ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: November 4, 2017

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

May 3, 2017

ALTO Extension: Path Vector Cost Mode
draft-ietf-alto-path-vector-00.txt

Abstract

The Application-Layer Traffic Optimization (ALTO) Service has defined network and cost maps to provide basic network information, where the cost maps allow only scalar (numerical or ordinal) cost mode values. This document introduces a new cost mode called path-vector to allow ALTO clients to support use cases such as capacity regions for applications. This document starts with a non-normative example called multi-flow scheduling (or capacity region) to illustrate that ALTO cost maps without path vectors cannot provide sufficient information. This document then defines path-vector as a new cost mode.

Requirements Language

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

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute

working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on November 4, 2017.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction

The ALTO base protocol [RFC7285] is designed for exposing network information through services such as the Network Map service and the Cost Map service. These services use the extreme "single-node"

abstraction, which represents the whole network with a single node and hosts are connected to the node's access ports.

Although the "single-node" abstraction works well for many settings, new use cases, such as inter-datacenter data transfers and scientific high-performance computing data transfers, require additional network information beyond the single-node abstraction, to support application capabilities, in particular, the ability of application flow scheduling.

Specifically, providing network information to support application flow scheduling introduces multiple complexities. First, the underlying assumption of existing ALTO services is single-commodity flows. Hence, given the flow from a source to a destination, ALTO computes the network metrics of the flow and returns them to the application. The metric values for different flows are independent. Application flow scheduling, however, requires network information to compute application-desirable multi-commodity flows, where multiple flows under the control of the same application may share common network bottlenecks. This requirement is beyond the capability of the single-node abstraction adopted by the base ALTO protocol. Second, some flow scheduling problems may consider end-to-end metrics at the same time and thus require multiple costs to be updated simultaneously. Such a requirement, even though already addressed by [I-D.ietf-alto-multi-cost], still needs to be handled very carefully.

To address these complexities in supporting the new flow scheduling use case, this document specifies a path-vector extension to the base ALTO protocol. This extension introduces a new cost type, which uses "path-vector" as cost mode and "ane" (abstract network element) as cost metric. It extends "Filtered Property Map" defined in [I-D.roome-alto-unified-props] to address the issue of scalability and consistency in providing path vectors with fine-grained information. This document also registers a new entity domain, entity specification and properties in the ALTO Entity Domain Registry. Besides, (Filtered) Cost Map and Endpoint Cost Service are extended to support path-vector and flow query format. The concept of "query specific" is put forward to correlate the response information of (Filtered) Cost Map/Endpoint Cost service and query of property map.

The document is organized as follows. Section 3 gives an example of flow scheduling to illustrate the need to introduce cost mode "path-vector" and cost metric "ane". Section 4 gives an overview of the path-vector extension. Section 5 specifies the new cost mode and cost metric, new domain type, entity properties and protocol extensions for (Filtered) Cost Maps and Endpoint Cost Services. Section 6 presents several examples. Section 7 discusses the

compatibility issues with some other proposed ALTO extensions. Section 8 discusses the time to live problem. Section 9 discusses several design decisions. Section 10 and Section 11 discusses about security and IANA Considerations.

2. Terminology

This document uses the following terms: Network Element, Network Element Name, Network Element Property, Network Element Property Map and Path Vector.

- Network Element (NE): A Network Element is an abstract network element, which can be a link, a network switching device, or a set of devices.
- Network Element Name (NEN): The id of a NE. It can be referenced by Cost Map, Filtered Cost Map, Endpoint Cost Service and Property Map.
- Network Element Property (NEP): The property of a NE. It can be the available bandwidth of a link, the delay or other available properties.
- Network Element Property Map (NEP Map): It is a Filtered Property Map resource defined in [I-D.roome-alto-unified-props] and supports "ane" domain in its "domain-types" capability.
- Cost Map, Filtered Cost Map, Endpoint Cost Service: If not explained explicitly, they are the same definition as of [I-D.ietf-alto-multi-cost] by default.
- Path Vector (PV): An array of NENs. It can be used to convey the routing information.

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

Consider the case that routing is given. Then what application-layer traffic optimization will focus on is traffic scheduling 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 each link is 100 Mbps.

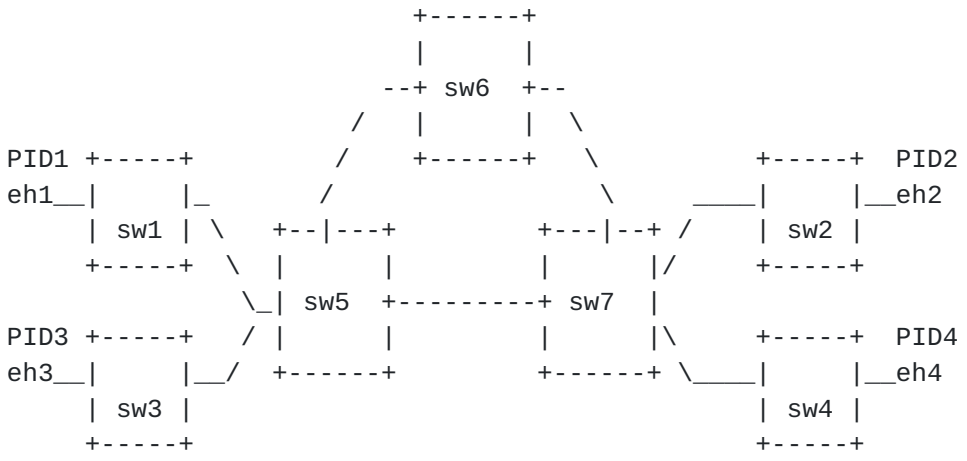

Figure 1: Raw Network Topology.

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

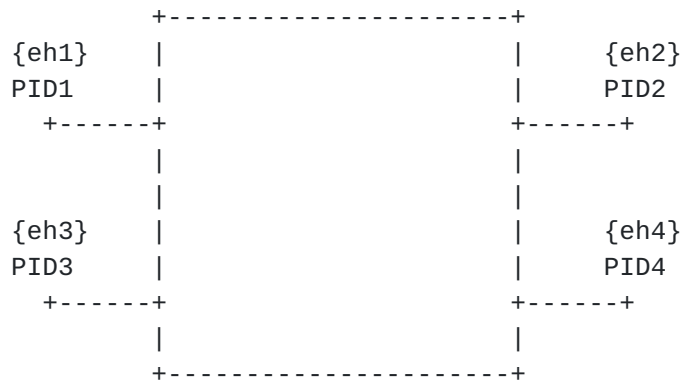

Figure 2: Base Single-Node Topology Abstraction.

Consider an application overlay (e.g., a large data analysis system) which needs to schedule the traffic among a set of end host source-destination pairs, say eh1 -> eh2, and eh3 -> eh4. The application can request a cost map providing end-to-end available bandwidth, using 'availbw' as cost-metric and 'numerical' as cost-mode.

Assume that the application receives from the ALTO server that the bandwidth of eh1 -> eh2 and eh3 ->eh4 are both 100 Mbps. It cannot determine that if it schedules the two flows together, whether it will obtain a total of 100 Mbps or 200 Mbps. This depends on whether

the routing paths of the two flows share a bottleneck in the underlying topology:

- Case 1: If eh1 -> eh2 and eh3 -> eh4 use different paths, for example, when the first uses sw1 -> sw5 -> sw7 -> sw2, and the second uses sw3 -> sw5 -> sw6 -> sw7 -> sw4. Then the application will obtain 200 Mbps.

- Case 2: If eh1 -> eh2 and eh3 -> eh4 share a bottleneck, for example, when both use the direct link sw5 -> sw7, 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, it needs to provide the following new capabilities:

- 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 as possible.

The path-vector extension defined in this document will satisfy 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 Approach

This section presents a non-normative overview of the path-vector extension defined in this document. It assumes the readers are familiar with (Filtered) Cost Map and Endpoint Cost Service defined in [RFC7285], their extensions defined in [I-D.ietf-alto-multi-cost] and Filtered Property Map defined in [I-D.roome-alto-unified-props].

4.1. Path Vector Data Format

The path-vector extension defined in this document introduces "path-vector" as a cost mode, where a path vector between two entities (either PIDs or endpoints) is an array of abstract network elements. Each abstract network element is represented by a name. An abstract network element can be a link, a network switching device, or a set of devices.

One issue of introducing path vectors is how to encode them. The cost entries contained in an ALTO (Filtered) Cost Map or Endpoint Cost Service are formally defined in Section 11.2.3.6 of [RFC7285] to

be any type of JSON value. But the section also suggests that implementations may assume the cost values are numbers unless specifically defined by an extension. This document extends the definition of Cost Map and Endpoint Cost Map to allow the returned cost to be a path vector, which is a JSONArray of Network Element Names.

4.2. Network Element Property Map

The response with only the names of network elements does not provide enough information. To be useful to applications, the extension need to provide desired properties of the abstract network elements as well. This document registers a new entity domain "ane" and entity specifications. The Filtered Property Map defined in [I-D.roome-alto-unified-props] supporting "ane" domain (named Network Element Property Map) is used for ALTO clients to retrieve the properties of abstract network elements.

4.3. Flow Query Format

Current queries of the (Filtered) Cost Map/Endpoint Cost Service are of cross-product format. Such format fails to support lists of single src to single dst costs. So the input scheme of the Filtered Cost Map and Endpoint Cost Service is extended to support the above flow query format.

4.4. Query Specific

Path Vector extension has raised a new occasion that different resources can be associated with the same query. Here, the cost path vector of the (Filtered) Cost Map/Endpoint Cost Service and the properties of the abstract network elements of the Network Element Property Map are generated at the same time and they are related. So, a query-specific attribute "query-id" is introduced in the "vtag" to express such information. Also, the input parameters of the Network Element Property Map is extended to support the query-specific queries.

5. Protocol Extensions

This section formally specifies the path-vector extension which includes the following components: a new cost type, a new entity domain, extensions of (Filtered) Cost Map and Endpoint Cost Map. Below we specify each component in details.

5.1. Cost Type

The path-vector extension defined in this document enriches the cost types defined in Section 6.1 of [RFC7285].

5.1.1. Cost Metric

This document specifies a new cost metric: "ane". It is of type CostMetric as defined in Section 10.6 of [RFC7285]. The cost metric "ane" MUST NOT be used when the cost mode is not "path-vector" unless it is explicitly specified in a future extension. Meanwhile, an ALTO server with path-vector extension MUST support the cost metric "ane".

Cost metric "ane": This cost metric MUST be encoded as the JSONString "ane". When cost metric is "ane", Network Element Names contained in the path vectors MUST be query-specific.

5.1.2. Cost Mode: Path Vector

This document specifies a new cost mode: "path-vector". The path-vector cost mode is of type CostMode as defined in Section 6.1.2 of [RFC7285] and is encoded as the JSONString "path-vector".

A (Filtered) Cost Map resource or Endpoint Cost Service, when queried with this cost mode, MUST return a CostMapData or EndpointCostMapData whose cost value is a JSONArray of type NetworkElementName as specified in Section 5.3.

This cost mode MUST be used with the cost metric "ane" unless it is explicitly specified by a future extension.

5.2. Version Tag

To support the concept of query specific, a new field named "query-id" is introduced to correlate the query information between the (Filtered) Cost Map/Endpoint Cost Service response and the corresponding property map query. The object VersionTag as defined in Section 10.3 of [RFC7285] is extended as follows:

```
object {
  ResourceID   resource-id;
  JSONString   tag;
  [JSONString  query-id;]
} VersionTag;
```

resource-id, tag: As defined in Section 10.3 of [RFC7285].

query-id: A string used to uniquely identify the abstract network elements in the property map.

5.3. Network Element Name

This document also extends [RFC7285] with a new basic data type: NetworkElementName. A NetworkElementName is of type EntityAddr as defined in Section 2.3 of [I-D.roome-alto-unified-props] and is encoded as a JSONString. A NetworkElementName MUST be an EntityAddr of the ANE domain (Section 5.4).

5.4. ANE Domain

This document specifies a new domain in addition to the ones in [I-D.roome-alto-unified-props].

5.4.1. Domain Name

ane

5.4.2. Domain-Specific Entity Addresses

The entity address of ane domain is encoded as a JSON string. The string MUST be no more than 64 characters, and it MUST NOT contain characters other than US-ASCII alphanumeric characters (U+0030-U+0039, U+0041-U+005A, and U+0061-U+007A), the hyphen ('-', U+002D), the colon (':', U+003A), the at sign ('@', code point U+0040), the low line ('_', U+005F), or the '.' separator (U+002E). The '.' separator is reserved for future use and MUST NOT be used unless specifically indicated in this document, or an extension document.

5.5. Filtered Network Element Property Map

A Filtered Network Element Property Map MUST be a Filtered Property Map as defined in Section 5 of [I-D.roome-alto-unified-props].

5.5.1. Accept Input Parameters

This document extends ReqFilteredPropertyMap defined in Section 5.3 of [I-D.roome-alto-unified-props] by introducing an optional field named "query-id". The ReqFilteredPropertyMap is extended as follows:

```
object {
  EntityAddr     entities<1..*>
  PropertyName   properties<1..*>;
  [JSONString    query-id;]
} ReqFilteredPropertyMap;
```

entities, properties: The same as defined in Section 5.3 of [I-D.roome-alto-unified-props].

query-id: Like the "query-id" defined in Section 5.2, the "query-id" here is also a string used to uniquely identify the abstract network elements in the property map.

5.5.2. Capabilities

A Network Element Property Map MUST have capabilities "domain-types" and "prop-types" as defined in Section 4.4 of [I-D.roome-alto-unified-props]. The "domain-types" capability MUST contain domain "ane". And the "prop-types" capability MUST be a subset of property types which are registered with the "ALTO Network Element Property Type Registry" defined in Section 11.4 of this document.

5.5.3. Response

The response is the same as for the Property Map, as defined in Section 4.6 of [I-D.roome-alto-unified-props], except that only the requested entities and properties are returned for Filtered Network Element Map. Examples can be found in Section 6.6 and Section 6.7.

5.6. IRD Extensions

This document extends IRDResourceEntry defined in Section 9.2.2 of [RFC7285] by introducing a new entry named "propertymap". The IRDResourceEntry object is extended as follows:

```
object {
  JSONString uri;
  JSONString media-type;
  [JSONString accepts;]
  [Capabilities capabilities;]
  [ResourceID uses<0..*>;]
  [ResourceID propertymap<0..*>;]
} IRDResourceEntry;
```

uri, media-type, accepts, capabilities, uses: The same as defined in Section 9.2.2 of [RFC7285].

propertymap: A list of resource IDs, defined in the same IRD, that indicates where the specific properties of the returned abstract network elements can be retrieved.

5.7. Cost Map Extensions

This document extends Cost Map defined in Section 11.2.3 of [RFC7285] by returning JSONArray instead of JSONNumber as the cost value, including "vtag" in "meta" field in the response, and adding a new field referencing to the Network Element Property Map.

The media type, HTTP method, capabilities, accept input parameter and uses specifications are unchanged.

5.7.1. Propertymap

If the Cost Map supports the path-vector extension, the field "propertymap" provides a list of resource ids of Network Element Property Map. Each network element property map resource provides properties for the dynamically generated abstract network elements.

5.7.2. Response

The response is the same as defined in Section 11.2.3.6 of [RFC7285] except the follows:

- If the cost mode is "path-vector", the cost is a JSONArray of Network Element Names.

- If the query sent by the client includes cost type path vector, the "vtag" filed defined in Section 5.2 has to be included in the response. And the "query-id" information in "vtag" MUST be provided to clients.

5.8. Filtered Cost Map Extensions

This document extends the Filtered Cost Map defined in Section 4.1 of [I-D.ietf-alto-multi-cost] by specifying details on capabilities, returning JSONArray instead of JSONNumber as the cost value, including "vtag" in "meta" field in the response, and adding a new field referencing to the Network Element Property Map.

The media type, HTTP method and uses specifications are unchanged.

5.8.1. Capabilities

The FilteredCostMapCapabilities object is the same as defined in Section 4.1.1 of [I-D.ietf-alto-multi-cost] expect the follow:

testable-cost-type-names: Cost type "path-vector" MUST NOT appear in the "testable-cost-type-names" field.

5.8.2. Accept Input Parameters

The ReqFilteredCostMap defined in Section 4.1.2 of [I-D.ietf-alto-multi-cost] is extended, the meanings and the constraints of some fields are also extended.

```
  object {
    [CostType cost-type;]
    [CostType multi-cost-types<1..*>;]
    [CostType testable-cost-types<1..*>;]
    [JSONString constraints<0..*>;]
    [JSONString or-constraints<1..*><1..*>;]
    [PIDFilter pids;]
    [PIDFlowFilter pid-flows<1..*>;]
  } ReqFilteredCostMap;

  object {
    PIDName src;
    PIDName dst;
  } PIDFlowFilter;
```

cost-type: If the cost type is path-vector and neither the "testable-cost-type-names" field is provided by the server nor the "testable-cost-types" field is provided by the client, the "constraints" field and the "or-constraints" field SHOULD NOT appear. If not, the ALTO server MUST return an error with error code E_INVALID_FIELD_VALUE. The server MAY include an optional field named "field" in the "meta" field of the response, the value of "field" is "constraints" or "or-constraints". The server MAY also include an optional field named "value" in the "meta" field, the value of "value" is "path-vector". If the "value" field is specified, the "field" field MUST be specified.

multi-cost-types: If "multi-cost-types" includes cost type path-vector, meanwhile, neither the "testable-cost-type-names" field is provided by the server nor the "testable-cost-types" field is provided by the client, the cost type index of path-vector MUST NOT appear in the "constraints" field or the "or-constraints" field. If not, the ALTO server MUST return an error with error code E_INVALID_FIELD_VALUE. The server MAY include an optional field named "field" in the "meta" field of the response, the value of "field" is "constraints" or "or-constraints". The server MAY also include an optional field named "value" in the "meta" field, the value of "value" is "path-vector". If the "value" field is specified, the "field" field MUST be specified.

testable-cost-types: "testable-cost-types" SHOULD NOT include path-vector cost type. If "testable-cost-types" contains path-vector

cost type, the ALTO server MUST return an error with error code E_INVALID_FIELD_VALUE. The server MAY include an optional field named "field" in the "meta" field of the response, the value of "filed" is "testable-cost-types/cost-mode". The server MAY also include an optional field named "value" in the "meta" field, the value of "value" is "path-vector". If the "value" field is specified, the "field" field MUST be specified.

pid-flows: A list of PID src to PID dst for which path costs are to be returned.

Additional requirement is that the Client MUST specify either "pids" or "pid-flows", but MUST NOT specify both.

5.8.3. Propertymap

If the Filtered Cost Map supports the path-vector extension, the field "propertymap" provides a list of resource ids of Network Element Property Map. Each network element property map resource provides properties for the dynamically generated abstract network elements.

5.8.4. Response

The response is the same as defined in Section 4.1.3 of [I-D.ietf-alto-multi-cost] except the follows:

- Whether the "cost-type" field or the "multi-cost-types" field includes cost type path-vector, the cost is a JSONArray of Network Element Names.

- If the query sent by the client includes cost type path vector, the "vtag" filed defined in Section 5.2 has to be included in the response. And the "query-id" information in "vtag" MUST be provided to clients.

5.9. Endpoint Cost Service Extensions

This document extends the Endpoint Cost Service defined in Section 4.2 in [I-D.ietf-alto-multi-cost] by specifying details on capabilities, returning JSONArray instead of JSONNumber as the cost value, including "vtag" in "meta" field in the response, and adding a new field referencing to the Network Element Property Map.

The media type, HTTP method and uses specifications are unchanged.

5.9.1. Capabilities

The same as defined in Section 5.8.1.

5.9.2. Accept Input Parameters

The ReqFilteredCostMap defined in Section 4.1.2 of [I-D.ietf-alto-multi-cost] is extended, the meanings and the constraints of some fields are also extended.

```
object { [CostType cost-type;][CostType multi-cost-types<1..*>;]
[CostType testable-cost-types<1..*>;][JSONString constraints<0..*>;]
[JSONString or-constraints<1..._><1..._>;][EndpointFilter endpoints;]
[EndpointFlowFilter endpoint-flows<1..*>;] } ReqEndpointCostMap;

object { TypedEndpointAddr src; TypedEndpointAddr dst; }
EndpointFlowFilter;
```

cost-type: If the cost type is path-vector and neither the "testable-cost-type-names" field is provided by the server nor the "testable-cost-types" field is provided by the client, the "constraints" field and the "or-constraints" field SHOULD NOT appear. If not, the ALTO server MUST return an error with error code E_INVALID_FIELD_VALUE. The server MAY include an optional field named "field" in the "meta" field of the response, the value of "field" is "constraints" or "or-constraints". The server MAY also include an optional field named "value" in the "meta" field, the value of "value" is "path-vector". If the "value" field is specified, the "field" field MUST be specified.

multi-cost-types: If "multi-cost-types" includes cost type path-vector, meanwhile, neither the "testable-cost-type-names" field is provided by the server nor the "testable-cost-types" field is provided by the client, the cost type index of path-vector MUST NOT appear in the "constraints" field or the "or-constraints" field. If not, the ALTO server MUST return an error with error code E_INVALID_FIELD_VALUE. The server MAY include an optional field named "field" in the "meta" field of the response, the value of "field" is "constraints" or "or-constraints". The server MAY also include an optional field named "value" in the "meta" field, the value of "value" is "path-vector". If the "value" field is specified, the "field" field MUST be specified.

testable-cost-types: "testable-cost-types" SHOULD NOT include path-vector cost type. If "testable-cost-types" contains path-vector cost type, the ALTO server MUST return an error with error code E_INVALID_FIELD_VALUE. The server MAY include an optional field named "field" in the "meta" field of the response, the value of

"filed" is "testable-cost-types/cost-mode". The server MAY also include an optional field named "value" in the "meta" field, the value of "value" is "path-vector". If the "value" field is specified, the "field" field MUST be specified.

endpoint-flows: A list of endpoint src to endpoint dst for which path costs are to be returned.

Additional requirement is that the Client MUST specify either "endpoints" or "endpoint-flows", but MUST NOT specify both.

5.9.3. Propertymap

If the Endpoint Cost Service supports the path-vector extension, the field "propertymap" provides a list of resource ids of Network Element Property Map. Each network element property map resource provides properties for the dynamically generated abstract network elements.

5.9.4. Response

The response is the same as defined in Section 4.2.3 of [I-D.ietf-alto-multi-cost] except the follows:

- Whether the "cost-type" field or the "multi-cost-types" field includes cost type path-vector, the cost is a JSONArray of Network Element Names.

- If the query sent by the client includes cost type path vector, the "vtag" filed defined in Section 5.2 has to be included in the response. And the "query-id" information in "vtag" MUST be provided to clients.

6. Examples

This section lists a series of examples to proceed the multi-flow scheduling like the use case in Section 3.

The recommended workflow is like the following:

6.1. Workflow

- Send the GET request for the whole Information Resource Directory.

- 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 property map.

- Look for the resource of the property map to check whether it supports the desired "prop-types".

- Send request to the (Filtered) Cost Map/Endpoint Cost Service and get the response with "query-id" information. Meanwhile, the properties of the returned abstract network elements are generated in the corresponding Network Element Property Map.

- Send request to the Network Element Property Map with the returned "query-id" information and get the response.

6.2. Information Resource Directory Example

Here is an example of an ALTO server's Information Resource Directory. In this example, both "filtered-multi-cost-map" and "default-endpoint-cost-map" support the multi-cost extension and the path-vector extension. Resource "ane-property-map-availbw" supports property "availbw" and resource "ane-property-map-delay" supports property "delay".

```
  "meta": {
    "cost-types": {
      "pv-ane": {
        "cost-mode": "path-vector",
        "cost-metric": "ane"
      },
      "num-hopcount": {
        "cost-mode": "numerical",
        "cost-metric": "hopcount"
      },
      "num-routingcost": {
        "cost-mode": "numerical",
        "cost-metric": "routingcost"
      },
    }
  },
  "resources": {

    ... NetworkMap Resource ...

    "default-network-map": {
      "uri": "http://alto.example.com/networkmap",
      "media-type": "application/alto-networkmap+json"
    },

    ... Cost Map Resource ...

    "pv-cost-map": {
```

```
      "uri": "http://alto.example.com/pv/ane",
      "media-type": "application/alto-costmap+json",
      "capabilities": {
        "cost-type-names": ["pv-ane"]
      },
      "uses": ["default-network-map"],
      "propertymap": ["ane-property-map-delay",
                      "ane-property-map-availbw"]
    }

    ... Multi-Cost Filtered Map Resource ...

    "filtered-multi-cost-map": {
      "uri": "http://alto.example.com/costmap/multi/filtered",
      "media-type": "application/alto-costmap+json",
      "accepts": "application/alto-costmapfilter+json",
      "uses": ["default-network-map"],
      "capabilities": {
        "max-cost-types": 3,
        "cost-type-names": ["pv-ane",
                            "num-routingcost",
                            "num-hopcount"],
        "cost-constraints": true
      },
      "propertymap": ["ane-property-map-delay",
                      "ane-property-map-availbw"]
    },

    ... Endpoint Cost Service Resource ...

    "default-endpoint-cost-map": {
      "uri": "http://alto.example.com/endpointcost/lookup",
      "media-type": "application/alto-endpointcostmap+json",
      "accepts": "application/alto-endpointcostparams+json",
      "capabilities": {
        "max-cost-types": 3,
        "cost-type-names": ["pv-ane",
                            "num-routingcost",
                            "num-hopcount"],
        "testable-cost-types-names": ["num-hopcount",
                                      "num-routingcost"],
      },
      "propertymap": ["ane-property-map-delay"]
    }

    ... Network Element Property Map Resource ...

    "ane-property-map-delay": {
```

```
        "uri": "http://alto.example.com/propmap/lookup/delay",
        "media-type": "application/alto-propmap+json",
        "accepts": "application/alto-propmapparams+json",
        "capabilities": {
          "domain-types": ["ane"],
          "prop-types": ["delay"]
        }
      }

      "ane-property-map-availbw": {
        "uri": "http://alto.example.com/propmap/lookup/availbw",
        "media-type": "application/alto-propmap+json",
        "accepts": "application/alto-propmapparams+json",
        "capabilities": {
          "domain-types": ["ane"],
          "prop-types": ["availbw"]
        }
      }
    }
```

6.3. Cost Map Example

Here is an example of the Cost Map request for path-vector and the corresponding response. In response, the extended "vtag" field is included in "meta" to provide "query-id" information.

```
GET /costmap/pv/ane HTTP/1.1
Host: alto.example.com
Accept: application/alto-costmap+json,application/alto-error+json

HTTP/1.1 200 OK
Content-Length: [TBD]
Content-Type: application/alto-costmap+json

{
  "meta" : {
    "dependent-vtags" : [
      { "resource-id": "default-network-map",
        "tag": "5eb2cb7f8d63a9fab71d9b34cbf763436315542f"
      }
    ],
    "vtag": [
      { "resource-id": "pv-cost-map",
        "tag": "aef527ca2eb7a7566db0597407893e3f8eb1d9dff",
        "query-id": "query0"
      }
    ],
    "cost-type" : {"cost-mode"  : "pv",
                   "cost-metric": "ane"
    }
  },
  "cost-map" : {
      "PID1": { "PID2": ["ane:L001", "ane:L003"],
                "PID3": ["ane:L002", "ane:L004"] },
      "PID2": { "PID1": ["ane:L003", "ane:L002"],
                "PID3": ["ane:L003", "ane:L004"] },
      "PID3": { "PID1": ["ane:L004", "ane:L002"],
                "PID2": ["ane:L004", "ane:L003"] }
  }
}
```

6.4. Multi-Cost Filtered Cost Map Example

The following example presents the request and response of "filtered-multi-cost-map". In this example, the client is interested in the path-vector and numerical routing cost information. The client uses "or-constraints" but all the results satisfy the conditions. In resposne, the extended "vtag" field is included in "meta" to provide "query-id" information.

```
POST /costmap/multi/filtered HTTP/1.1
Host: alto.example.com
Accept: application/alto-costmap+json,application/alto-error+json
Content-Length: [TBD]
```

```
Content-Type: application/alto-costmapfilter+json

{
  "multi-cost-types": [
    {
      "cost-mode": "path-vector",
      "cost-metric": "ane"
    },
    {
      "cost-mode": "numerical",
      "cost-metric": "routingcost"
    }
  ],
  "testable-cost-types": [
    { "cost-mode": "numerical", "cost-metric": "routingcost" },
    { "cost-mode": "numerical", "cost-metric": "hopcount" }
  ],
  "or-constraints": [
    ["[0] ge 50", "[1] le 60"]
  ],
  "pid-flows": [
    { "src": "PID1", "dst": "PID2" },
    { "src": "PID2", "dst": "PID3" }
  ]
}

HTTP/1.1 200 OK
Content-Length: [TBD]
Content-Type: application/alto-costmap+json

{
  "meta": {
    "dependent-vtags": [
      {
        "resource-id": "default-network-map",
        "tag": "75ed013b3cb58f896e839582504f622838ce670f"
      }
    ],
    "vtag": [
      {
        "resource-id": "filtered-multi-cost-map",
        "tag": "27612897acf278ffu3287c284dd28841da78213",
        "query-id": "query1"
      }
    ]
  }
  "cost-type": {},
  "multi-cost-types": [
```

```
        { "cost-mode": "path-vector", "cost-metric": "ane" },
        { "cost-mode": "numerical",   "cost-metric": "routingcost"}
      ],

      "cost-map": {
        "PID1": {
          "PID2": [ [ "ane:L001", "ane:L003" ], 55 ]
        },
        "PID2": {
          "PID3": [ [ "ane:L003", "ane:L004" ], 60 ]
        }
      }
    }
```

6.5. Endpoint Cost Service Example

If the ALTO client expects the routing state information between endpoints, it can also query the "default-endpoint-cost-map" resource which supports path vector. In resposne, the extended "vtag" field is included in "meta" to provide "query-id" information.

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

{
  "multi-cost-types": [
    { "cost-mode": "path-vector", "cost-metric": "ane" },
    { "cost-mode": "numerical",   "cost-metric": "hopcount" }

  ],
  "endpoint-flows": [
    { "src": "ipv4:192.0.2.2",     "dst": "ipv4:203.0.113.45"  },
    { "src": "ipv4:192.0.2.89",    "dst": "ipv4:198.51.100.34" },
    { "src": "ipv4:194.2.3.67",    "dst": "ipv4:202.56.54.230" },
    { "src": "ipv4:203.32.56.102", "dst": "ipv4:202.76.89.103" }
  ]
}

HTTP/1.1 200 OK
Content-Length: [TBD]
Content-Type: application/alto-endpointcost+json

{
  "meta": {
    "vtag": [
```

```
        {
          "resource-id": "default-endpoint-cost-map",
          "tag": "73182ffa829dc28c218a1823bb1293dea232885",
          "query-id": "query2"
        }
      ],
      "cost-type": {},
      "multi-cost-types": [
        { "cost-mode": "path-vector", "cost-metric": "ane" },
        { "cost-mode": "numerical", "cost-metric": "hopcount" }
      ]
    },
    "endpoint-cost-map": {
      "ipv4:192.0.2.2": {
        "ipv4:203.0.113.45": [ [ "ane:L10", "ane:L11",
                                 "ane:L15", "ane:L16",
                                 "ane:L17", "ane:L13" ], 100]
      },
      "ipv4:192.0.2.89": {
        "ipv4:198.51.100.34": [ [ "ane:L14", "ane:L11",
                                  "ane:L12", "ane:L18",
                                  "ane:L19" ], 124]
      },
      "ipv4:194.2.3.67": {
        "ipv4:202.56.54.230": [ [ "ane:L21", "ane:L15",
                                  "ane:L16", "ane:L17",
                                  "ane:L18", "ane:L20"], 78]
      },
      "ipv4:203.32.56.102": {
        "ipv4:202.76.89.103": [ [ "ane:L22", "ane:L16",
                                  "ane:L23"], 168]
      }
    }
  }
```

6.6. Network Element Property Map Example # 1

After the client send the query to the Multi-Cost Filtered Cost Map "filtered-multi-cost-map" and get the response with query-id "query1" (See Section 6.4). The client send request to the "ane-property-map-availbw" with query-id "query1" and get the response.

```
POST /propmap/lookup/availbw HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
Content-Length: [TBD]
Content-Type: application/alto-propmapparams+json

{
  "query-id": "query1",
  "entities" : [ "ane:L001",
                 "ane:L003",
                 "ane:L004" ],
  "properties" : [ "availbw" ]
}

HTTP/1.1 200 OK
Content-Length: [TBD]
Content-Type: application/alto-propmap+json

{
  "property-map": {
    "ane:L001": { "availbw": "50" },
    "ane:L003": { "availbw": "70" },
    "ane:L004": { "availbw": "80" }
  }
}
```

6.7. Network Element Property Map Example # 2

After the client send the query to the Endpoint Cost Service "default-endpoint-cost-map" and get the response with query-id "query2" (See Section 6.5). The client send request to the "ane-property-map-delay" with query-id "query2" and get the response.

```
POST /propmap/lookup/delay HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
Content-Length: [TBD]
Content-Type: application/alto-propmapparams+json

{
  "query-id": "query2",
  "entities" : [ "ane:L11",
                 "ane:L15",
                 "ane:L16",
                 "ane:L18" ],
  "properties" : [ "delay" ]
}

HTTP/1.1 200 OK
Content-Length: [TBD]
Content-Type: application/alto-propmap+json

{
  "property-map": {
    "ane:L11": { "delay": "25" },
    "ane:L15": { "delay": "40" },
    "ane:L16": { "delay": "60" },
    "ane:L18": { "delay": "10" }
  }
}
```

7. Compatibility

7.1. Compatibility with Legacy ALTO Clients/Servers

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

7.2. Compatibility with Multi-Cost Extensions

Path Vector is not a testable cost type. So two conditions have to be satisfied when multi-cost supports path-vector extension. Any format of constraints SHOULD NOT be applied to cost type path-vector.

- 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.

7.3. Compatibility with Incremental Update

There is no compatibility issue with incremental update extension.

8. Time to live

The 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.

9. Design Decisions and Discussions

9.1. Path Vector or Path Graph

When we introduce the "path-vector" as a cost mode in the Cost Map, an unavoidable problem is how to handle multipath. Because a PID is a group of endpoints, it is common that there are multiple paths between two PIDs. The valid routing state information is all of the accessible paths. So in this scenario, the Cost Map Resource SHOULD provide the cost values including of the multiple paths.

A natural solution is to provide an array of path vectors as the cost value. Every path vector in this array means an accessible path between the source PID and the destination PID. It is different from the solution of the path vector extension which provides an array of network elements. So it requires to introduce a different cost mode. This document proposes this new cost mode named "path-graph".

However, the "path-graph" will increase the complexity of the Cost Map Response. Since the applications select ALTO as the protocol to get the network information rather than other topology-based solution such as I2RS, the major reason should be the simplicity. If we provide "path-graph" for each PID pairs, the ALTO client has to handle the complex data structure.

What's more, the "path-vector" is powerful enough to express multiple paths. The simple solution is to list the network elements of all accessible paths in a single path vector. This solution will lose the information about paths. Another solution is to define the path as a new type of network elements. In this way, the path vector can

provide an array of paths. Each element of this array contains a path vector of network elements in the Network Element Property Map.

So in this document, we just introduce "path-vector" as the only required cost mode for routing state information.

9.2. 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.

9.3. Snapshot and real-time update

As the suggested workflow mentioned in Section 6.1, the properties of the abstract network elements are computed ahead of the query of these properties. So the properties of abstract network elements are likely to be snapshot rather than realt-time value when queried.

10. Security Considerations

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.

11. IANA Considerations

11.1. ALTO Cost Mode Registry

This document specifies a new cost mode "path-vector". 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 1.

Identifier	Intended Semantics
ane	See Section 5.1.1

Table 1: ALTO Cost Metrics

11.3. ALTO Entity Domain Registry

As proposed in Section 9.2 of [I-D.roome-alto-unified-props], "ALTO Entity Domain Registry" is requested. Besides, a new domain is to be registered, listed in Table 2.

Identifier	Entity Address Encoding	Hierarchy & Inheritance
ane	See Section 5.4.2	None

Table 2: ALTO Entity Domain

11.4. ALTO Network Element Property Type Registry

The "ALTO 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 Network Element Property Types

12. Acknowledgments

The authors would like to thank discussions with 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, <http://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.gao-alto-fcs]
Gao, K., Zhang, J., Wang, J., Xiang, Q., and Y. Yang, "ALTO Extension: Flow-based Cost Query", draft-gao-alto-fcs-01 (work in progress), March 2017.

[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-incr-update-sse]
Roome, W. and Y. Yang, "ALTO Incremental Updates Using Server-Sent Events (SSE)", draft-ietf-alto-incr-update-sse-03 (work in progress), September 2016.

[I-D.ietf-alto-multi-cost]
Randriamasy, S., Roome, W., and N. Schwan, "Multi-Cost ALTO", draft-ietf-alto-multi-cost-07 (work in progress), March 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.

[I-D.roome-alto-unified-props]
Roome, W., "Extensible Property Maps for the ALTO Protocol", draft-roome-alto-unified-props-01 (work in progress), July 2016.

[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, <http://www.rfc-editor.org/info/rfc7285>.

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