and illustrates the limitations of the base ALTO protocol in such a use case. Section 4 gives an overview of the path-vector extension, before specifying the details of the extension in Section 5 and Section 6. Section 7 presents several examples, and Section 9 explains some design decisions. Section 8 discusses compatibility issues with some other ALTO extensions. Section 10 and Section 11 discusses about security and IANA considerations.

2. Terminology

This document uses the same terms as defined in [RFC7285], [RFC8189] and [I-D.ietf-alto-unified-props-new] with the following additional terms: Abstract Network Element, Abstract Network Element Name, Abstract Network Element Property, Abstract Network Element Property Map and Path Vector.

- Abstract Network Element (ANE): An abstract network element is an abstraction of network components, it can be an aggregation of links, middle boxes, Virtualized Network Function (VNF), or even a sub-network. An abstract network element has two attributes:

abstract network element name and abstract network element property, which are defined below.

- Abstract Network Element Name (ANEN): An abstract network element name is an identifier which uniquely identifies an abstract network element, as defined in Section 5.3.

- Abstract Network Element Property (ANEP): An abstract network element property is a specific metric associated with a given abstract network element, as introduced in Section 4.2. An abstract network element can have several network element properties.

- Abstract Network Element Property Map (ANE Property Map): An abstract network element property map is a Filtered Property Map defined in [I-D.ietf-alto-unified-props-new] which supports the "ane" domain in its "domain-types" capability.

- Path Vector (PV): A path vector is an array of ALTO Abstract Network Elements (ANEs), which presents an abstract network path between entities such as PIDs or endpoints. An ANE represents a selected part of an end-to-end path that the ALTO Server considers worth exposing. An ANE is a set of one or more network elements such as links, switches, middle boxes and their aggregations, it is expected to have properties that may influence the applications e.g. when they select an endpoint or want to estimate their performance.

3. Use Case: Capacity Region for Multi-Flow Scheduling

Once routing has been configured in the network, application-layer traffic optimization may want to schedule traffic among application-layer paths. Specifically, assume that an application has control over a set of flows F = {f_1, f_2, ..., f_|F|}. If routing is given, what the application can control is x_1, x_2, ..., x_|F|, where x_i is the amount of traffic for flow i. Let x = [x_1, ..., x_|F|] be the vector of the flow traffic amounts. Due to shared links, feasible values of x where link capacities are not exceeded can be a complex polytype.

Specifically, consider a network as shown in Figure 1. The network has 7 switches (sw1 to sw7) forming a dumb-bell topology. Switches sw1/sw3 provide access on one side, sw2/sw4 provide access on the other side, and sw5-sw7 form the backbone. End hosts eh1 to eh4 are connected to access switches sw1 to sw4 respectively. Assume that the bandwidth of link eh1 -> sw1 and link sw1 -> sw5 are 150 Mbps, and the bandwidth of the rest links are 100 Mbps.

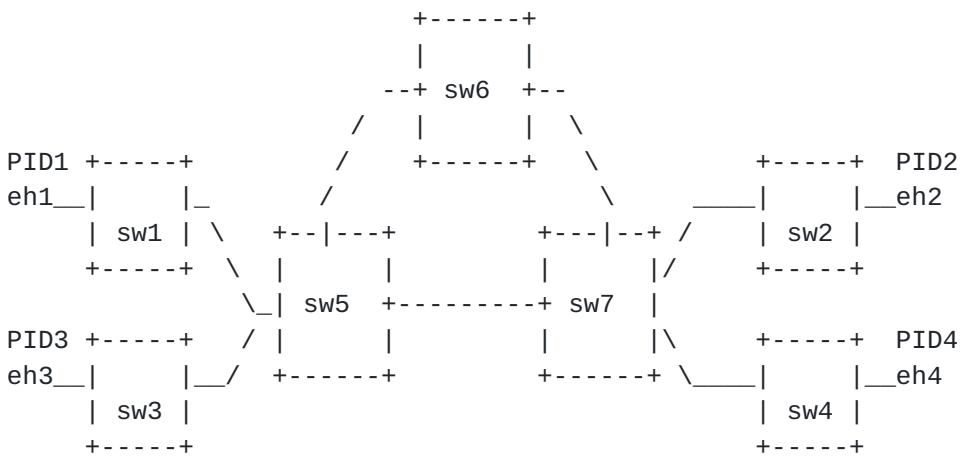

Figure 1: Raw Network Topology.

The single-node ALTO topology abstraction of the network is shown in Figure 2.

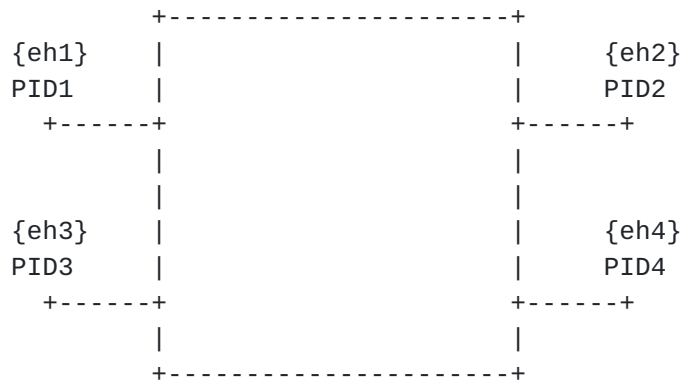

Figure 2: Base Single-Node Topology Abstraction.

Consider an application overlay (e.g., a large data analysis system) which wants to schedule the traffic among a set of end host source-destination pairs, say eh1 -> eh2 and eh1 -> eh4. The application can request a cost map providing end-to-end available bandwidth, using 'availbw' as cost-metric and 'numerical' as cost-mode.

The application will receive from ALTO server that the bandwidth of eh1 -> eh2 and eh1 -> eh4 are both 100 Mbps. But this information is not enough. Consider the following two cases:

- Case 1: If eh1 -> eh2 uses the path eh1 -> sw1 -> sw5 -> sw6 -> sw7 -> sw2 -> eh2 and eh1 -> eh4 uses path eh1 -> sw1 -> sw5 -> sw7 -> sw4 -> eh4, then the application will obtain 150 Mbps.

- Case 2: If eh1 -> eh2 uses the path eh1 -> sw1 -> sw5 -> sw7 -> sw2 -> eh2 and eh1 -> eh4 uses the path eh1 -> sw1 -> sw5 -> sw7 -> sw4 -> eh4, then the application will obtain only 100 Mbps.

To allow applications to distinguish the two aforementioned cases, the network needs to provide more details. In particular:

- The network needs to expose more detailed routing information to show the shared bottlenecks.

- The network needs to provide the necessary abstraction to hide the real topology information while providing enough information to applications.

The path-vector extension defined in this document meets all the requirements.

See [I-D.bernstein-alto-topo] for a survey of use-cases where extended network topology information is needed.

4. Overview of Path Vector Extensions

This section presents the approaches taken to support the path-vector extension. It assumes the readers are familiar with (Filtered) Cost Map and Endpoint Cost Service defined in [RFC7285] and their extensions defined in [RFC8189]. It also uses features such as Filtered Property Map defined in [I-D.ietf-alto-unified-props-new].

4.1. Path Vector Cost Type Extensions

None of current cost types defined in [RFC7285] can be used to convey path vector information. So, a new cost type with a new cost metric "ane-path" and a new cost mode "array" is defined in this document. Below are brief descriptions. Detailed information and specifications are given in Section 5.1.1 and Section 5.1.2.

4.1.1. New Cost Metric for Path Vector

To represent an abstract network path, this document introduces a new cost metric named "ane-path". A cost value in this metric is a list containing the names of the ALTO ANEs that the ALTO Server has specified as describing the network path elements. The ANE names array is organized as a sequence beginning at the source of the path and ending at its destination.

4.1.2. New Cost Mode for Path Vector

A cost mode as defined in Section 6.1.2 of [RFC7285], a cost mode is either "numerical" or "ordinal" and none of these can be used to present a list of ANE names. Therefore, this document specifies a new cost mode named "array" for the cost metric "ane-path". The new cost mode "array" means each cost value in the cost maps is a list.

4.1.3. Path Vector Cost Type Semantics

The new cost type follows the convention of the cost types in the legacy ALTO protocol. Table 1 lists some of the current defined cost types and their semantics.

Cost Mode	Cost Metric	Semantics
numerical	routingcost	a number representing the routing cost
numerical	hopcount	a number representing the hop count
ordinal	routingcost	a ranking representing the routing cost
ordinal	hopcount	a ranking representing the hop count
array	ane-path	a list representing the ane path

Table 1: Cost Types and Their Semantics

The "routingcost" and "hopcount" can encoded in "numerical" or "ordinal", however, the cost metric "ane-path" can only be applied to the cost mode "array" defined in this document to convey path vector information. The cost metric "ane-path" can not be used in "numerical" or "ordinal" unless it is defined in future extensions. If the ALTO server declares that it support cost type with cost metric being "ane-path" and cost mode not being "array", the ALTO client SHOULD ignore them.

4.2. ANE Property Map

Given that Cost Map and Endpoint Cost service now provide the abstract network element names along a flow path, ALTO clients can learn that there exist bottlenecks shared by different flows. However, only providing the abstract network element names without abstract network element properties is not enough, some ALTO clients may want to have information on specific ANE properties such as link capacity or delay. This document adopts the property map resources defined in [I-D.ietf-alto-unified-props-new] to encode the properties of ANEs. Draft [I-D.ietf-alto-unified-props-new] defines a new

entity domain called "ane" and each entity in the "ane" domain has an identifier of an ANE. An ANE identifier is the ANE name used in the values of the "ane-path" metric defined in the present draft. ANE properties are provided in information resources called "Property Map Resource" and "Filtered Property Map Resource". The "Filtered Property Map" resource which support the "ane" domain is used to encode the properties of ane entities, and it is called an ANE Property Map in this document.

4.3. media type for path vector: multipart/related

In the legacy ALTO protocol, ALTO servers use media types in the HTTP header to indicate the type of the response. Typically one response only contains a single media type, such as "application/alto-costmap+json" or "application/alto-propmap+json". This has limited the capability of ALTO servers to return multiple services in a single response.

Thus, an ALTO client needs to make separate queries to get the information of related services. This may cause a data synchronization problem between dependent ALTO services because when making the second query, the result for the first query may have already changed. The very same problem can happen to Network Map and Cost Map resources. However, unlike Network Map and Cost Map which are considered more stable, Path Vectors and the dependent ANE Property Maps might change more frequently.

Instead of introducing a new media type to encapsulate multiple types in a single response, this document adopts the "multipart/related" media type defined in [RFC2387]. In this way, a response can contain both the Path Vectors in a Filtered Cost Map (or Endpoint Cost Map) and the associated ANE Property Map. The media types of the cost map and the property map can still be retrieved from the response. The interpretation of each media type in the "multipart/related" response is consistent with the base ALTO protocol.

4.4. Applicable ALTO services for Path Vector costs

This document defines Filtered Cost Map and Endpoint Cost Map are applicable for path vector costs. Although the new cost type for path vector can also be used in the GET-mode Cost Map service from [RFC7285], the behaviours of the ALTO server and client for such a GET-mode service is not defined. So it is not recommended to apply path vector costs to the GET-mode Cost Map service.

4.5. Impact of backwards compatibility on the PV design

The path vector extension on Filtered Cost Map and Endpoint Cost Service is backward compatible with the base ALTO protocol. If the ALTO server provides path vector extended Filtered Cost Map or Endpoint Cost Service, but the client is a base ALTO client, then the client will ignore the path vector cost type without conducting any incompatibility. If the client sents a request with path vector cost type, but the server is a base ALTO server, the server will return an "E_INVALID_FIELD_VALUE" error.

4.6. Requirements for PV on Clients and Servers

A path vector extended ALTO server MUST implement the legacy ALTO protocol specified in [RFC7285] with the following additional requirements:

- If an ALTO server supports path vector extension, it MUST support the Unified Property Map defined in [I-D.ietf-alto-unified-props-new].

- If an ALTO server supports path vector extended Filtered Cost Map or Endpoint Cost Service, the server MUST provide the associated Property Map simultaneously.

- If an ALTO server provides "multipart/related" media type for path vector, the server MUST provide the associated Filtered Cost Map or Endpoint Cost Service and the Property Map simultaneously.

An ALTO client supported path vector extension MUST be able to interpret Unified Property Map correctly. If the ALTO client wants to interpret "multipart/related" path vector response, the client MUST implement the path vector extension on Filtered Cost Map or Endpoint Cost Service at first.

5. Path-Vector Extension: Basic Data Types

This section formally specifies a new cost type.

5.1. Cost Type

This document extends the cost types defined in Section 6.1 of [RFC7285] by introducing a new cost mode "array" and a new cost metric "ane-path".

5.1.1. Cost Mode: array

This document extends the CostMode defined in Section 10.5 of [RFC7285] with a new cost mode: "array". This cost mode indicates that every cost value in a cost map represents an array rather than a simple value. The values are arrays of JSONValue. The specific type of each element in the array depends on the cost metric.

5.1.2. Cost Metric: ane-path

This document specifies a new cost metric: "ane-path". This cost metric indicates that the cost value is a list of abstract network elements which the path from a source to a destination goes across. The values are arrays of ANE Names which are defined in Section 5.3.

The cost metric "ane-path" SHOULD NOT be used when the cost mode is not "array" unless it is explicitly specified by a future extension. If an ALTO client send queries with the cost metric "ane-path" and a non "array" cost mode, the ALTO server SHOULD return an error with the error code "E_INVALID_FIELD_VALUE"; If an ALTO server declares the support of a cost type with the cost metric "ane-path" and a non "array" cost mode, the ALTO client SHOULD assume such a cost type is invalid and ignore it.

5.2. ANE Domain

This document uses the same definition of entity domain name 'ane' as defined in Section 3.4 of [I-D.ietf-alto-unified-props-new].

5.3. Abstract Network Element Name

An Abstract Network Element Name is encoded as an EntityAddr of the "ane" domain as defined in Section 3.4.2 of [I-D.ietf-alto-unified-props-new].

6. Path-Vector Extension: Services

This section extends Filtered Cost Map Service and Endpoint Cost Service.

6.1. Filtered Cost Map Extensions

This document extends the Filtered Cost Map defined in Section 4.1 of [RFC8189].

The specifications for the "media type", "HTTP method" and "uses" are the same as defined in Section 4.1 of [RFC8189].

6.1.1. Capabilities

The FilteredCostMapCapabilities object is extended with a new member "property-map":

```
object {
  [ResourceID property-map;]
} PathVectorFilteredCostMapCapabilities : FilteredCostMapCapabilities
```

property-map: A resource ID defined in the same IRD pointing to an ANE Property Map as defined in Section 2. This field MUST be present if the path vector cost type is present in the "cost-type-names" field.

Other fields of the FilteredCostMapCapabilities object has the same format as defined in Section 4.1.1 of [RFC8189] with the following constraint:

testable-cost-type-names: The path vector cost type with "ane-path" as the cost metric and "array" as the cost mode MUST NOT be included in "testable-cost-type-names".

6.1.2. Accept Input Parameters

The ReqFilteredCostMap uses the same format as defined in Section 4.1.2 of [RFC8189], with the following constraints:

constraints, or-constraints: If the path vector cost type is included in either "cost-type" or "multi-cost-types", ALTO clients MUST NOT use it in "constraints" or "or-constraints". Otherwise, the ALTO server MUST return an error with error code "E_INVALID_FIELD_VALUE".

testable-cost-types: The path vector cost type MUST NOT be included in the "testable-cost-types" field. Otherwise, the ALTO server MUST return an error with error code "E_INVALID_FIELD_VALUE".

6.1.3. Response

If the ALTO client includes the path vector cost type in the "cost-type" or "multi-cost-types" field of the input parameter, the response use the same format as defined in Section 4.1.3 of [RFC8189], but the corresponding cost value MUST be encoded as a JSONArray of AbstractNetworkElementName.

6.2. Endpoint Cost Service Extensions

This document extends the Endpoint Cost Service defined in Section 4.2 in [RFC8189].

The specifications for "HTTP method" and "uses" are the same as defined in Section 4.2 in [RFC8189].

6.2.1. Capabilities

The same as defined in Section 6.1.1.

6.2.2. Accept Input Parameters

The ReqEndpointCostMap uses the same format as defined in Section 4.2.2 of [RFC8189], with the following constraints:

cost-type, multi-cost-types: ALTO clients MUST include the path vector cost type, e.g. the one with "ane-path" as cost metric and "array" as cost mode, in either "cost-type" or "multi-cost-types" to activate the path vector extension.

constraints, or-constraints: If the path vector cost type is included in either "cost-type" or "multi-cost-types", ALTO clients MUST NOT use it in "constraints" or "or-constraints". Otherwise, the ALTO server MUST return an error with error code "E_INVALID_FIELD_VALUE".

testable-cost-types: The path vector cost type MUST NOT be included in the "testable-cost-types" field. Otherwise, the ALTO server MUST return an error with error code "E_INVALID_FIELD_VALUE".

6.2.3. Response

If the ALTO client specifies the path vector cost type in the "cost-type" or "multi-cost-types" field of the input parameter, the response use the same format as defined in Section 4.2.3 of [RFC8189], but the corresponding cost value MUST be encoded as a JSONArray of AbstractNetworkElementName.

6.3. Multipart Cost Property Service

This document introduces a new ALTO service called "Multipart Cost Property Service", which provides the path vector information and the associated ANE property information in the same response.

6.3.1. Media Type

The media type of the Multipart Cost Property service is "multipart/related".

6.3.2. HTTP Method

The Multipart Cost Property service is requested using the HTTP POST method.

6.3.3. Accept Input Parameters

The input parameters of the Multipart Cost Property service MUST be encoded as a JSON object in the body of an HTTP POST request. The media type of the request SHOULD be one of "application/alto-costmapfilter+json" and "application/alto-endpointcostparams+json". The format of the request body depends on the media type:

- If the media type of the request is "application/alto-costmapfilter+json", the request body MUST be the same type as defined by Section 6.1.2.

- If the media type of the request is "application/alto-endpointcostparams+json", the request body MUST be the same type as defined by Section 6.2.2.

The path vector cost type MUST be the only cost type in the input parameter.

6.3.4. Capabilities

TBD

6.3.5. Uses

The "uses" attribute MUST be an array with at least one resource id. The first resource id MUST point to a Filtered Cost Map or an Endpoint Cost Service resource. And the path vector cost type MUST be in its "cost-type" capability. If there are more than one resource id in the "uses" attribute, the ALTO client SHOULD ignore any additional resource ids.

According to Section 6.1.1, the "property-map" field MUST be present in the first resource. So the ALTO client MUST infer that the Property Map pointed by the "property-map" field of the first resource is also a dependent resource.

6.3.6. Response

If an ALTO client sends a request of the media type "application/alto-costmapfilter+json" and accepts "multipart/related", the HTTP body of the response MUST consist of two parts with the media types "application/alto-costmap+json" and "application/alto-propmap+json" accordingly. The part with media type "application/alto-costmap+json" MUST be the first part. The content of the "application/alto-endpointcost+json" part has the same format as defined in Section 6.1.3.

If an ALTO client sends a request of the media type "application/alto-endpointcostparams+json" and accepts "multipart/related", the HTTP body of the response MUST consist of two parts with the media types "application/alto-endpointcost+json" and "application/alto-propmap+json" accordingly. The part with media type "application/alto-endpointcost+json" MUST be the first part. The content of the "application/alto-endpointcost+json" part has the same format as defined in Section 6.2.3.

7. Examples

This section lists some examples of path vector queries and the corresponding responses.

7.1. Workflow

This section gives a typical workflow of an ALTO client using the path-vector extension.

1. Send a GET request for the whole Information Resource Directory.

2. Look for the resource of the (Filtered) Cost Map/Endpoint Cost Service which contains the path vector cost type and get the resource ID of the dependent abstract network element property map.

3. Check whether the capabilities of the property map includes the desired "prop-types".

4. Send a path-vector request which accepts "multipart/related" media type following "application/alto-costmap+json" or "application/endpointcost+json".

7.2. Information Resource Directory Example

Here is an example of an Information Resource Directory. In this example, filtered cost map "cost-map-pv" doesn't support the multi-cost extension but support the path-vector extension, "endpoint-multicost-map" supports both multi-cost extension and path-vector extension. Filtered Property Map "propmap-delay-availbw" supports properties "availbw" and "delay", and "propmap-location" supports property "location".

```
{
  "meta": {
    "cost-types": {
      "pv": {
        "cost-mode": "array",
        "cost-metric": "ane-path"
      },
      "num-routingcost": {
        "cost-mode": "numerical",
        "cost-metric": "routingcost"
      },
      "num-hopcount": {
        "cost-mode": "numerical",
        "cost-metric": "hopcount"
      }
    }
  },
  "resources": {
    "my-default-networkmap": {
      "uri" : "http://alto.example.com/networkmap",
      "media-type" : "application/alto-networkmap+json"
    }
    "cost-map-pv" : {
      "uri": "http://alto.example.com/costmap/pv",
      "media-type": "application/alto-costmap+json",
      "accepts": "application/alto-costmapfilter+json",
      "capabilities": {
        "cost-type-names": [ "pv", "num-hopcount" ]
      },
      "property-map": "propmap-delay",
      "uses": [ "my-default-networkmap" ]
    },
    "endpoint-multicost-map" : {
      "uri": "http://alto.exmaple.com/endpointcostmap/multicost",
      "media-type": "application/alto-endpointcost+json",
      "accepts": "application/alto-endpointcostparams+json",
      "capabilities": {
        "cost-constraints": true,
```

```
            "cost-type-names": [ "pv", "num-routingcost" ],
            "max-cost-types": 2
          },
          "property-map": "propmap-availbw"
        },
        "propmap-availbw-delay" : {
          "uri": "http://alto.exmaple.com/propmap/availbw",
          "media-type": "application/alto-propmap+json",
          "accepts": "application/alto-propmapparams+json",
          "capabilities": {
            "domain-types": [ "ane" ],
            "prop-types": [ "availbw" ]
          }
        },
        "propmap-location" : {
          "uri": "http://alto.exmaple.com/propmap/delay",
          "media-type": "application/alto-propmap+json",
          "accepts": "application/alto-propmapparams+json",
          "capabilities": {
            "domain-types": [ "pid" ],
            "prop-types": [ "location" ]
          }
        }
      }
    }
```

7.3. Example # 1

```
POST /costmap/pv HTTP/1.1
Host: alto.example.com
Accept: multipart/related, application/alto-costmap+json,
        application/alto-propmap+json, application/alto-error+json
Content-Length: [TBD]
Content-Type: application/alto-costmapfilter+json

{
  "cost-type": {
    "cost-mode": "array",
    "cost-metric": "ane-path"
  },
  "pids": {
    "srcs": [ "PID1" ],
    "dsts": [ "PID2", "PID3" ]
  }
}

HTTP/1.1 200 OK
Content-Length: [TBD]
```

```
   Content-Type: multipart/related; boundary=42

   --42
   Content-Type: application/alto-costmap+json

   {
     "meta": {
       "dependent-vtags": [
         {
           "resource-id": "default-network-map",
           "tag": "75ed013b3cb58f896e839582504f622838ce670f"
         }
       ],
       "cost-type": {
         "cost-mode": "array",
         "cost-metric": "ane-path"
       },
     },

     "cost-map": {
       "PID1": {
         "PID2": [ "ane:L001", "ane:L003" ],
         "PID3": [ "ane:L001", "ane:L004" ]
       }
     }
   }

   --42
   Content-Type: application/alto-propmap+json

   {
     "property-map": {
       "ane:L001": { "delay": 46},
       "ane:L003": { "delay": 50},
       "ane:L004": { "delay": 70}
     }
   }

   --42--
```

7.4. Example # 2

```
POST /endpointcostmap/multicost HTTP/1.1
Host: alto.example.com
Accept: multipart/related, application/alto-endpointcost+json,
        application/alto-propmap+json, application/alto-error+json
Content-Length: [TBD]
Content-Type: application/alto-endpointcostparams+json
```

```
{
  "multi-cost-types": [
    {
      "cost-mode": "array",
      "cost-metric": "ane-path"
    },
    {
      "cost-mode": "numerical",
      "cost-metric": "routingcost"
    }
  ],
  "endpoints": {
    "srcs": [ "ipv4:192.0.2.2" ],
    "dsts": [ "ipv4:192.0.2.89",
              "ipv4:203.0.113.45",
              "ipv6:2001:db8::10" ]
  }
}

HTTP/1.1 200 OK
Content-Length: [TBD]
Content-Type: multipart/related; boundary=example-2

--example-2
Content-Type: application/alto-endpointcost+json

{
  "meta": {
    "multi-cost-types": [
      {"cost-mode": "array", "cost-metric": "ane-path"},
      {"cost-mode": "numerical", "cost-metric": "routingcost"}
    ]
  },

  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89":   [[ "ane:L001", "ane:L003", "ane:L004" ], 77],
      "ipv4:203.0.113.45": [[ "ane:L001", "ane:L004", "ane:L005" ], 68],
      "ipv6:2001:db8::10": [[ "ane:L001", "ane:L005", "ane:L007" ], 98]
    }
  }
}

--example-2
Content-Type: application/alto-propmap+json

{
  "property-map": {
```

```
    "ane:L001": { "availbw": 50 },
    "ane:L003": { "availbw": 48 },
    "ane:L004": { "availbw": 55 },
    "ane:L005": { "availbw": 60 },
    "ane:L007": { "availbw": 35 }
  }
}

--example-2--
```

8. Compatibility

8.1. Compatibility with Legacy ALTO Clients/Servers

Legacy ALTO clients SHOULD NOT send queries with the path-vector extension and ALTO servers with this extension SHOULD NOT have any compatibility issue. Legacy ALTO servers do not support cost types with cost mode being "array" and cost metric being "ane-path", so they MUST NOT announce the extended cost types in IRD. Thus, ALTO clients MUST NOT send queries specified in this extension to base ALTO servers according to Section 11.3.2.3 [RFC7285].

8.2. Compatibility with Multi-Cost Extensions

Path Vector is not a testable cost type. Any format of constraints SHOULD NOT be applied to cost type path-vector in order for multi-cost to support the path-vector extension. Specifically,

- Cost type path-vector MUST NOT be included in "testable-cost-types-names" or "testable-cost-types".

- When "testable-cost-types-names" is omitted in the "capabilities" and "testable-cost-types" is omitted in the input parameters, "constraints" or "or-constraints" SHOULD NOT add any format of constraints on cost type path-vector.

8.3. Compatibility with Incremental Update

Without considering the incremental update of multipart/related information, there is no compatibility issue with incremental update extension. Compatibility issue with the incremental update of multipart/related information will be discussed and addressed in the next version.

9. Design Decisions and Discussions

9.1. Provide More General Calendar Extension

Cost Calendar is proposed as a useful ALTO extension to provide the historical cost values for Filtered Cost Map Service and Endpoint Cost Service. Since path vector is an extension to these services, it SHOULD be compatible with Cost Calendar extension.

However, the calendar of a path-vector (Endpoint) Cost Map is insufficient for the application which requires the historical data of routing state information. The (Endpoint) Cost Map can only provide the changes of the paths. But more useful information is the history of network element properties which are recorded in the dependent Network Element Property Map.

Before the Unified Property Map is introduced as an ALTO extension, Filtered Cost Map Service and Endpoint Cost Service are the only resources which require the calendar supported. Because other resources don't have to be updated frequently. But Network Element Property Map as a use case of Unified Property Map will collect the real-time information of the network. It SHOULD be updated as soon as possible once the metrics of network elements change.

So the requirement is to provide a general calendar extension which not only meets the Filtered Cost Map and Endpoint Cost Service but also applies to the Property Map Service.

10. Security Considerations

10.1. Privacy Concerns

We can identify multiple potential security issues. A main security issue is network privacy, as the path-vector information may reveal more network internal structures than the more abstract single-node abstraction. The network should consider protection mechanisms to reduce information exposure, in particular, in settings where the network and the application do not belong to the same trust domain. On the other hand, in a setting of the same trust domain, a key benefit of the path-vector abstraction is reduced information transfer from the network to the application.

The path-vector query may also reveal more information about the application. In particular, the application may reveal all potential transfers sites (e.g., where the data source is replicated, and where the potential replication sites are). The application should evaluate the potential privacy concerns.

Beyond the privacy issues, the computation of the path-vector is unlikely to be cachable, in that the results will depend on the particular requests (e.g., where the flows are distributed). Hence, this service may become an entry point for denial of service attacks on the availability of an ALTO server. Hence, authenticity and authorization of this ALTO service may need to be better protected.

10.2. Resource Consumption on ALTO Servers

The Abstract Network Element Property Map is dynamically enriched when the (Filtered) Cost Map/Endpoint Cost Service is queried of the path-vector information. The properties of the abstract network elements can consume a large amount of resources when cached. So, a time-to-live is needed to remove outdated entries in the Network Element Property Map.

11. IANA Considerations

11.1. ALTO Cost Mode Registry

This document specifies a new cost mode "array". However, the base ALTO protocol does not have a Cost Mode Registry where new cost mode can be registered. This new cost mode will be registered once the registry is defined either in a revised version of [RFC7285] or in another future extension.

11.2. ALTO Cost Metric Registry

A new cost metric needs to be registered in the "ALTO Cost Metric Registry", listed in Table 2.

Identifier	Intended Semantics
ane-path	See Section 5.1.2

Table 2: ALTO Cost Metrics

11.3. ALTO Network Element Property Type Registry

The "ALTO Abstract Network Element Property Type Registry" is required by the ALTO Entity Domain "ane", listed in Table 3.

Identifier	Intended Semantics
availbw	The available bandwidth
delay	The transmission delay

Table 3: ALTO Abstract Network Element Property Types

12. Acknowledgments

The authors would like to thank discussions with Randriamasy Sabine, Andreas Voellmy, Erran Li, Haibin Son, Haizhou Du, Jiayuan Hu, Qiao Xiang, Tianyuan Liu, Xiao Shi, Xin Wang, and Yan Luo.

13. References

13.1. Normative References

[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>.

13.2. Informative References

[I-D.amante-i2rs-topology-use-cases]
Medved, J., Previdi, S., Lopez, V., and S. Amante, "Topology API Use Cases", draft-amante-i2rs-topology-use-cases-01 (work in progress), October 2013.

[I-D.bernstein-alto-topo]
Bernstein, G., Yang, Y., and Y. Lee, "ALTO Topology Service: Uses Cases, Requirements, and Framework", draft-bernstein-alto-topo-00 (work in progress), October 2013.

[I-D.clemm-i2rs-yang-network-topo]
Clemm, A., Medved, J., Tkacik, T., Varga, R., Bahadur, N., and H. Ananthakrishnan, "A YANG Data Model for Network Topologies", draft-clemm-i2rs-yang-network-topo-01 (work in progress), October 2014.

[I-D.ietf-alto-cost-calendar]
Randriamasy, S., Yang, Y., Wu, Q., Lingli, D., and N. Schwan, "ALTO Cost Calendar", draft-ietf-alto-cost-calendar-01 (work in progress), February 2017.

[I-D.ietf-alto-unified-props-new]
Roome, W. and Y. Yang, "Extensible Property Maps for the ALTO Protocol", draft-ietf-alto-unified-props-new-00 (work in progress), July 2017.

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

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

[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>.

[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>.

Authors' Addresses

Greg Bernstein
Grotto Networking
Fremont, CA
USA

Email: gregb@grotto-networking.com

Shiwei Dawn Chen
Tongji University
4800 Caoan Road
Shanghai 201804
China

Email: dawn_chen_f@hotmail.com

Kai Gao
Tsinghua University
Beijing Beijing
China

Email: gaok12@mails.tsinghua.edu.cn

Young Lee
Huawei
TX
USA

Email: leeyoung@huawei.com

Wendy Roome
Nokia/Bell Labs
600 Mountain Ave, Rm 3B-324
Murray Hill, NJ 07974
USA

Phone: +1-908-582-7974
Email: wendy.roome@nokia.com

Michael Scharf
Nokia
Germany

Email: michael.scharf@nokia.com

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

Email: yry@cs.yale.edu

Jingxuan Jensen Zhang
Tongji University
4800 Caoan Road
Shanghai 201804
China

Email: jingxuan.n.zhang@gmail.com